OKTATÁSI HIVATAL

NAT 2020

9

Digitális kultúra

tankönyv

NYOMDAI ELŐKÉSZÍTÉS ALATT

A kiadvány 2020. 06. 11-től 2025. 08. 31-ig tankönyvi engedélyt kapott a TKV/3178–7/2020. számú határozattal. A tankönyv megfelel a Kormány 5/2020 (I.31.) Korm. rendelete a Nemzeti alaptanterv kiadásáról, bevezetéséről és alkalmazásáról szóló 110/2012. (VI.4.) Korm. rendelet módosításáról megnevezésű jogszabály alapján készült Kerettanterv a középiskola 9. évfolyama számára megnevezésű kerettanterv Informatika tantárgy előírásainak.

A tankönyvvé nyilvánítási eljárásban közreműködő szakértő: Györgyi Tamás

Tananyagfejlesztők: Varga Péter, Jeneiné Horváth Kinga, Reményi Zoltán, Farkas Csaba, Takács Imre, Siegler Gábor, Abonyi-Tóth Andor
Kerettantervi szakértő: Siegler Gábor
Lektor: Farkasfalvy Judit
Fedélterv: Slezák Ilona
Fotók: schutterstock

ISBN 978-615-81539-5-9

Oktatási Hivatal
1055 Budapest, Szalay utca 10–14.
Telefon: (+36-1) 374-2100
E-mail: tankonyv@oh.gov.hu

A kiadásért felel: dr. Gloviczki Zoltán elnök

Raktári szám: OH-DIG09TA
Tankönyvkiadási osztályvezető: Horváth Zoltán Ákos
Műszaki szerkesztő: Görög Istvánné
Nyomdai előkészítés: WOW Stúdió Kft.
Terjedelem: 20,38 (A/5) ív, tömeg: 610 gramm

1. kiadás, 2020

Ez a tankönyv a Széchenyi 2020 Emberi Erőforrás Fejlesztési Operatív Program EFOP-3.2.2-VEKOP-15-2016-00001 számú, „A köznevelés tartalmi szabályozóinak megfelelő tankönyvek, taneszközök fejlesztése és digitális tartalomfejlesztés” című projektje keretében készült.
A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

Nyomtatta és kötötte:
Felelős vezető:
A nyomdai megrendelés törzsszáma:

SZÉCHENYI 2020

MAGYARORSZÁG KORMÁNYA

Európai Unió
Európai Szociális Alap

BEFEKTETÉS A JÖVŐBE

Tartalom

Előszó

Kedves Diákok!

Talán egyszer majd a mai korszakot, amelynek egyik fontos ismérve a digitális átalakulás, kattintottegér-kornak fogják hívni… A digitális átalakulás ugyanis nemcsak a termelési eszközöket változtatta meg. Nem csupán átalakulnak vagy eltűnnek régi szakmák, miközben olyan új, számítógéppel segített munkahelyek jelennek meg, amelyeket ma még el sem tudunk képzelni. Teljesen megváltozott a kommunikáció stílusa (e-mail, chat, sms sajátos nyelvezete), új szimbólumok terjedtek el, 🙂☹👍, és gyakran a nagymamák meg sem értik az unokák szóhasználatát, életükben az újfajta kommunikáció szerepét. A digitális kultúra tantárgy így nem csupán az informatika tudományterületét tárgyalja, hanem fontos szerepet kap benne megváltozott kultúránk, az új eszközök felelősségteljes használata is.

A tankönyv első részében a *digitális írástudással* foglalkozunk. Ennek keretében gyakorlatot szerzünk a dokumentumok készítését segítő szoftverek használatában, új dokumentumokat hozunk létre, alakítunk át, illetve megismerkedünk a stílusok használatával. Foglalkozunk az adott probléma megoldásához szükséges raszter- és vektorgrafikus ábrák létrehozásával, szerkesztésével. Gyakorlatot szerzünk a fotó-, hang- és videószerkesztésben, a bemutatókészítő eszközök használatában, és felhasználjuk a létrehozott multimédiás elemeket új dokumentumok készítéséhez. Megismerkedünk a dokumentumok tartalomkezelő rendszerbe történő elhelyezésével és szerkesztésével, a HTML-formátumú dokumentumok szerkezeti elemeivel, valamint a CSS használatának alapelveivel.

A második részben az *informatikai eszközökkel és módszerekkel történő problémamegoldással* kapcsolatos ismereteinket bővítjük. Az előző években használt robot- és/vagy blokkprogramozás után a strukturált programozás alapfogalmait, módszereit vezetjük be a problémák megoldásához. Egy magas szintű, széles körben elterjedt, de egyszerű programozási nyelvet, a Pythont használjuk a gyakorlatban. Fontos hangsúlyoznunk, hogy a választott nyelv csupán egy eszköz, amelyet most vagy később más elterjedt nyelv(ek) válthatnak fel. Megismerkedünk az adatkezelés alapfogalmaival, tovább bővítjük a táblázatkezelési ismereteinket.

Ezután a *mobil- és az információs hálózatok* használatára térünk át. Rendszerezzük az eddigi ismereteket, és készülünk az internet tudatos használatára. Megismerjük az e-világ elvárásait, az online kommunikáció lehetőségeit, biztonsági és jogi kérdéseit, valamint foglalkozunk a digitális személyazonosság és az információhitelesség fogalmával is.

Könyvünk utolsó fejezetében *az informatikai eszközök használatáról* olvashatunk. Ezt a részt ne önállóan dolgozzuk fel, hanem tartalmi elemeit kisebb részletekben, a többi témakör tárgyalásakor megjelenő fogalmakhoz kapcsolva nézzük át, lapozzunk majd időnként oda!

A tankönyv szerves részét képezik az elérhető elektronikus anyagok, fájlok, amelyek a https://www.tankonyvkatalogus.hu/site/kiadvany/OH-DIG09TA oldalról tölthetők le.

Jó munkát és a tankönyv eredményes használatát kívánjuk:

a szerzők

Szövegszerkesztés

Mielőtt elkezdenénk…

Egy kis történelem

Az *írás* gondolataink rögzítése. Az évezredek során különböző fajtái váltották egymást (képírás, ideogramok, hangírás stb.) és különböző adathordozókra (anyagtábla, pergamen, papír stb.) különböző íróeszközökkel rögzítették (nádszálból készült stílus, vágott hegyű toll, töltőtoll, golyóstoll stb.) azt. Hamar felmerült az igény a dokumentumok sokszorosításának gépesítésére is: nyomdagépeket az ókori Kínában már Kr. u. 200 körül is használtak.

A *nyomdagépek* Európában a középkorban terjedtek el. Kezdetben egy falap felületéből faragták ki az oldal tükörképét, és ezzel pecsételtek, ám így minden könyv minden lapjához új nyomólemezt kellett készíteni. *Johannes Gutenberg* nagy találmánya az 1430-as években az volt, hogy mozgatható, fémből készült betűket alkalmazott.

▶ Gutenberg nyomdagépe

A XIX. század végétől az irodák egyik jellemző eszköze volt az *írógép*. Az írógép – a nyomdához hasonló elven – az előre kivésett betűnegatívokat egy festékszalagon át ütötte a papírra. A billentyűzeten használt betűkiosztás is az írógépek öröksége. Az írógépeket az 1980-as években felváltották előbb a *szövegszerkesztő gépek*, majd a *számítógépes szövegszerkesztő programok*.

A szövegszerkesztő programok lehetővé teszik a szöveg beírását, megformázását, mentését, nyomtatását. A programok funkciói hasonlítanak egymásra, hiszen maga a szövegszerkesztés is évszázadok alatt alakult ki. Fontos tehát, hogy a szövegszerkesztés elsajátítása során a hangsúlyt *az általános alapelvekre és funkciókra helyezzük,* így könnyen át tudunk állni egyik programról a másikra vagy az adott program egy újabb változatára.

Ma többféle szövegszerkesztő programmal találkozhatunk, ilyenek például a *Microsoft Word* vagy a *LibreOffice Writer*. Az „asztali” programok mellett azonban terjednek a telefonos alkalmazások (appok, applikációk), valamint az interneten át használható, közös munkát is támogató online megoldások, például a *Google Docs*.

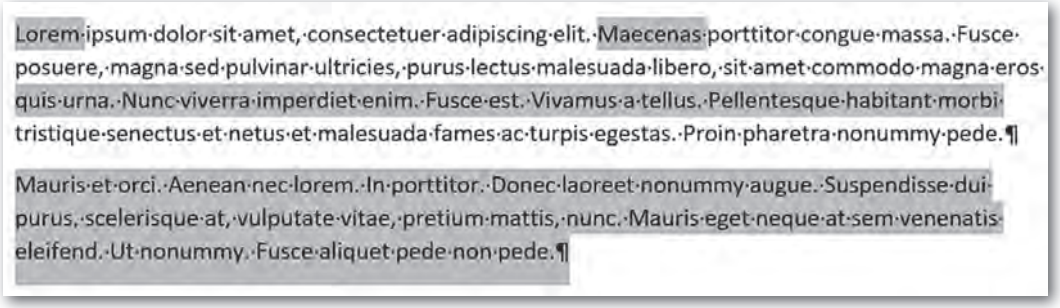
Lorem ipsum

Ha csupán egy oldal leendő kinézetét, a betűtípusokat és az elrendezést szeretnénk szemléltetni, általában zavar bennünket a látható szöveg tartalma, ugyanis ösztönösen elkezdjük olvasni. Ezért már az 1500-as években egy ismeretlen nyomdász egy látszólag értelmetlen mintaszöveget készített, amikor összeállította a saját nyomdakészletét:

„Lorem ipsum dolor sit amet, consectetur adipisicing elit…"

Csak jóval később derült ki – miután ez a megoldás széles körben elterjedt –, hogy a szöveg eredetileg Cicero Kr. e. 45-ben írt *A legfőbb jóról és rosszról* című műve néhány bekezdésének véletlenszerűen összevágott szavaiból készült.

Tippek. Véletlen szöveget gyakran a szövegszerkesztő programok segítségével is elő tudunk állítani. Latinos hangzású, Lorem ipsum kezdetű szöveget a *LibreOffice Writerben* például a **Lorem** szó beírásával és az F3 funkciógomb lenyomásával készíthetünk. A *Microsoft Wordben* pedig egy 12 bekezdésből, bekezdésenként 8 mondatból álló véletlen szöveget az **=Lorem(12,8)** szöveg beírásával generálhatunk. Ha a *Microsoft Wordben* magyar nyelvű ékezetes szöveget szeretnénk előállítani, akkor az **=Rand(12,8)** „utasítást" kell kiadnunk.


Lorem·ipsum·dolor·sit·amet,·consectetuer·adipiscing·elit.·Maecenas·porttitor·congue·massa.·Fusce·posuere,·magna·sed·pulvinar·ultricies,·purus·lectus·malesuada·libero,·sit·amet·commodo·magna·eros·quis·urna.·Nunc·viverra·imperdiet·enim.·Fusce·est.·Vivamus·a·tellus.·Pellentesque·habitant·morbi·tristique·senectus·et·netus·et·malesuada·fames·ac·turpis·egestas.·Proin·pharetra·nonummy·pede.¶

Mauris·et·orci.·Aenean·nec·lorem.·In·porttitor.·Donec·laoreet·nonummy·augue.·Suspendisse·dui·purus,·scelerisque·at,·vulputate·vitae,·pretium·mattis,·nunc.·Mauris·eget·neque·at·sem·venenatis·eleifend.·Ut·nonummy.·Fusce·aliquet·pede·non·pede.¶

► Generáljunk egy véletlenszöveget a Lorem utasítás használatával, és próbáljuk ki rajta a kijelölés minél több módját! Hogyan készülhetett az ábrán látható kijelölés?

A formázandó szöveg kijelölése

Ha egy szövegrész formátumát módosítani szeretnénk, meg kell adnunk, hogy melyik szövegrészre gondolunk. A különböző programok sokféle lehetőséget kínálnak, tekintsük át a leggyakoribb megoldásokat!

Azt a szót vagy azt a bekezdést, ahol a kurzor áll, nem kell kijelölni, ha a szó betűformátumát vagy a bekezdés bekezdésformátumát szeretnénk módosítani.

Általános módszer, hogy a kijelölendő részen *az egér lenyomott bal gombjával* végighúzzuk a kurzort. Így akár nem összefüggő részeket is kijelölhetünk, ha közben nyomjuk a CTRL gombot is.

Hasznos eszköz lehet *a kijelölés bővítése.* Ha többször kattintunk az egér bal gombjával, akkor egyre nagyobb szövegegységet jelölhetünk ki. Például dupla kattintás az adott szót, háromszoros kattintás az adott mondatot *(LibreOffice Writer)* vagy az adott bekezdést *(Microsoft Word)* jelöli ki.

A *teljes dokumentum kijelölését* a CTRL + A billentyűkombinációval (A: all, jelentése minden) billentyűzetkombinációval tehetjük meg.

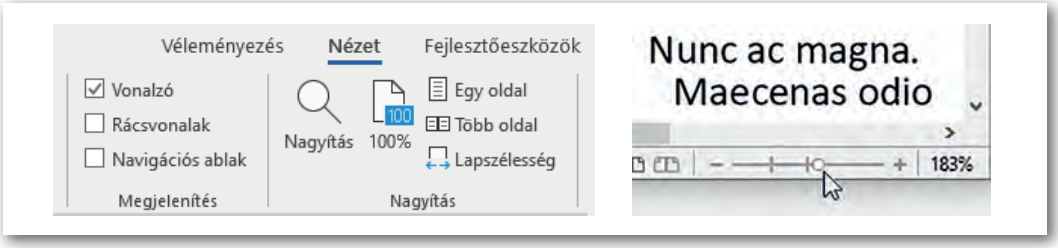
A kijelölés egér nélkül is lehetséges. Például szöveg kijelölésére használhatjuk *a SHIFT gomb lenyomása mellett a kurzormozgató* billentyűket is, illetve a kijelölés bővítése történhet az F8 funkciógomb ismételt lenyomásával is *(Microsoft Word)*.

A munkakörnyezet kialakítása

Jobban átlátjuk **a szöveg tagolását,** ha a képernyőn figyelemmel tudjuk kísérni, hogy hol nyomtuk le a szóközt, a tabulátor gombot vagy a bekezdésjelet (ENTER). Ezeket **a nem nyomtatható karaktereket speciális szimbólumokkal jeleníthetjük meg** a megfelelő parancs kiadásával, többnyire egy ¶ feliratú gombra kattintva.

Amikor beírjuk a szöveget, akkor azt az ablak teljes szélességében szeretnénk látni, míg formázás közben látnunk kell azt is, hogy a formázandó szövegrész hol helyezkedik el az oldalon. *A szöveg nagyítását* ennek megfelelően többféle módon, *például a menüben vagy a jobb alsó sarokban lévő csúszkával módosíthatjuk*.

Az egyes alakzatok helyének pontos megadását megkönnyíti, ha a képernyőn látjuk a *vonalzót*. A vonalzó bekapcsolását általában a *Nézet* menüben tehetjük meg, és a szöveg fölött és bal oldalán jelenik meg.



▶ A szöveg nagyítása és a vonalzó bekapcsolása menüből (Microsoft Word, balra), illetve a nagyítás állítása csúszkával (LibreOffice Writer, jobbra)

A szöveg bevitele és javítása

Ha új dokumentumot kezdünk, és nem kapjuk meg készen a nyers szöveget, akkor első feladatunk a szöveg beírása. Nem szerencsés eközben a szöveg formázását is elkezdeni, elegendő csupán a szöveg helyes bevitelére koncentrálnunk.

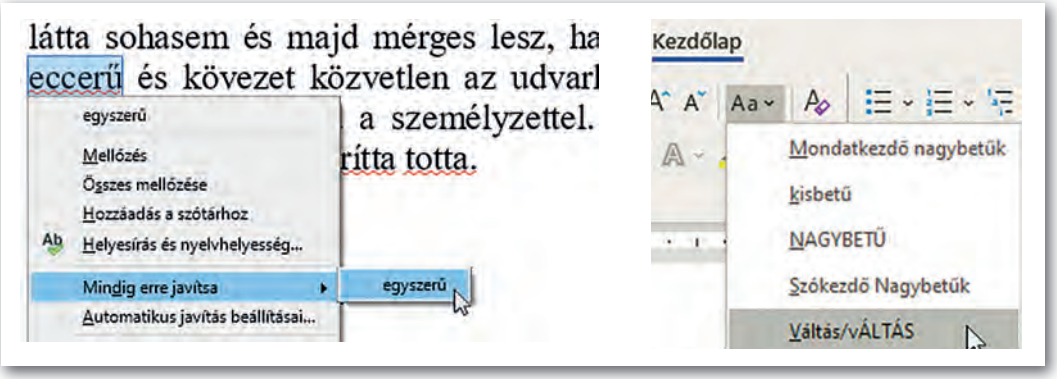
A szöveg beírása során a legfontosabb alapelv, **hogy ne használjuk indokolatlanul az ENTER és a szóköz gombot.** Az ENTER gombot csak a bekezdés végén nyomjuk le, a bekezdés sorokra tördelését bízzuk a szövegszerkesztő programra. A szavak közé pedig mindig csak egy-egy szóközt írjunk, ne használjuk a szóköz gombot igazításra, például szavak, címek esetében.

Csak a szöveg beírása után célszerű a gépelési és a nyelvi hibákat javítani.

A gépelési hibák közül egyet érdemes külön is kiemelni. Ha véletlenül lenyomjuk a CAPS LOCK gombot (CAPS: Capital letters, jelentése: nagybetűk, LOCK jelentése: zár), a program attól kezdve kisbetűk helyett nagybetűkkel ír, és fordítva. Szerencsére ennek javítása egyszerű: a hibásan írt szövegrészt kijelöljük, majd a menü megfelelő pontjával a kis- és nagybetűs formátumot megcseréljük.

A hibák javításában segít a szövegszerkesztő program is. Például piros hullámos vonallal aláhúzza a helyesírási, dupla kék vonallal a nyelvi hibákat. Ha ilyenkor az egér jobb gombjával a megjelölt szövegre kattintunk, a helyi menüben választhatunk a szövegszerkesztő program javaslatai közül. Általában arra is van lehetőségünk, hogy az adott hiba kezelésére „megtanítsuk” a programot, így azt a későbbiekben már beíráskor automatikusan javítja.

Sajnos **a nyelvi ellenőrző programok** nem készülhetnek fel minden lehetőségre, azért a programok **által javasolt módosításokat érdemes kritikával fogadnunk**.



► Az „eccerű” szót helytelenül írtuk, a program javaslatot ad a javításra (LibreOffice Writer)

► Kis- és nagybetű megcserélése (Microsoft Word)

Ebben a fejezetben főleg két asztali szövegszerkesztő programot, a *Microsoft Wordöt* és a *LibreOffice Writert* fogjuk példaként bemutatni. A funkciók elérésére vonatkozó leírásban, ott, ahol a két szövegszerkesztő programot eltérően kell használni, a *Microsoft Wordre* vonatkozó megoldás szerepel, mögötte pedig zárójelben a *LibreOffice Writerben* használt. Például: „Táblázatot a *Beszúrás* > *Táblázat* (illetve a *Táblázat* > *Táblázat beszúrása*) menüponttal illeszthetünk be.”

Feladatok

1. A szöveg terjedelmét általában a leütések számával adják meg. A leütések számába a betűkön és írásjeleken túl beletartozik a szóköz is. Hány leütésből áll az általunk generált szöveg? Milyen módon tudjuk ezt kideríteni az általunk használt szövegszerkesztőben?
2. Jelenítsük meg a vonalzót, és olvassuk le róla a margók nagyságát, valamint a főszöveg által elfoglalt terület *(a szövegtükör)* méreteit!
3. Szövegszerkesztőnek tekinthető-e a Microsoft Windowsban található Jegyzettömb? Válaszunkat indokoljuk!
4. Rejtő Jenő *Piszkos Fred a kapitány* című regényében Fülig Jimmy naplót vezet, ám rendkívül rossz a helyesírása. Töltsük le a napló egy részletét a tankönyv weblapjáról, és javítsuk ki benne a hibákat!

Betűformázás

1. példa: Apróhirdetés

Készítsük el az ábrán látható hirdetést! A szöveget gépeljük be vagy töltsük le a tankönyv weblapjáról!

CSALÁDI HÁZ ELADÓ!

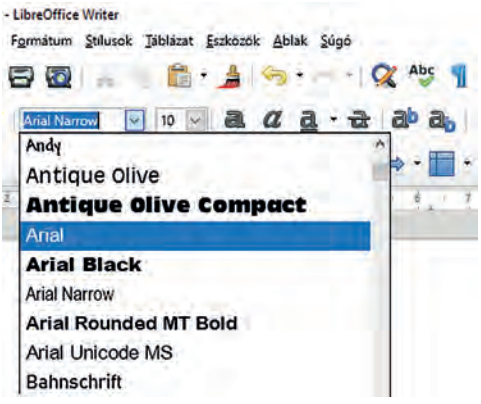
Talpas belterületén, az iskola mellett, *kétszintes családi ház* eladó. A 120 m^2-es házhoz 600 m^2-es telek tartozik. Az udvaron 45 m^2 alapterületű, utcára nyíló, **vállalkozásra alkalmas** épület is van.

Érdeklődni *kizárólag személyesen*: SONKA SÁMUEL, Talpas, Egyenes u. 3.

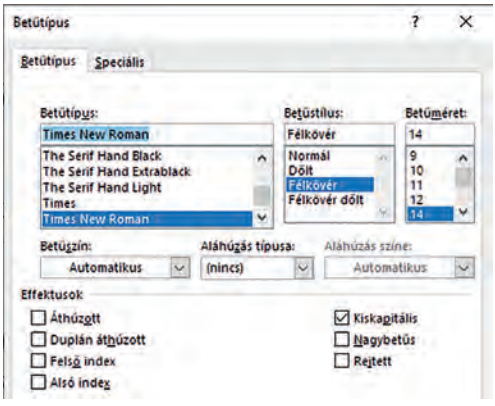
A hirdetés szövegében három különböző **betűtípust** látunk. A hirdetés címe figyelemfelhívó, *díszes* betűtípusú (példánkban Algerian), a ház leírása jól olvasható, változó vonalvastagságú, *talpas* betűtípusú (Times New Roman), míg a kapcsolattartó elérhetősége *talp nélküli,* állandó vonalvastagságú betűtípussal (Arial) jelenik meg.

A ház leírásában a hirdető a fontosnak tartott részeket azonos típusú, de eltérő **stílusú** betűkkel emelte ki: *dőlt, aláhúzott, félkövér*. A m^2-ben a kitevő felső indexben van.

A három bekezdésben a hirdető eltérő **betűméretet** alkalmazott: a cím betűmérete a legnagyobb (pl. 16 pontos) a ház leírása kisebb (pl. 14 pontos), míg az elérhetőség a legkisebb (pl. 10 pontos).



► A betűformátumok beállítása a menüben (LibreOffice Writer)



► A Betűtípus dialógusdobozban minden betűformátum elérhető (Microsoft Word).

Érdekes, hogy a szövegben mind a címet, mind a hirdető nevét nagybetűsnek látjuk, pedig a nyers szövegben a címben csak az első betű, a névben pedig csak a két kezdőbetű nagy. Az első esetben olyan betűtípust alkalmaztunk, amelyben nincsenek kisbetűk, a másodikban pedig olyan betűstílust (kiskapitális), amelyben a kisbetűk kisebb méretű nagybetűnek látszanak.

A betűformátumok áttekintése

A betűformátumok beállításánál leggyakrabban a betűk típusát, stílusát, méretét és színét szoktuk módosítani.

A betűtípus azonos grafikai elvek szerint megtervezett ábécé. A betűtípust a neve azonosítja, például Times New Roman, Courier, Calibri, Old English Text.

Egy adott betűtípus változatait betűstílusoknak nevezik. A legtöbb betűtípushoz külön terveznek a normál mellett félkövér, dőlt és félkövér dőlt változatot. Ezeket a szövegszerkesztők a grafikus megjelenítés során tovább bővítik, így áll elő az aláhúzott, áthúzott, keskeny, felső index, kiskapitális stb.

A betűk méretének egysége a pont (nyomdai pont), **amely az inch (2,54 cm) 1/72-ed része.** A főszöveg tipikus betűmérete 10–12 pont (kb. 0,3-0,4 cm).

► A Times New Roman betűtípus négy változata: normál, félkövér, dőlt és félkövér dőlt

Érdemes megjegyeznünk, hogy **a félkövér betűstílust a CTRL+B** (B: bold, jelentése feltűnő), **a dőltet pedig a CTRL+I** (I: italic, jelentése a könyvnyomtatásban dőlt) **billentyűkombinációval is beállíthatjuk.**

A betűtípusokat formai szempontból három fő kategóriába sorolhatjuk.

A talpas betűk vonalvastagsága változó, a szárak talpakban végződnek. A talpak vezetik a szemet, ami megkönnyíti az olvasást, ezért könyvek, újságok esetében előszeretettel használják. Ilyenek például: Cambria, Times, Garamond.

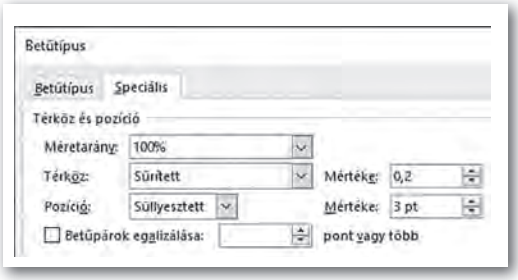
Monitoron vagy projektorral megjelenített szöveg esetén jobban felismerhetők **a talp nélküli betűtípusok**, amelyek **vonalvastagsága többnyire állandó**. Ilyenek például: Arial, Calibri, Helvetica.

A többi betűtípust általában dísz- vagy reklámbetűként használják. Sajnos közöttük sok olyan van, amelyből a magyar ékezetes karakterek egy része hiányzik.

► A mintapélda betűtípusai: Times New Roman (talpas), Arial (talp nélküli) és Algerian (egyéb)

Speciális beállítások

A további betűformázási lehetőségek közül kettőt érdemes kiemelni. Az egyik a *betűköz* (térköz), amelynek kis mértékű csökkentése (növelése) nem rontja az olvashatóságot, ám rövidebbé (hosszabbá) teheti a szöveget. A másik pedig a *pozíció*, a betűk emelése (süllyesztése), mellyel látványos kiemeléseket tehetünk vagy ikonokat simíthatunk a szövegbe.

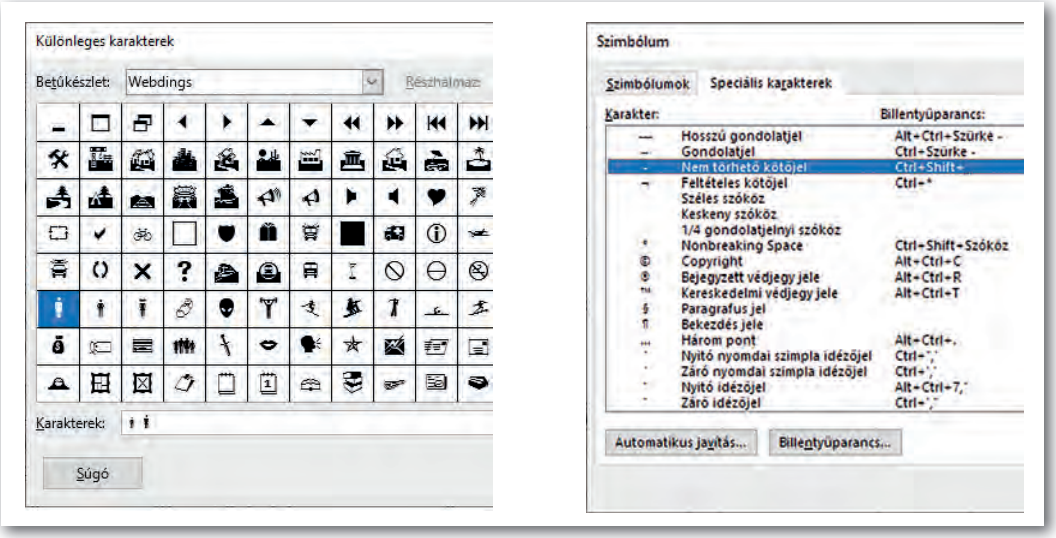


► A betű speciális beállításai (Microsoft Word)

2. példa: **Szimbólumok**

Készítsük el az alábbi tájékoztató táblát!

A billentyűzet nem tartalmazhatja valamennyi karaktert, hiszen „csak” körülbelül 100 billentyű van rajta. Ugyanakkor *minden szövegszerkesztő program lehetővé teszi a karakterkészletek megtekintését és a megfelelő karakterek kiválasztását*, majd beszúrását. Esetünkben (Windows alatt) a *Webdings* és a *Wingdings* karakterkészletben kell keresnünk a megfelelő jeleket, de gyakran használják a *Symbol* karakterkészletet is, amely főleg matematikai és műszaki jeleket tartalmaz. Az így beszúrt karaktereket **szimbólumoknak** is nevezik. A szimbólumok formázása a betűkével teljesen azonos, az eltérés csupán a bevitel módjában van.



► Bal oldalon szimbólumok kiválasztása (LibreOffice Writer), jobb oldalon a szöveg tördeléséhez szükséges speciális karakterek elérhetősége (Microsoft Word)

Szavak és kifejezések elválasztása a sorok végén

Ha egy szó nem fér ki az adott sorban, akkor a szövegszerkesztő program az adott szót megelőző szóköznél vagy – ha ilyen van – az elválasztójelnél vagy a kötőjelnél automatikusan új sort kezd. Gyakran előfordul, hogy az adott szöveg különleges volta miatt ezt nem tehetjük meg, például nem választható szét egy mennyiség a mérőszám és a mértékegység között, de nem választható szét egy telefonszám sem a benne lévő kötőjelnél. Ilyen esetekben úgynevezett *nem törhető szóközt* kell normál szóköz helyett beszúrnunk, illetve *nem törhető kötőjelet* normál kötőjel helyett.

Néha fordítva, azt szeretnénk elérni, hogy a szövegszerkesztő program a sor végén – ha erre szükség van –, az adott helyen válasszon el. Ilyenkor az adott helyre *feltételes kötőjelet* kell beszúrnunk. A feltételes kötőjel a nyomtatásban nem jelenik meg.

Ezeket a különleges karaktereket vagy billentyűzetkombinációval írjuk be (például a nem törhető szóköz: CTRL + SHIFT + szóköz), vagy kiválasztjuk például a *Beszúrás* > *Szimbólum* (illetve a *Beszúrás* > *Különleges karakter*) menüponttal.

További kiegészítések

Vajon mit is jelent az, hogy egy betű 12 pontos? A betűk ténylegesen egy téglalap alakú részt foglalnak el, ennek a magassága 12 pont.

Nem minden betűkészlet tartalmazza a szövegszerkesztő program által felkínált valamennyi betűstílust. Ha például a félkövér változat nem áll rendelkezésre, akkor a szövegszerkesztő program azt matematikai úton állítja elő.

Az aláhúzott betűstílus használata nem szerencsés, mivel a vonal átvágja a lelógó szárakat, és ez rontja az olvashatóságot.

Feladatok

1. Tervezzünk iskolánk részére céges levélpapírt, vagy cégespapírt! Használjunk hozzá többféle betűformátumot, de ne legyen túldíszített! A címet, telefonszámot, webcímet szimbólumokkal (például: ☎, ✉) vezessük be!
2. Írjuk be a mintán látható szöveget a megadott formában!

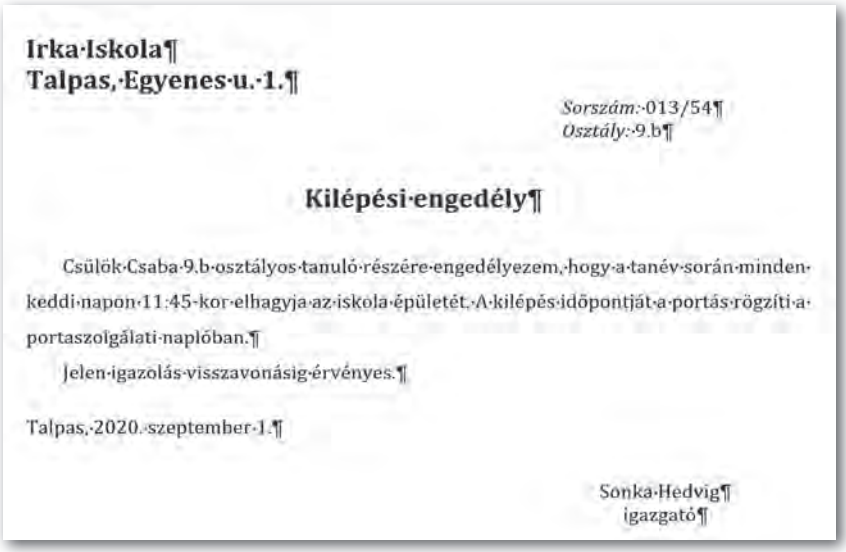
> **TÉTEL:**
> Az **euklideszi geometriában** a háromszög szögeinek összege 180°, vagyis ha a szögek jele rendre α, β, illetve γ, akkor $\alpha + \beta + \gamma = 180°$.

3. Nagyon sok alkalmazásban bizonyos szimbólumokat bevihetünk megfelelő karakterek egymás után történő beírásával, például egy smiley beszúrható a :) sorozattal vagy a → nyíl a --> sorozattal. Milyen további példákat tudunk mondani?
4. Tudunk-e magunk is karaktereket tervezni a szövegszerkesztő program számára? Járjunk utána!

Bekezdésformázás

3. példa: Kilépési engedély

Készítsük el az ábrán látható kilépési engedélyt! A szöveget gépeljük be, vagy töltsük le a tankönyv weblapjáról! A szöveg végig Cambria betűtípussal készült, 16, illetve 12 pontos betűmérettel.



Irka·Iskola¶
Talpas,·Egyenes·u.·1.¶

Sorszám:·013/54¶
Osztály:·9.b¶

Kilépési·engedély¶

Csülök·Csaba·9.b·osztályos·tanuló·részére·engedélyezem,·hogy·a·tanév·során·minden·keddi·napon·11:45-kor·elhagyja·az·iskola·épületét.·A·kilépés·időpontját·a·portás·rögzíti·a·portaszolgálati·naplóban.¶

Jelen·igazolás·visszavonásig·érvényes.¶

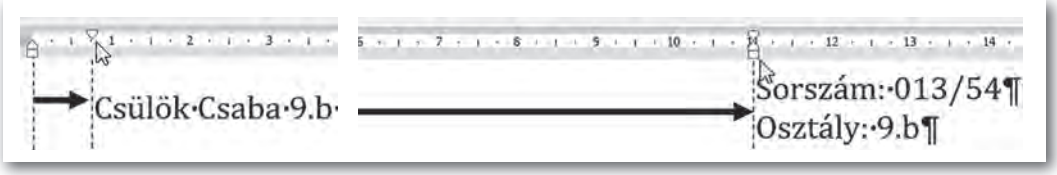
Talpas,·2020.·szeptember·1.¶

Sonka·Hedvig¶
igazgató¶

Nézzük végig, hogyan alakítottuk ki a tagolást! A nem nyomtatható karakterekből látható, hogy ehhez nem használtuk sem a szóköz, sem az ENTER gombot.

Első lépésben állítsuk be a menü ikonjaival a bekezdések **igazítását**! Láthatóan a cím és az aláírás *középre zárt*, míg az engedély szövege *sorkizárt* (vagyis a sorok mindkét vége egymás alatt van). A többi bekezdés *balra zárt.*

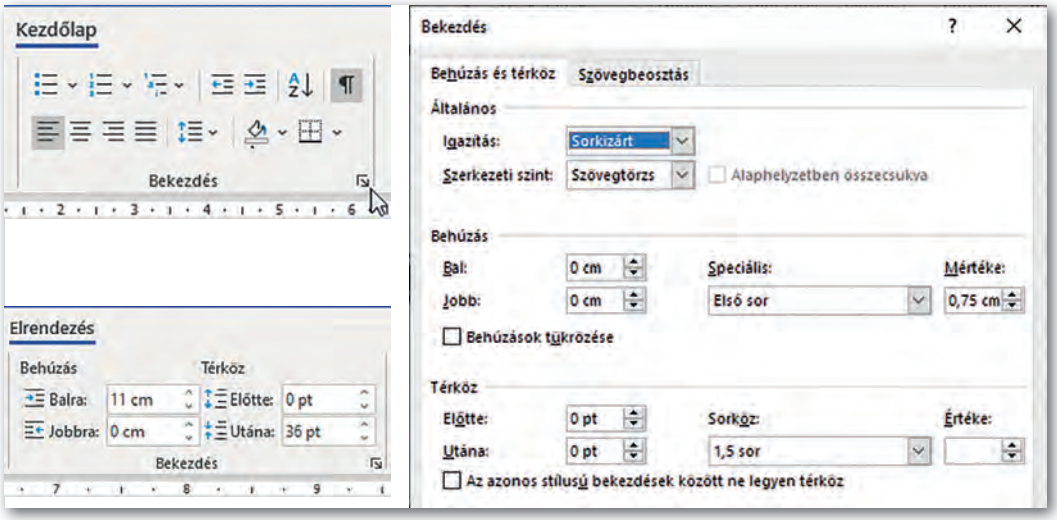
Második lépésben állítsuk be a bekezdések **behúzását** (vízszintes elrendezését)! A cégjelzést követő két adat például ugyan balra zárt, de a bal margótól vett távolságuk *(bal behúzás)* 11 cm. Ugyanígy, az aláírás is középre zárt, de itt is megnöveltük a bal behúzást (10 cm). Végül az engedély szövegében az *első sorokat* húztuk be, ennek mértéke 0,75 cm. A behúzást kényelmesen elvégezhetjük a vonalzón lévő csúszkákkal.



► Az első sor behúzás a bal felső, a bal behúzás a négyzet alakú csúszkával történik (Microsoft Word)

Harmadik lépésként a bekezdések függőleges elrendezését, a **térközt** állítjuk. Ez előkészítést igényel, mivel néhány szövegszerkesztő szokatlan alapértékeket (például 8 pontos térközt, 1,08-os sorközt) állít be. Jelenítsük meg például a *Bekezdés* ablakot, ahol a térköz csoportban a térközöket állítsuk 0-ra, a sorközt pedig szimplára!

A mintában a cím *előtt és mögött 24 pontos*, a dátum előtt 12 pontos, utána 24 pontos térköz van. Az engedély szövegében a *sorköz másfeles.* (A térközöket egyaránt állíthatjuk az előző bekezdés mögé vagy az adott elé. Ha mindkettőt beállítottuk, akkor például a *Microsoft Word* a nagyobb értéket alkalmazza, míg a *LibreOffice Writer* a két értéket összeadja.)



▶ Bekezdésállítási lehetőségek a Microsoft Word menürendszerében. A Bekezdés párbeszédablakot az egér által mutatott nyitójelre kattintva nyithatjuk meg.

Bekezdésformátumok

A bekezdések formázásánál a három legfontosabb lehetőségünk: az igazítás kiválasztása, a behúzás megadása és a térköz beállítása.

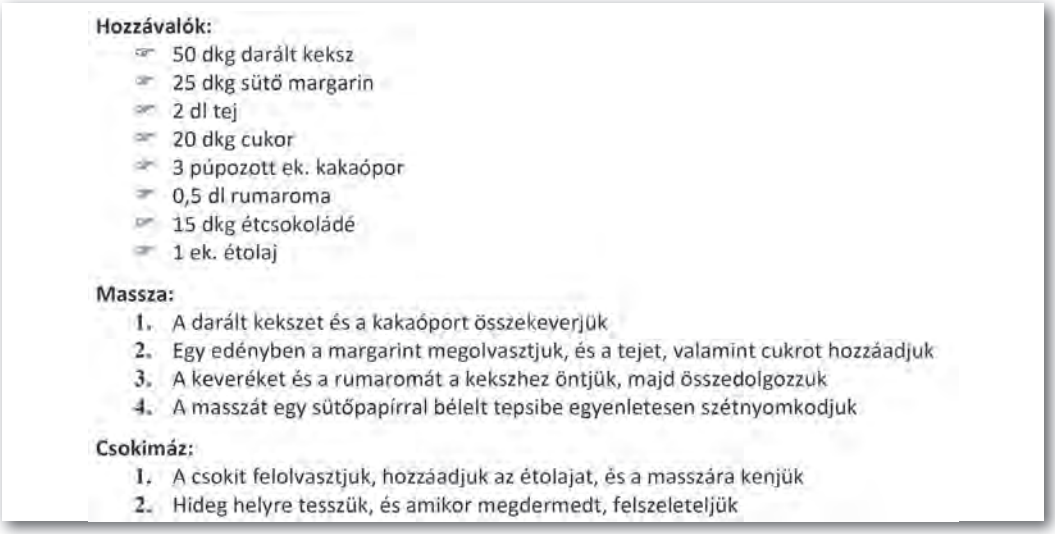
Az igazítás a bekezdés sorainak egymáshoz viszonyított helyzete. A négy leggyakoribb igazítás a **balra zárt, jobbra zárt, középre zárt, illetve sorkizárt.** Sorkizárt esetben a program a szóközök szélességét állítja be úgy, hogy a bekezdés sorainak két vége egymás alatt legyen.

A behúzás a bekezdés két szélének távolsága a margótól. Folyó szövegben gyakori, hogy a bekezdés első sorát beljebb vagy kijjebb kezdik. A bekezdés első soron kívüli, margótól vagy bal behúzástól való távolságát *függő behúzásnak* nevezzük.

A térköz a bekezdés távolsága az előző, illetve következő bekezdéstől. Többnyire a térköz beállításai között adhatjuk meg a sorköz értékét is.

4. példa: Recept

Készítsük el a *sportszelet* nevű sütemény ábrán látható receptjét! A szöveget gépeljük be vagy töltsük le a tankönyv weblapjáról!



Hozzávalók:

- 50 dkg darált keksz
- 25 dkg sütő margarin
- 2 dl tej
- 20 dkg cukor
- 3 púpozott ek. kakaópor
- 0,5 dl rumaroma
- 15 dkg étcsokoládé
- 1 ek. étolaj

Massza:

1. A darált kekszet és a kakaóport összekeverjük
2. Egy edényben a margarint megolvasztjuk, és a tejet, valamint cukrot hozzáadjuk
3. A keveréket és a rumaromát a kekszhez öntjük, majd összedolgozzuk
4. A masszát egy sütőpapírral bélelt tepsibe egyenletesen szétnyomkodjuk

Csokimáz:

1. A csokit felolvasztjuk, hozzáadjuk az étolajat, és a masszára kenjük
2. Hideg helyre tesszük, és amikor megdermedt, felszeleteljük

Listakezelés

Összetartozó bekezdésekből listát hozhatunk létre. A lista bevezető jellel ellátott bekezdésekből épül fel.

Első esetben a hozzávalók egy **felsorolást** alkotnak, *minden bekezdést ugyanaz a listajel* vezet be. A lista kiválasztása után az egységes jelet a *Felsorolás* ikonra kattintva állíthatjuk be. A listajelet akár módosíthatjuk is a megfelelő szimbólum vagy kép kiválasztásával. Érdemes tudnunk, hogy bizonyos szövegszerkesztő programok (például a *Microsoft Word* is) a lista elemei közül automatikusan eltávolítja a térközt. Ezt a beállítást például a *Bekezdés* ablakban bírálhatjuk felül.

Második esetben egy **számozást** látunk, ahol *a listaelemeket számok, betűk, római számok stb. vezetik be*. Ennek beállítása a felsoroláshoz hasonlóan, például a menü *Számozás* nevű ikonjával történik. Amikor a második (a „Csokimáz” alatti) számozást beállítjuk, az alapértelmezetten az előző folytatása lesz. A helyi menü megfelelő parancsaival a számozást újraindíthatjuk, sőt akár kezdősorszámot is beállíthatunk.

Szükség esetén **többszintű listát** is kialakíthatunk. Ennek módja a programtól függ, *Microsoft Wordben* például használhatjuk a *Többszintű lista* ikont. Sok esetben azonban legegyszerűbb a megfelelő listaelemeket a *Behúzás növelése* és a *Behúzás csökkentése* ikonnal léptetni.

A szövegszerkesztő programok támogatják **a lista elemeinek rendezését** is. Így például elegendő beírni osztálytársaink nevét (mindegyiket önálló bekezdésbe), majd például a *Rendezés* paranccsal névsorba állítani. Rendezéskor megadhatjuk a rendezés irányát (emelkedő, csökkenő), de ezen kívül a rendezendő adatok típusát is (szám, szöveg, dátum).

5. példa: Figyelmeztető tábla

Készítsük el az alábbi figyelemfelhívó táblát az iskolai klubhelyiség ajtajára!

Írjuk be a szöveget, és formázzuk meg a betűket a minta alapján! Jelöljük ki mind a két bekezdést, zárjuk középre, állítsuk be a betű és a háttér színét, majd lépjünk be a szegélyek beállítását tartalmazó ablakba! Az ablakot például a *Kezdőlap* > *Bekezdés* csoportjából (illetve a *Formátum* > *Bekezdés* menüponttal) érhetjük el.

Szegély *beállításakor megadhatjuk a vonal vastagságát, színét és stílusát* (szimpla, dupla, szaggatott stb.) akár oldalanként eltérő módon is. Utolsó lépésként a vonalzón lévő csúszkákkal állítsuk be a bal és jobb behúzást!

Feladatok

1. Írjunk kérvényt az iskola igazgatójának, amelyben kérvényezzük, hogy minden nap csak a második órára kelljen bejönnünk!
2. Tervezzünk hirdetőtáblákra kitehető apróhirdetést, amelyben hétvégén takarítást vállalunk!
3. Hozzuk létre a bal oldali ábrán látható vázlatot!

1. Földrajz
   a) Domborzat
   b) Vízrajz
   c) Éghajlat
   d) Élővilág, természetvédelem
      I. Nemzeti parkok
      II. Világörökség
2. Gazdaság
   a) Általános adatok
   b) Gazdasági ágazatok
      I. Mezőgazdaság
      II. Ipar
      III. Külkereskedelem
      IV. Egyéb ágazatok
3. Közlekedés
   a) Közúti
   b) Vasúti
   c) Vízi
   d) Légi

4. Készítsük el az Irka Iskola jobb oldali ábrán látható újsághirdetését!

Az adatok áttekinthető elrendezése

6. példa: Órarend készítése

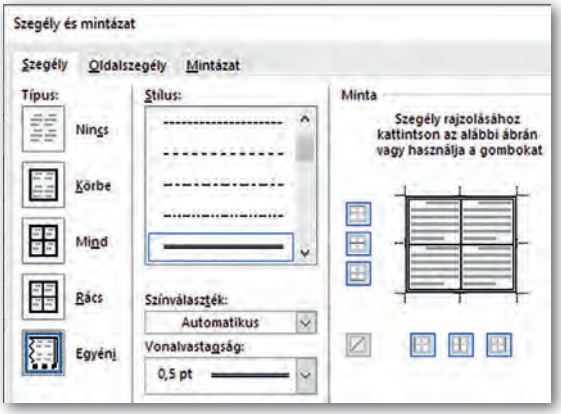
Készítsük el az ábrán látható egyszerű órarend nyomtatványt!

	Hétfő	Kedd	Szerda	Csütörtök	Péntek
1.					
2.					
3.					
4.					
5.					
6.					
7.					

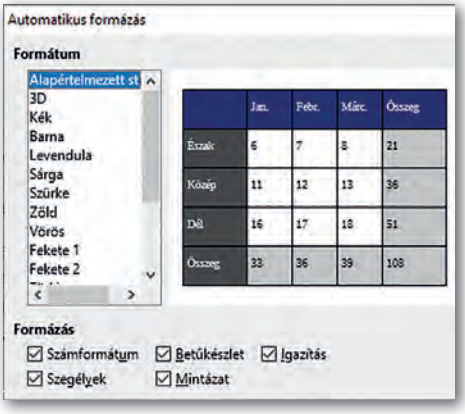
Szúrjunk be egy 8 × 6-os táblázatot, majd írjuk be a napok nevét és a tanórák sorszámát a megfelelő cellákba, majd formázzuk meg!

Táblázatot például a *Beszúrás* > *Táblázat* (illetve a *Táblázat* > *Táblázat beszúrása*) menüponttal illeszthetünk be. A táblázat formázását pedig például a *Táblázattervező* és az *Elrendezés* menük parancsaival (illetve a *Táblázat* menü pontjaival) érhetjük el. Cellatartományt az egér húzásával jelölhetünk ki. Egész sorokat a sor elé, egész oszlopokat az oszlop fölé kattintva is kiválaszthatunk, ilyenkor az egér húzásával akár többet is.

Következő lépés a táblázat oszlopszélességeinek megadása (például legyen az első oszlop szélessége 1 cm, a többié 2 cm), amit a szegély kialakítása követ. Ezt végezhetjük például a *Szegélyek* ablak segítségével, de a szövegszerkesztő programok általában lehetőséget kínálnak arra is, hogy a szegélyeket a megfelelő vonalstílus kiválasztása után egérrel húzzuk meg.



▶ Szegélyek beállításának lehetőségei a Szegély és mintázat ablakban (Microsoft Word)



▶ Táblázat automatikus formázása (LibreOffice Writer)

Táblázatok formázása

A mai szövegszerkesztő programokban, ha megadjuk *a sorok és oszlopok számát,* a program automatikusan létrehoz egy táblázatot valamilyen alapbeállításokkal. Később a táblázatba további sorokat vagy oszlopokat szúrhatunk be, de akár törölhetünk is. A táblázatot nemcsak megformázhatjuk, hanem a képekhez hasonlóan akár a *szöveggel körbe is futtathatjuk*, illetve feliratot is rendelhetünk hozzá.

A **táblázat önálló objektum,** szerepe, hogy az adatokat áttekinthetően rendezze el. Ezért lehetőségünk van a *sorok és oszlopok szélességének, szegélyének, színének* megadására. A cellák tartalmát *igazíthatjuk vízszintesen és függőlegesen,* de beállíthatunk belső margót is. A cellák tartalmára pedig alkalmazhatjuk a megismert bekezdésformátumokat.

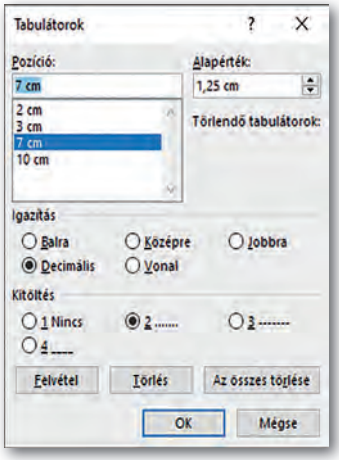
Összetett szerkezetű adatok esetén a táblázat *celláit feloszthatjuk* vagy *egyesíthetjük*, illetve a cellákba további objektumot, például képet vagy táblázatot is beszúrhatunk.

A szövegszerkesztő programok nagyon sok automatikus lehetőséget biztosítanak, például lehetővé teszik az oszlopok szélességének szöveghez vagy oldalhoz illeszkedő automatikus méretezését. A táblázat megformázásához pedig előre elkészített formátumok használatát is felkínálják, amelyet további finomításokkal testre szabhatunk.

Tabulátorok használata

A tabulátorokat az írógépen találták ki az adatok elrendezésére. Az írógép felső sorában lévő szélső gomb lenyomásával megjelölhetjük az aktuális oszloppozíciót. Később, ha a felső sorban lenyomjuk a megfelelő gombot, a kocsi automatikusan a megadott pozícióra ugrik, vagy rendre egy, két, három... betűhellyel előbb áll meg, hogy a számokat helyi érték szerint egymás alá lehessen írni.

▸ A tabulátor beállítása után a kocsival a megadott helyre léphetünk az írógépen.



▸ Tabulátorpozíciók beállítása (Microsoft Word)

A szövegszerkesztő programokban a billentyűzeten lévő **tabulátor** gombot nyomkodva a kurzor *adott pozíciókon lépked végig*, amelyek egyenlő távolságra vannak egymástól (tipikusan 1,25 cm-re).

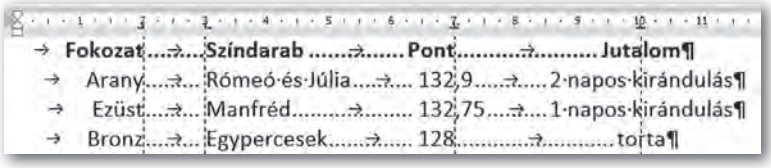
Az alapértelmezett *tabulátorpozíciókat módosíthatjuk.* Előírhatjuk továbbá, hogy az adott pozícióhoz a program a szöveg bal szélét, jobb szélét, közepét, továbbá számok esetén a tizedesvesszőt helyezze-e el. (Így **balra zárt, jobbra zárt, középre zárt,** illetve **decimális tabulátorpozícióról** beszélünk.)

A tabulátorpozíciókba helyezett szöveget előre megadott stílusú, pl. szaggatott, pontozott vagy folytonos *vonallal összeköthetjük*.

A cellák tartalmában a TAB, valamint a SHIFT + TAB segítségével lehet előre, illetve visszafelé haladni. Tabulátorokat táblázatba is elhelyezhetünk, ebben az esetben a tabulátor gombbal együtt a CTRL gombot is nyomnunk kell.

7. példa: Eredménylista

Készítsük el tabulátorpozíciók segítségével az iskolai színjátszó fesztivál eredményét tartalmazó alábbi összefoglalót!



Írjuk be egy-egy tabulátorpozícióval elválasztva az első sor szavait! Ezután anélkül, hogy új bekezdést kezdtünk volna, jelöljük be a vonalzón az ábrán látható tabulátorpozíciókat! A megfelelő jelet a vonalzó bal szélén választhatjuk ki. A hibás tabulátorpozíciót úgy töröljük, hogy az egérrel megfogjuk, és lehúzzuk a vonalzóról.

A tabulátorpozíció bekezdésformátum, csak az adott bekezdésre vonatkozik. A sor végén lenyomva az ENTER gombot azonban az új bekezdés felveszi az előző formátumát, és folytathatjuk a lista beírását.

A *Tabulátorok* ablakba a vonalzón valamelyik tabulátorpozícióra kettőt kattintva (illetve a *Bekezdés* > *Tabulátorok* fület választva) léphetünk be, itt számszerűen is megadhatjuk vagy módosíthatjuk a pozíciókat. Ugyanitt adhatjuk meg a *Kitöltést,* vagyis azt, hogy a tabulátorokkal tagolt tartalmat az előzővel milyen vonal kösse össze.

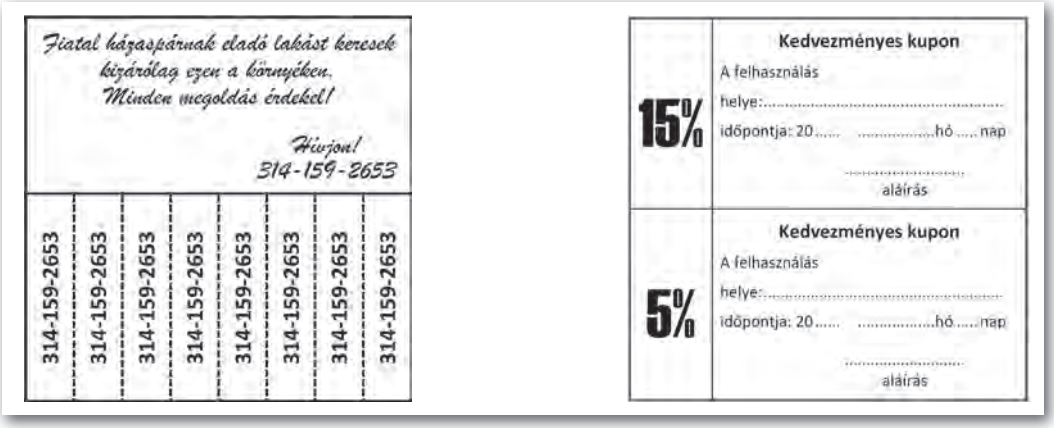
Feladatok

1. Az alábbi ábra egy feladatlap fejlécén szerepelt. Készítsük el tabulátorok és táblázatok alkalmazásával!

2. Az alábbi táblázat azt tartalmazza, hogy mennyi szénhidrát van 100 gramm gyümölcsben, illetve zöldségben. Készítsük el tabulátorok felhasználásával!

GYÜMÖLCS		ZÖLDSÉG	
Alma	7 g	Uborka	1 g
Meggy	11 g	Káposzta	6 g
Szőlő	18 g	Sárgarépa	8 g
Banán	23 g	Zöldborsó	14 g

3. Készítsük el a bal oldali ábrán látható, kiragasztható hirdetést!



4. Készítsük el a jobb oldali ábrán látható kedvezményes kuponokat!
5. Készítsük el tabulátorok segítségével az alábbi iskolalátogatási igazolást!

Iskolalátogatási igazolás

Hivatalosan igazolom, hogy.....

(született ....., □□□□ év ..... hó .....napján, anyja neve: ..... ,)

a(z) .....(iskola) ...../..... -es tanévre beírt tanulója.

Ezt az igazolást ..... céljából állítottam ki.

Kelt:..... , 20..... év.....hó .....nap

.....
igazgató

Képek, ábrák

8. példa: Az idő pénz!

Készítsük el az alábbi mintaoldalt Benjamin Franklin (1706–1790) amerikai tudós, politikus híres mondásáról! Az idézet a szakiskolai közismereti tankönyv (FI-511010902) 18. oldaláról származik. A szövegrészt és a képet töltsük le a tankönyv weblapjáról!

Az idő pénz

„Az idő pénz” – mondta a képen látható **Benjamin Franklin**. Sőt, még azt is hozzátette, hogy: „Szereted az életet? Akkor ne vesztegesd az időt, hisz belőle áll az élet.”

Tizenkét évesen szegődött tanoncként – apja engedélyével – a nála kilenc évvel idősebb bátyjához, Jameshez, aki saját nyomdát működtetett. Az itt eltöltött évek alatt alaposan kitanulta a nyomdász szakmát, ami későbbi vállalkozásainak alapja lett. De a tanoncság azt is jelentette, hogy alkalma nyílt rengeteget olvasni, valamint könyvgyűjteményét is gyarapította megtakarított pénzéből. Írásai ekkor jelenhettek meg először a nyilvánosság előtt, igaz, csak álnév alatt.

Benjamin Franklin (1706–1790) amerikai tudós, politikus

Tizenhét évesen, egyetlen holland dollárral a zsebében utazott a pennsylvaniai Philadelphiába. 1728-ban társtulajdonos lett egy nyomdában, és megjelentette a Pennsylvania Gazette és a Poor Richard’s Almanach című folyóiratokat. Szorgalmas, igényes munkájának és körültekintő gazdálkodásának hála, fokozatosan fölé került a konkurenciájának, vállalkozása egyre nagyobb haszonnal működött. Riválisaival ellentétben nem hagyta, hogy adósságai elnyeljék, mihamarabb rendezte őket. Erre ösztönözte az az erős meggyőződése, miszerint az adósság valójában egyfajta rabszolgaság. Később a postához szegődött, ahol a postaügyi miniszterségig vitte, miközben megreformálta a postaforgalmat.

Formázzuk meg a szöveget a mintának megfelelően, majd szúrjuk be a képet például a *Beszúrás* menü *Kép* menüpontjával!

Első lépésként csökkentsük a kép méretét! Ezt megtehetjük a kép sarkainak húzásával is, de célszerűbb a méretet pontosan beállítani például a *Képformátum* > *Méret* csoportjában (illetve a *helyi menü* > *Tulajdonságok* pontjával). Ügyeljünk arra, hogy az átméretezés az oldalarányok megtartásával történjen!

A következő lépés a kép körbefuttatása a szöveggel. A kép elhelyezésekor ügyelnünk kell arra, hogy a szöveg könnyen olvasható maradjon, a képet a szöveg bal vagy jobb széléhez kell igazítanunk. A két lépést megtehetjük például a *Képformátum* > *Szöveg körbefuttatása* (illetve a *helyi menü* > *Körbefuttatás*), majd *Igazítás* menüpontjával.

Végül illesszük a kép alá az ábrán látható képaláírást például a helyi menü *Felirat beszúrása* pontjával! A szövegszerkesztő programok ilyenkor automatikusan gondoskodnak a képek számozásáról, ha ezt a lehetőséget nem szeretnénk kihasználni, töröljük a kész képaláírásból. A kép és a felirat a szövegszerkesztő programokban általában két, nem ösz-

szetartozó objektum, így, ha a képet áthelyezzük, az ábraszöveg lemarad. Egy objektumba szervezhetjük azonban azokat, ha mindkettőt kijelöljük és csoportosítjuk, például a helyi menü *Csoportosítás* lehetőségével.

Képek beillesztése

A beszúrt kép *átméretezése* a kép négy sarkában és az oldalfelező pontokban megjelenő *nyolc méretező pont* húzásával történik, de a megfelelő menüpontokkal pontos értéket is megadhatunk. Gyakran lehetőségünk van a kép *elforgatására* is.

A képet elhelyezhetjük *a szöveg elé* vagy *mögé, körbefuttathatjuk* a szöveggel, vagy *beszúrhatjuk karakterként* is.

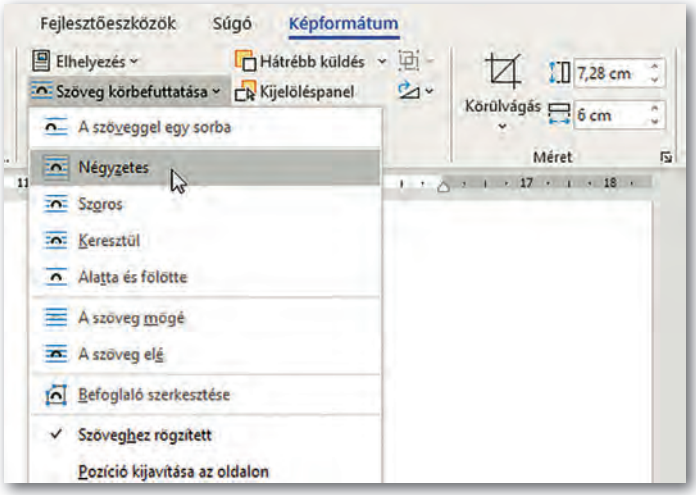
Körbefuttatás esetén ügyelnünk kell arra, hogy *a kép a szöveg szélére kerüljön,* mert a kép előtt kezdődő és utána folytatódó sorokat nehéz szemmel követni. A kép *igazítását* legegyszerűbben és legpontosabban a menü megfelelő lehetőségével végezhetjük.

A *karakterként beszúrt képet* karakterként formázhatjuk, például emelhetjük vagy süllyeszthetjük az alapvonalhoz képest.

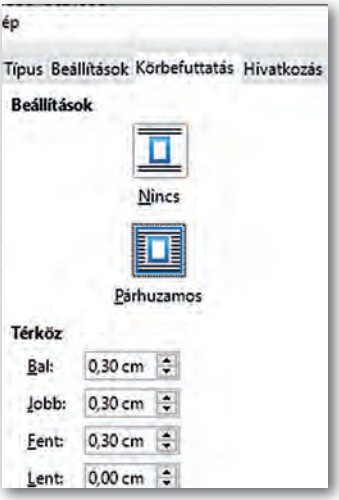
A legtöbb szövegszerkesztő program lehetővé teszi a kép szélének *levágását*, *színeinek módosítását*, *szegélyezését,* esetleg még a *háttér eltávolítását* is.

A szöveg mögé helyezett képek rontják a szöveg olvashatóságát, ezért érdemes *halványítani*, míg a szöveg elé helyezett képek esetén az *átlátszóságot* beállítani.

Nem szép, ha egy körbefuttatott kép és a szöveg összeérnek. A *kép és szöveg távolságát* ezért érdemes szükség esetén a megfelelő oldalakon megnövelni, például *helyi menü* > *Méret és pozíció* (illetve *Tulajdonságok*) menüpontjával megjeleníthető ablakban *A szöveg körbefuttatása* (illetve *Körbefuttatás*) fülön.



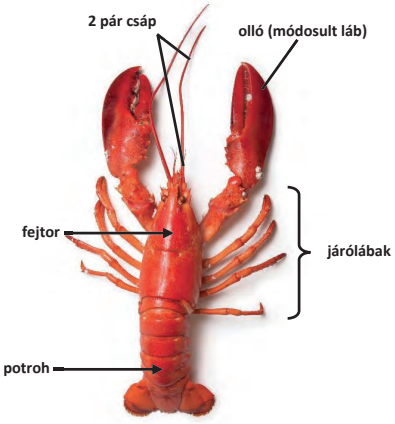
▶ A körbefuttatás lehetőségei a Képformátum menüben (Microsoft Word)



▶ Kép és szöveg távolsága (LibreOffice Writer)

9. példa: Értelmező feliratok készítése

A rákok osztálya



A rákok döntő többsége vízi állat, a tengerek és az édesvizek lakója. Sokféle testfelépítésű rák van, ebben a fejezetben csak a legfejlettebb 10 lábú rákokkal foglalkozunk. A testük fejtorra és potrohra tagolódik, járólábaik a fejtorról erednek. Kitinpáncéljukat mész szilárdítja. A vizek aljzatán lépegetnek, de ha megijesztik őket, a potrohukat hirtelen a fejtor alá csapják, és nagy sebességgel elmenekülnek. Kopoltyúval lélegeznek, amely a fejtor páncélja alatt, védett helyen található. A rákok általában ragadozók vagy dögevők. Zsákmányukat az első pár lábon lévő ollóval ragadják meg, és rágó szájszervükkel darabolják fel. Fejlett összetett szemükkel elég jól látnak. A csápok szagló- és tapintószervek is. A rákok petéből, lárva alak nélkül fejlődnek. A rákok osztályának hazai képviselője a folyami rák. A tengerpartokon gyakoriak a tarisznyarákok.

► Értelmező feliratok kialakítása a beépített alakzatok segítségével

Készítsük el a fenti mintán látható szövegrészt!

Töltsük le a tankönyv weblapjáról a rákokról szóló dokumentumot és a *Rák.png* nevű képet! A szöveg és a kép a hetedikes biológia-tankönyv (FI-505030701/1) 64. oldaláról ismerős lehet.

Szúrjuk be a képet, futtassuk körbe a szöveggel négyzetesen, majd küldjük a szöveg mögé! Itt fogjuk elkészíteni az ábrát, amelyet aztán csoportosítva, egy objektumként szúrunk be a szövegbe. A szükséges objektumokat például a *Beszúrás* > *Alakzatok* menüpontjával érhetjük el. Az alakzatok formátumát minden programban többféleképpen, például a *helyi menüből* elérhető menüpontokkal állíthatjuk be.

Illesszünk be egy *vonal* alakzatot, majd helyezzük el és formázzuk meg úgy, hogy a végén nyílhegy legyen! Növeljük meg a vastagságát 2-3 pontosra, és állítsuk be színét például feketére, ezután húzzuk a fejtor mellé! A *fejtor* feliratot egy *téglalapba* érdemes helyezni, amelynek háttér- és szegélyszínét állítsuk átlátszóra, azaz válasszuk azt, hogy *nincs szín*, a benne lévő betűk színét pedig feketére. Készítsünk másolatot a két beszúrt alakzatról, és a másolatokat rendre húzzuk a megfelelő helyre! Hasonló módon szúrhatjuk be a többi alakzatot és a *kapcsos zárójelet* is.

A szövegszerkesztés során szükségünk lehet egyszerű **alakzatok** (vonalak, nyilak feliratok) beszúrására a szöveg vagy a beszúrt képek értelmezéséhez. Ehhez a programok kész alakzatokat kínálnak nagyon sok formázási lehetőséggel.

Összetettebb ábrák esetén érdemes inkább egy vektorgrafikus programot használni.

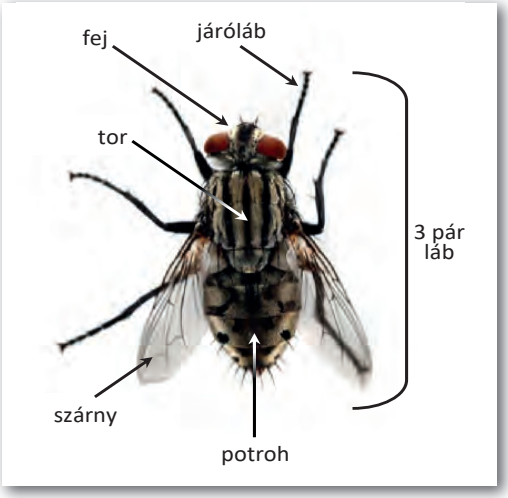
10. példa: Egyenletszerkesztő
Készítsük el az alábbi szövegrészt!

Az a és b nemnegatív számok **négyzetes közepe**: $\sqrt{\frac{a^2+b^2}{2}}$

A képlet elkészítéséhez *Microsoft Word* használata esetén válasszuk a *Beszúrás* > *Egyenlet* menüpontot, ahol az alábbi ábrán látható módon, lépésről lépésre építhetjük fel a képletet: *Gyökkifejezések* beszúrása → *Törtek* beszúrása → *Indexek* beszúrása.

Feladatok

1. Készítsünk egy kétoldalas tanulmányt a golyóstollról! A szöveget egységesre formázzuk meg, a képeket pedig azonos szélességben, négyzetesen körbefuttatva, ábraszöveggel ellátva tegyük be. Ne feledkezzünk meg a forrás megadásáról!
2. A tankönyv weblapjáról töltsük le az ízeltlábúakról szóló részt (ugyancsak a hetedikes biológiakönyvből származik), majd készítsük el és szúrjuk be a mellékelt ábrát, amely az ízeltlábúak testfelépítését szemlélteti!
3. Írjuk be egyenletszerkesztő felhasználásával a harmonikus, mértani, számtani és négyzetes középre fennálló alábbi matematikai összefüggést!



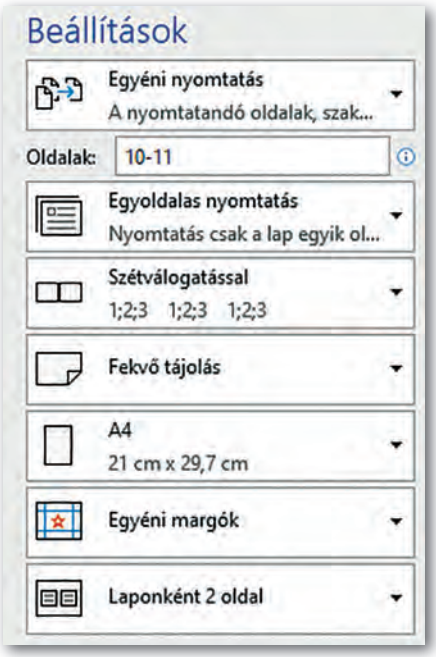
$$\frac{1}{\frac{1}{a}+\frac{1}{b}} \leq \sqrt{ab} \leq \frac{a+b}{2} \leq \sqrt{\frac{a^2+b^2}{2}}$$

Fájlok kezelése, megosztása

Oldalbeállítás és nyomtatás

Mivel a szövegszerkesztővel készült dokumentumot alapvetően papíralapú megjelenítéshez tervezzük, ezért beállíthatjuk a *papír méretét* és *tájolását* (álló vagy fekvő), illetve a *margók* nagyságát.

Látványos elemeket helyezhetünk az oldalak hátterébe is, ilyen a *vízjel* (például egy halvány MINTA felirat), az *oldalszegély* vagy a *háttérszín*.

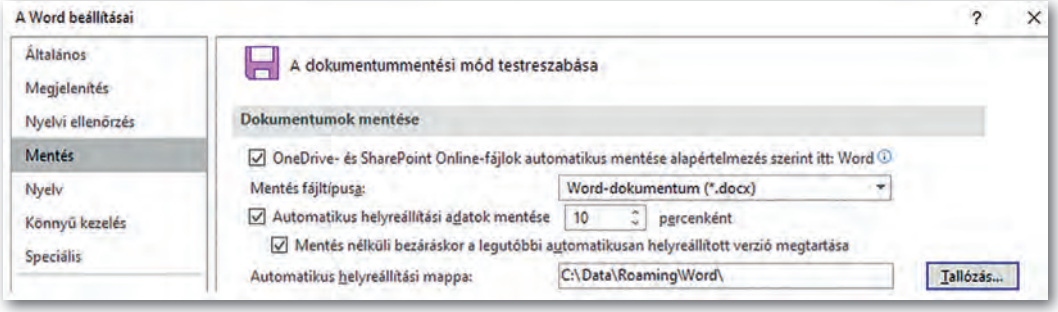


Ezeket a funkciókat például az *Elrendezés* > *Oldalbeállítás (Formátum* > *Oldal)* pontjával érjük el.

Nyomtatáskor papírt spórolhatunk azzal, ha csak a szükséges oldalakat nyomtatjuk ki, illetve ha egy lapra a dokumentum több oldala kerül. Az ábrán látható beállítások esetén a dokumentumnak csak két oldalát nyomtatjuk ki, egy lapra, egymás mellé.

Automatikus mentés

Az automatikus mentés azt jelenti, hogy a szövegszerkesztő program *bizonyos időnként automatikusan menti* a dokumentumot a háttérben, így áramkimaradás vagy egyéb hiba esetén csak az utolsó néhány perc munkája vész kárba.



▶ Az automatikus mentés beállításai a Fájl menü Beállítások pontjában (Microsoft Word)

A PDF fájlformátum

Fájlok mentésekor (vagy nyomtatásakor) gyakran használt lehetőség a PDF fájlformátum. **A PDF** *a dokumentumokat eszközfüggetlenül és felbontásfüggetlenül tárolja*. Sokan használják arra, hogy a dokumentum nyomtatott képét így küldjék el elektronikusan, amivel papírt lehet spórolni.

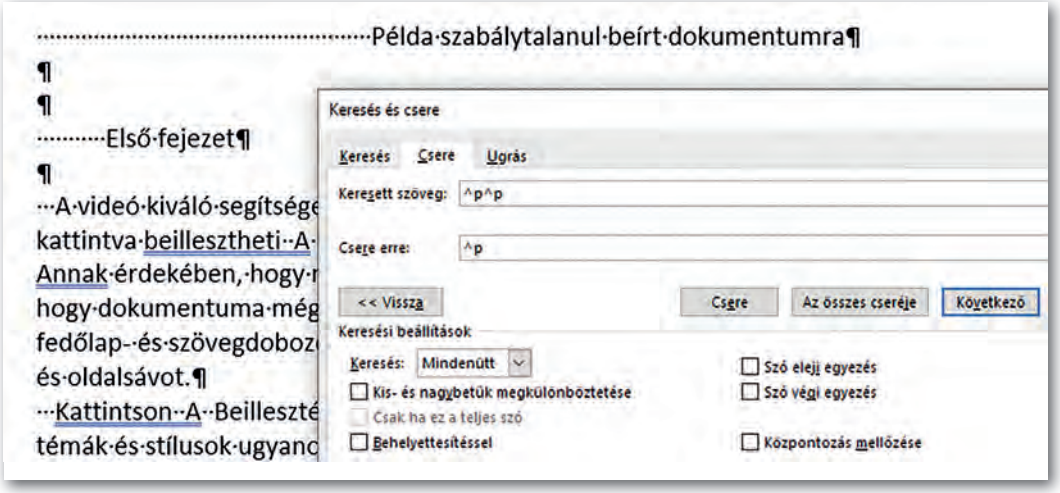
A PDF formátumban mentett fájlt a szövegszerkesztő programok általában meg tudják nyitni, és – bár a megjelenő kép az eredetitől eltérhet – többnyire szerkeszteni is tudják. Egy PDF formátum tartalmazhat interaktív elemeket, például beviteli mezőket, amelyek űrlapok esetén lehetővé teszik adatok bevitelét. A formátum támogatja továbbá a véleményezés lehetőségét is, például kiemelhetünk egyes szövegrészeket vagy megjegyzést fűzhetünk hozzájuk.

11. példa: Ismétlődő szóközök, bekezdésjelek és a formátum eltávolítása

Sajnos gyakran kapunk olyan dokumentumot, amelyről megnyitás után azonnal látjuk, hogy a munka érdemi megkezdéséhez hosszabb előkészítés szükséges. Ilyen például, ha a szöveget szabálytalanul gépelték be, vagy olyan formátumot alakítottak ki, amely nem felel meg az ízlésünknek.

Töltsük le a tankönyv weboldaláról a *Szabálytalan* nevű fájlt! A dokumentum készítője a szöveget – teljesen szabálytalanul – a szóköz és az ENTER ismételt lenyomásával tagolta. Sajnos túl hosszú ahhoz, hogy a hibákat kézzel javítsuk, viszont élhetünk a szövegszerkesztő programok által támogatott *Csere* funkcióval.

Első lépésként cseréjünk le minden dupla szóközt egyre! Mivel így egyesével csökkentjük a szóközök számát, a cserét újra és újra le kell futtatnunk, amíg a szoftver azt nem jelzi, nem történt csere. Hasonló módon tüntethetjük el az ismétlődő bekezdésjeleket, továbbá a bekezdés + szóköz → bekezdés, illetve a szóköz + bekezdés → bekezdésjeleket is. (Ezeket is ismételten, amíg van mit tenni.) Magát a bekezdésjelet speciális karakterként kell beszúrnunk.



► Két bekezdésjel egyre cserélésével a fölösleges bekezdésjelek eltávolítása (Microsoft Word)

Egy másik gyakori probléma, hogy a szöveget megformázva kaptuk, ám mi szeretnénk másképp megformázni. Ilyenkor az a legszerencsésebb, ha a formátumot eltávolítjuk, majd újra formázzuk a dokumentumot. A formázást például a *Kezdőlap* > *Összes formázás eltávolítása* ikonnal (illetve a *Formátum* > *Közvetlen formázás törlése* menüponttal) tehetjük meg.

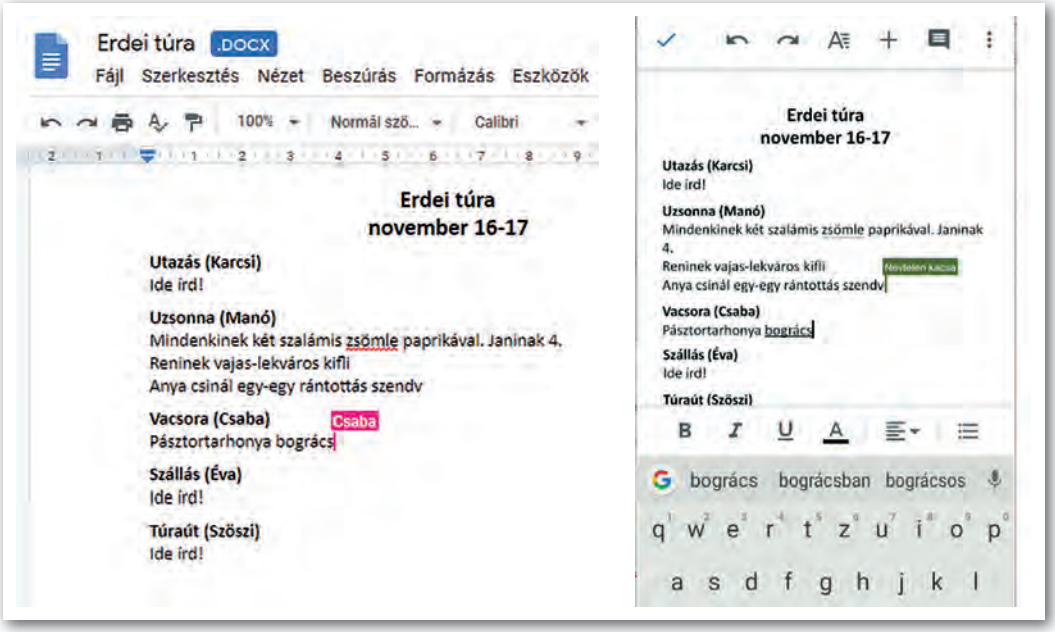
12. példa: Dokumentum közös szerkesztése

Következő példánkban egy közös hétvégi kirándulást szervezünk. A feladatokat szétosztottuk (szállásfoglalás, utazás megszervezése, túraút kijelölése stb.), és szeretnénk létrehozni egy közös dokumentumot, amelybe mindenki beírja, hogy mit végzett.

Készítsük el a nyers dokumentumot, amely tartalmazza a feladatok szétosztását, majd mentsük el egy felhőalapú tárhelyre, és osszuk meg az érintettek között!

Az így megosztott dokumentumot többféleképpen is megnyithatjuk: a szövegszerkesztő programmal közvetlenül a tárhelyről, böngészőn keresztül egy online szövegszerkesztővel, vagy mobiltelefonról a megfelelő alkalmazás telepítésével.

A megosztott dokumentumot akár egyszerre többen is szerkeszthetik. Eközben természetesen tiszteletben kell tartanunk a többiek munkáját.



▸ A dokumentumot ketten is szerkesztik, „Névtelen Kacsa” egy böngészőn keresztül, „Csaba” pedig egy mobilapplikációval

Feladatok

1. Életünk során gyakran kerülünk olyan helyzetbe, amikor önéletrajzot kell készítenünk. Ma több helyen a könnyen áttekinthető, úgynevezett *Europass* önéletrajzot várják el, amelyet weben keresztül is szerkeszthetünk, majd különböző formátumokban letölthetünk. Készítsünk *Europass* önéletrajzot Mátyás király részére!
2. Tervezzünk emléklapot Julius Caesarnak a Rubicon átlépése alkalmából!

3. Készítsük el az alábbi tájékoztatót *a minta alapján!* A szöveget írjuk be, vagy töltsük le a könyv weblapjáról! A sátrat tartalmazó képet egy másik képpel helyettesíthetjük.

Irka Iskola
61023 Talpas
Egyenes tér 1.
☎ 314-159-2653

logó helye

TÁJÉKOZTATÓ AZ *i*² NYÁRI NOMÁDTÁBORRÓL

Az *iskola vezetősége* a **diákönkormányzat** segítségével ebben az évben immár negyedik alkalommal szervezi meg az *i*² Nyári Nomádtábort.

> A Nomádtábort az erdő egy eldugott, háborítatlan részében alakítjuk ki. Itt a tanulók megtapasztalják a természet közelségét, rájönnek, hogy technika nélkül is lehet együtt élni, megtanulják a felelősségvállalást egymásért, a közös munkáért, a természet megóvásáért.

A táborlakók hazaérkezésükkor már biztosan tudnak

- térkép alapján **tájékozódni**,
- *sátrat állítani*,
- **tüzet rakni** a vonatkozó szabályok betartásával,
- egyszerű ételeket *bográcsban* elkészíteni,
- a természet értékeire vigyázni,
- telefon nélkül társalogni a többiekkel.

✂ .....

JELENTKEZÉSI LAP AZ *i*² NYÁRI NOMÁDTÁBORBA

Tanuló	neve:		osztálya:	
	ímél címe:			
Gondviselő	neve:			
	telefonszáma:			
	ímél címe:			
	Megjegyzések:			

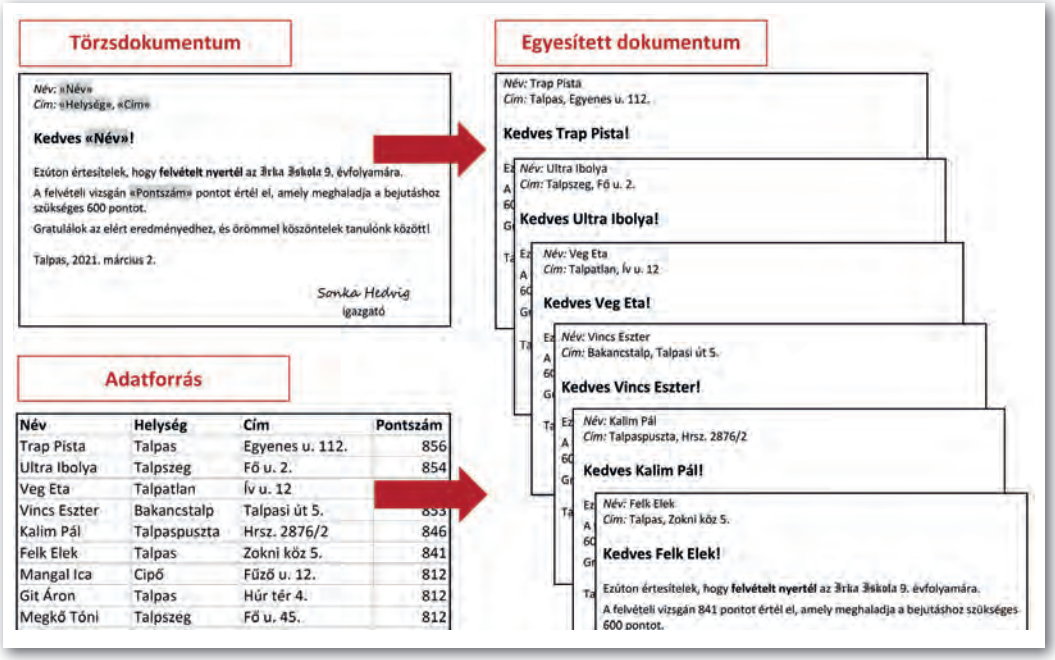
_____(város), 20_____(dátum)

.....
gondviselő aláírása

Körlevél készítése

Gyakran kapunk olyan hivatalos levelet vagy reklámlevelet, amelyet nagyon sok embernek küldtek el egyedi adatokkal, de egyébként azonos tartalommal. Ilyen például az ábrán látható felvételi tájékoztató is. Az ilyen dokumentumokat **körlevélnek** nevezzük.

Régen a körlevél úgy készült, hogy a személyes adatok üresen hagyásával létrehozták, majd sokszorosították a formalevelet, az egyedi adatokat pedig utólag egyenként beleírták. Ma ezt a folyamatot a szövegszerkesztő programok automatikusan elvégzik, eközben minden levélbe „belesimítják” az egyedi adatokat is.



▸ Példa körlevél előállítására

A **körlevél** készítése három lépésből áll. Első lépésben elő kell állítanunk az egyedi adatokat tartalmazó **adatforrást**. A második lépés a **törzsdokumentum** elkészítése, ez lesz valamennyi levél közös formanyomtatványa. Végül a harmadik lépés **a törzsdokumentum és az adatforrás egyesítése,** ekkor készülnek el az egyedi adatokat tartalmazó levelek.

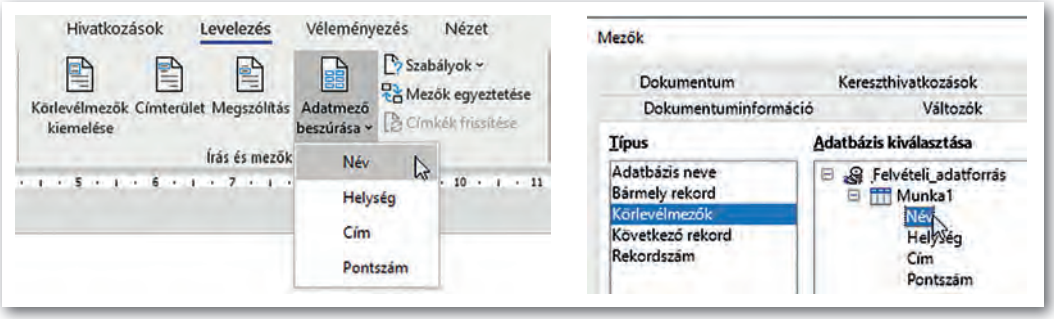
13. példa: Felvételi értesítés

Példánkban az ábrán látható felvételi értesítőt készítjük el. Az adatforrást ezúttal készen kapjuk (ahogy a valóságban is már a felvételi eljárás közben létrejön), tehát elegendő letölteni a tankönyv weblapjáról.

A letölthető adatforrás táblázatkezelővel készült, de természetesen készülhetett volna szövegszerkesztő vagy adatbázis-kezelő programmal is. *Az adatforrás oszlopait adatmezőknek hívják*, ezek tartalmazzák a beszúrandó adatokat, míg *az első sorban az adatmezők neve szerepel*.

A törzsdokumentum készítését egy új dokumentum létrehozásával indítjuk, és kiválasztjuk a használt adatforrást. A továbbiakban a szokásos módon készítjük el és formázzuk meg a szöveget, csak a mezőket kell a szövegszerkesztő program menürendszere segítségével beszúrni a megfelelő helyekre.

Az adatforrás kiválasztása például a *Levelezés* > *Címzett kiválasztása* (illetve az *Eszközök* > *Körlevéltündér*> *Címblokk beszúrása*) menüponttal történik, míg a mezők beszúrása a *Levelezés* > *Adatmező beszúrása* (illetve a *Beszúrás* > *Mező* > *További mezők* dialógusdoboz) lehetőséggel.



► Adatmező beszúrása a törzsdokumentumba (balra Microsoft Word, jobbra LibreOffice Writer)

Ha elkészültünk, egyesítjük a két dokumentumot. Az egyesítés történhet közvetlenül a nyomtatóra, de akár létrehozhatunk egy új dokumentumot is, amely a személyre szóló leveleket *külön szakaszokban* tárolja. Az egyesítést például a *Levelezés* > *Befejezés és egyesítés* menüponttal (illetve a megjelenő *Körlevél* > *Eszköztár* gombjaival) végezhetjük el.

A szakasz a dokumentum egy formailag összetartozó részét jelenti, amelyet szakasztörések határolnak. A szakaszokkal részletesebben a *Nagy dokumentum formázása* fejezetben ismerkedünk meg.

Feladatok

1. Az egyesített dokumentum elkészítése után (de még annak kinyomtatása előtt) kiderül, hogy maradt a szövegben egy helyesírási hiba. Hogyan célszerű azt kijavítani?
2. A körlevelet nem feltétlenül szükséges az adatforrásban szereplő minden címzett részére elküldeni. Hogyan tudjuk ezt beállítani az általunk használt szövegszerkesztő programban?
3. Készítsünk emléklapot az iskolai papírgyűjtésben részt vevő osztályok részére az osztály és a leadott papírmennyiség megadásával!

Stílusok, tartalomjegyzék

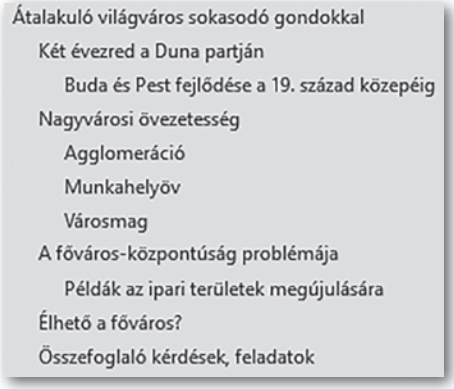
Címsorok kialakítása

Ha egy hosszabb lélegzetű dokumentumot készítünk, akkor a szöveget úgy kell a címek kialakításával tagolnunk, hogy abban az olvasó könnyen eligazodjon. Ehhez a szövegszerkesztő programok kész formátumokat, úgynevezett *címsorstílusokat* kínálnak fel. A címsorstílusok hierarchikusak, például ha egy fejezet címe *Címsor 1* stílusú, akkor azon belül az alcímeket *Címsor 2* stílusú bekezdésekkel alakítjuk ki, míg egy *Címsor 2* stílusú címmel bevezetett részben az alcímeket már *Címsor 3* stílussal, és így tovább.

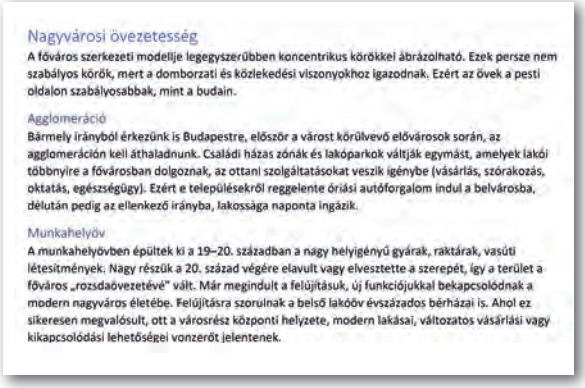
14. példa: Szöveg tagolása címsorok alkalmazásával

Töltsük le a tankönyv weboldaláról a *Budapest.txt* fájlt, és illesszük be egy új dokumentumba! (A szöveg ismerős lehet nyolcadikos földrajztankönyvünk [FI-506010801/1] 80–84. oldaláról.) Tagoljuk címsorokkal a dokumentumot az ábrának megfelelően! (Az ábrán a címsorok szintje a behúzás mértékének megfelelően rendre *Címsor 1*, *Címsor 2*, *Címsor 3.*)

Egy címsorstílus alkalmazásához kattintsunk a bekezdésre, majd válasszuk ki a megfelelő stílust a *Kezdőlap* > *Stílusok* csoportjában (illetve a *Stílusok* menüben)!



► A címsorok hierarchiája a mintában



► A Címsor 2 és Címsor 3 stílusú címsorok a szövegben

Valószínű, hogy a szövegszerkesztő program által alkalmazott stílusok nem felelnek meg az elképzeléseinknek. Ezért a szövegszerkesztő programok lehetőséget adnak arra is, hogy az általuk felkínált stílusokat módosítsuk.

Esetünkben például a *Címsor 2* stílus 16 pontos sötétkék Times New Roman betűvel készült, a bekezdés előtt 18 pont, a bekezdés után 12 pont térköz van. Az összes *Címsor 2* stílusú bekezdés formátumát egyszerre megváltoztathatjuk, ha valamelyikre alkalmazzuk ezeket a beállításokat, majd előírjuk, hogy ezután ilyen legyen a *Címsor 2* stílus. Ehhez például kattintsunk a *Kezdőlapon* jobb gombbal a *Címsor 2* stílusra, és a *helyi menüben* válasszuk a *Címsor 2 stílus frissítése a kijelölés formátumára* (illetve válasszuk a *Stílusok* > *Stílus frissítése*) menüpontot!

A stílus

A bekezdésstílus egy adott bekezdésre vonatkozó beállítások összessége, amelyet a stílus nevével azonosítunk.

Minden bekezdésnek van stílusa. Az alapértelmezett stílus általában a *normál*, míg a címek formázásához az előre definiált *Címsor 1*, *Címsor 2*, … stílusokat használják.

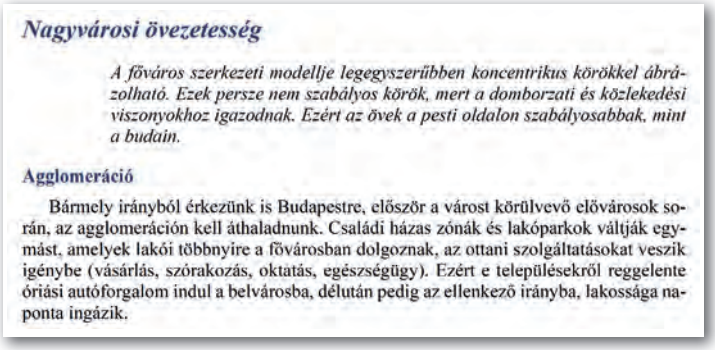
A felhasználó maga is létrehozhat új stílust, ha egy adott bekezdés formátumát új stílusnévvel látja el, és azt azon a néven elmenti.

A stílusok nem függetlenek egymástól, például a címsorstílusok tipikusan a *normál* stílusra épülnek. Ez azt jelenti, hogy ha a normál stílust módosítjuk, a módosítások kihatnak a címsor stílusokra is.

Például módosítsuk a *normál* stílust úgy, hogy az 12 pontos Times New Roman betűkből álljon, utána 6 pontos térköz következzen, sorköze szimpla, az első sor behúzása pedig 0,8 cm legyen! Láthatóan a címsorok első sorának behúzása is módosul.

A főcím utáni bekezdés bevezető jellegű, érdemes máshogy formázni, például a fentieken túl legyen dőlt, bal behúzása legyen 3 cm, a térköz utána pedig 18 pont.

Megformázás után például a *Kezdőlapon* a *Stílusok* lista legördítésével a *Stílus létrehozása* (illetve a *Stílusok* > *Új stílus*) menüpontot választva adjuk meg az új stílus nevét *(bevezető)*. Alkalmazzuk a *bevezető* stílust a *Nagyvárosi övezetesség* címet követő első bekezdésre is!



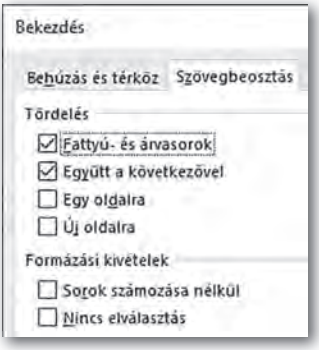
Nagyvárosi övezetesség

A főváros szerkezeti modellje legegyszerűbben koncentrikus körökkel ábrázolható. Ezek persze nem szabályos körök, mert a domborzati és közlekedési viszonyokhoz igazodnak. Ezért az övek a pesti oldalon szabályosabbak, mint a budain.

Agglomeráció

Bármely irányból érkezünk is Budapestre, először a várost körülvevő elővárosok során, az agglomeráción kell áthaladnunk. Családi házas zónák és lakóparkok váltják egymást, amelyek lakói többnyire a fővárosban dolgoznak, az ottani szolgáltatásokat veszik igénybe (vásárlás, szórakozás, oktatás, egészségügy). Ezért e településekről reggelente óriási autóforgalom indul a belvárosba, délután pedig az ellenkező irányba, lakossága naponta ingázik.

► A Címsor 2, a bevezető, a Címsor 3 és a normál stílusok a mintapéldában



► Bekezdés laphatáron (Microsoft Word)

Bekezdések laphatáron

Nem szerencsés, ha a címsor a lap aljára, a hozzá tartozó első bekezdés pedig már a következő oldal tetejére kerül. Ezért célszerű beállítani, hogy *a cím a következő bekezdéssel egy oldalra* kerüljön, vagy *az adott cím új oldalt kezdjen.*

Szakszövegek esetén az is előfordulhat, hogy egy bekezdést egyáltalán nem szabad laphatáron kettévágni. Ilyenkor előírhatjuk, hogy *az adott bekezdés* minden sora *egy oldalra* kerüljön.

Végül az is rontaná az áttekinthetőséget, ha csupán a bekezdés első sora kerülne a lap aljára *(árvasor)* vagy utolsó sora a lap tetejére *(fattyúsor).* A mai szövegszerkesztő programok ezt egy-egy sor átvitelével automatikusan kiküszöbölik, viszont igény esetén ez a funkció kikapcsolható.

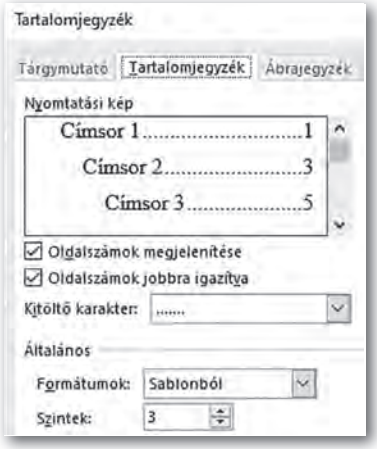
15. példa: Tartalomjegyzék beszúrása

A címsor stílusú bekezdésekből a szövegszerkesztő programok automatikusan el tudják készíteni a **tartalomjegyzéket**. A tartalomjegyzék a címsorok változása esetén nem frissül automatikusan, azt mindig a felhasználónak kell megtennie.

A tartalomjegyzék beszúrásához a dokumentum végén kezdjünk új oldalt (például a CTRL + ENTER billentyűzetkombinációval), majd az új oldal tetejére írjuk be a címet *(Tartalomjegyzék)* és formázzuk meg (például *Címsor 2* stílussal)!

A tartalomjegyzéket például a *Hivatkozások* > *Tartalom* (illetve a *Beszúrás* > *Tartalomjegyzék és tárgymutató*) menüponttal szúrhatjuk be.

Bár gyakran kiválaszthatunk kész formátumot is, célszerűbb „egyéni” beállításokkal dolgozni. Ebben az esetben megadhatjuk, hogy milyen szintig kerüljenek a tartalomjegyzékbe a címsorok (például a *Címsor 3* még benne legyen) megjelenjenek-e az oldalszámok, s ha igen, akkor milyen vonal kösse össze a címet az oldalszámmal (pl. pontozott, szaggatott).



▶ Tartalomjegyzék beállítása (Microsoft Word)

A szövegszerkesztő programok lehetőséget adnak arra is, hogy a tartalomjegyzék oldalszámai hivatkozásként működjenek, amely természetesen elektronikus olvasás esetén hasznos, és például PDF formátumba mentés után is megmarad.

Feladatok

1. Módosítsuk a *normál* stílus betűtípusát például Arialra! Változik-e a *címsor* stílusok vagy a *bevezető* stílus betűtípusa? Mi lehet ennek az oka?
2. Módosítsuk a dokumentumban a *Címsor 1*, illetve a *Címsor 3* stílust is úgy, hogy a betűméret és a térköz a *Címsor 2* megadott értékeinél nagyobb, illetve kisebb legyen! Állítsuk be a címsorstílusokat úgy, hogy ne legyen első soruk behúzva!
3. Amikor új dokumentumot hozunk létre, a szövegszerkesztő programok kész sablonokat kínálnak fel. Kezdjünk a szokásostól eltérő sablonnal egy új dokumentumot! Milyen stílusok vannak benne?

Nagy dokumentumok formázása

Ha nagyobb terjedelmű dokumentumot készítünk (például egy projektmunka beszámolóját), akkor az egységes formázáson túl az olvasót plusz információkkal is el kell látnunk (fejléc, lábléc, oldalszámozás, lábjegyzet, tartalomjegyzék).

Egységes formázás stílusokkal

A stílusok használatával az előző leckében részletesen is megismerkedtünk. Ez akkor hatékony, ha a dokumentumot szabályosan gépeltük be: nem használunk sem fölösleges szóközt, sem bekezdésjelet, sem más különleges karaktert a dokumentum tagolására.

A stílusokat gyakran utólag is módosítjuk. Mivel a *normál* stílus változtatása a többire is kihat, sokszor élnek azzal a megoldással, hogy a főszöveg formázására is létrehoznak egy stílust (például *főszöveg* néven), és a *normált* nem változtatják meg.

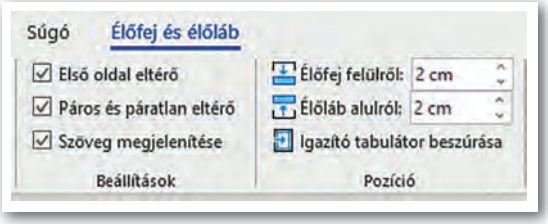
Élőfej és élőláb

Az *élőfej* az oldalak tetején, az *élőláb* az oldalak alján jelenik meg, és az olvasó eligazodását segíti a dokumentumban. Élőfejbe vagy élőlábba szokás tenni például a fejezetcímet vagy az oldalszámot.

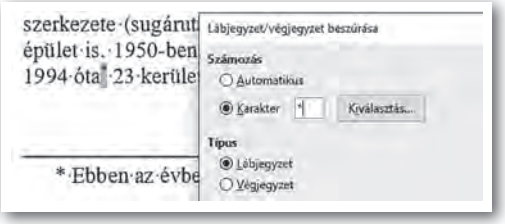
Az élőfej és az élőláb **eltérő lehet az első oldalon.** Az élőfej ugyanis gyakran a fejezet címét tartalmazza, ami az első oldalon amúgy is szerepel.

Ugyanígy gyakran eltér az élőfej és élőláb **a dokumentum páros és páratlan oldalain**. Általában egy könyv bal oldalán szerepelnek a páros, jobb oldalán a páratlan oldalszámok, amik úgy használhatók a legjobban, ha a lap szélén vannak, így elrendezésük egymás tükörképe.

Élőfejet például a *Beszúrás* > *Élőfej és élőláb* > *Élőfej* ponttal szúrhatunk be, de gyakran elegendő csupán kettőt kattintanunk a felső vagy az alsó margóra.



► Élőfej és élőláb beállítása (Microsoft Word)



► Lábjegyzet beszúrása (LibreOffice Writer)

Lábjegyzet

A *lábjegyzet* az olvasó számára szolgáló kiegészítés, amelyet a dokumentum egy adott pontjához kapcsolva általában a lap alján helyeznek el. Néha előfordul, hogy ezeket a kiegészítéseket egyben, a dokumentum végére teszik *(végjegyzet).*

A lábjegyzetre utaló hivatkozást a szövegszerkesztő programok alapértelmezetten egytől kezdve számokkal szúrják be, de beszúráskor megadhatunk más karaktert, például *-ot is. Ha a hivatkozásra mutatunk az egérrel, a lábjegyzet szövege egy buborékban megjelenik, míg ha rákattintunk, akkor a lábjegyzetre ugrik a program.

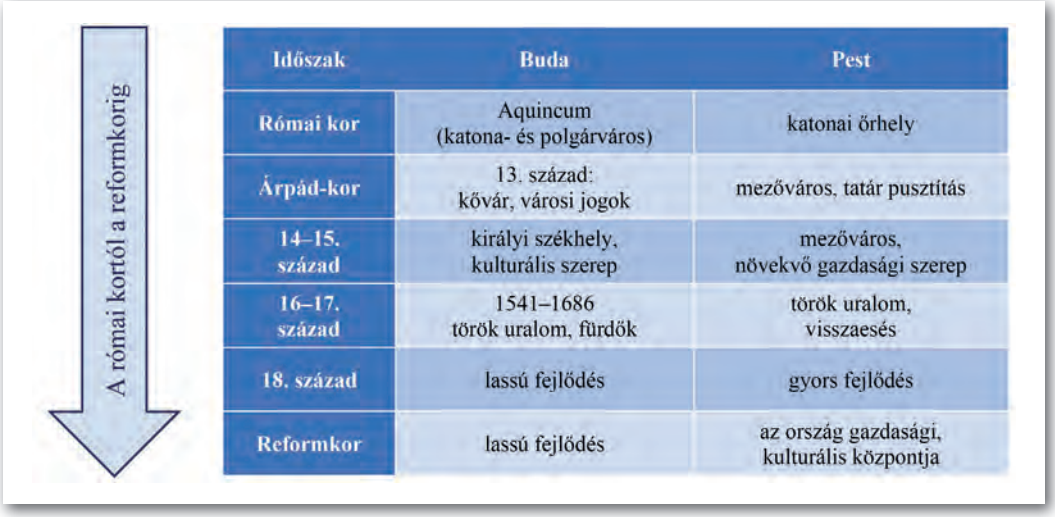
Arról a szövegszerkesztők automatikusan gondoskodnak, hogy a hivatkozási pont és a lábjegyzet egy oldalon maradjon a szöveg átírása esetén is, és csak sok, hosszú lábjegyzet esetén törik meg a lábjegyzet szövegét.

Lábjegyzetet például a *Hivatkozások* > *Lábjegyzet* csoportban (illetve a *Beszúrás* > *Lábjegyzet és végjegyzet* menüponttal) szúrhatunk be a dokumentum adott pontjához.

Táblázatok, képek, ábrák, listák, iniciálék beillesztése

Táblázatokat, képeket, ábrákat, listákat egy hosszabb beszámoló szinte minden esetben tartalmaz. Ezeket a korábbiakban megismert módon hozhatjuk létre.

Különösen *képek beszúrása esetén* fordul elő, hogy a kép nem fér el a főszövegben hozzá tartozó rész közelében, ezért távolabb, esetleg más oldalra kell tennünk. *Az olvasót ilyen esetben képaláírással*, és esetleg a képek számozásával is segítenünk kell.



Időszak	Buda	Pest
Római kor	Aquincum (katona- és polgárváros)	katonai őrhely
Árpád-kor	13. század: kővár, városi jogok	mezőváros, tatár pusztítás
14–15. század	királyi székhely, kulturális szerep	mezőváros, növekvő gazdasági szerep
16–17. század	1541–1686 török uralom, fürdők	török uralom, visszaesés
18. század	lassú fejlődés	gyors fejlődés
Reformkor	lassú fejlődés	az ország gazdasági, kulturális központja

▸ A táblázatos rész egy beszúrt nyíllal az első oldalon látványosan ábrázolja a mondanivalót

Főleg művészeti dokumentumok látványos eleme *az iniciálé*, vagyis *a bekezdés díszes kezdőbetűje.* Beszúrásakor általában megadhatjuk, hogy hány soros legyen, milyen legyen a betűtípusa és besüllyedjen-e a szövegbe. Sajnos a magyar nyelvben a legtöbb bekezdés A betűvel kezdődik, így alkalmazása egyhangú lehet.

Iniciálét például a *Beszúrás* > *Szöveg* csoportban (illetve a *Formátum* > *Bekezdés* > *Iniciálék* fülön) szúrhatunk be.

Szövegdoboz

A szöveg témájához kapcsolódó, nagyobb lélegzetű **kiegészítést gyakran szövegdobozba** (újság esetén „keretes részbe") helyezik el. A szövegdoboz – azon túl, hogy szöveget tartalmaz – lényegében ugyanúgy formázható, mint egy kép: átméretezhető, elforgatható, szöveggel körbefuttatható, szegélyezhető stb.

A szövegdobozt gyakran eltérő betűformátummal is elkülönítik a főszövegtől. Szövegdobozt például a *Beszúrás* > *Szövegdoboz* ponttal illeszthetünk be.

Többhasábos rész, a szakasz fogalma

Többhasábos megoldással főleg újságokban találkozunk, ahol a szedéstükör olyan széles, hogy nehézkes a szemünkkel követni. De többhasábos részt kialakíthatunk egy hosszú dokumentumban is, ha egy szövegrészt így szeretnénk kiemelni vagy elkülöníteni.

Többhasábos részt úgy alakíthatunk ki, hogy a megfelelő szövegrészt kijelöljük, majd például az *Elrendezés* > *Hasábok* ikont (illetve *Formátum* > *Hasábok* menüpontot) választjuk. A hasábok száma és szélessége mellett megadhatjuk azt is, hogy legyen-e hasábelválasztó vonal.

Többhasábos részben akár a hasábok közé is tehetünk képeket, a szöveg úgy is követhető, ilyenkor esztétikusabb, ha ezek legfeljebb egy-egy hasáb felét foglalják el.

▶ Többhasábos rész kialakítása (Microsoft Word)

Ha csak a dokumentum egy része lesz többhasábos, akkor a szövegszerkesztők a dokumentumot három *szakaszra* bontják. Egyik a többhasábos rész előtti, másik a többhasábos rész, harmadik a többhasábos rész utáni szakasz.

A szakasz a dokumentum formailag önálló része, *az oldal jellemzőit,* például margók, tájolás, oldalszámozás, hasábok száma stb. szakaszonként külön-külön is beállíthatjuk. Például egy álló tájolású szövegbe egy fekvő tájolású oldalt külön szakaszként illeszthetünk.

Szakaszokat szakasztörés beszúrásával hozhatunk létre, például az *Elrendezés* > *Töréspontok* (illetve *Beszúrás* > *Szakasz*) menüpont segítségével.

Elválasztás

Egy sorkizárt bekezdéseket tartalmazó dokumentumban – különösen akkor, ha a szövegrész keskeny, mert képeket illesztettünk be vagy többhasábos részt is kialakítottunk – a széles szóközök miatt csúnya, úgynevezett utcák alakulhatnak ki. Az utcákat legegyszerűbben a szavak elválasztásával küszöbölhetjük ki.

Az elválasztást például az *Elrendezés* > *Oldalbeállítás* csoportban (illetve az *Eszközök* > *Nyelv* menüpontban) kapcsolhatjuk be. Ügyeljünk arra, hogy a nyelvi modulok néha még tévednek, így elképzelhető, hogy az automatikusan kialakított elválasztást felül kell bírálnunk, például feltételes kötőjelek beszúrásával.

Feladatok

Az alábbi feladatokhoz tartozó mintákat a következő oldalakon találjuk.

1. Nyissuk meg a *Budapest* nevű dokumentumot, amelyet a korábbiakban stílusokkal már megformáztunk!
2. Állítsuk be a felső margót 2,5, a többit 3,0 cm-re!
3. Alakítsuk ki az élőfejet és az élőlábat a következőképpen:
   Az első oldalon ne legyen sem élőfej, sem élőláb!
   A páros oldalakon a *Világvárosok* szó, a páratlanokon a *Budapest* szó legyen!
   Az oldalszám kerüljön az élőlábba!
   A páros oldalakon az élőfejet és élőlábat balra, a páratlan oldalakon jobbra zárjuk!
4. Szúrjuk be a lábjegyzetbe a mintának megfelelően a *Két évezred a Duna partján* című rész utolsó mondatához *-gal kapcsolva az *Ebben az évben vált önálló kerületté Soroksár.* mondatot!
5. A *Buda és Pest fejlődése…* című részben lévő, tabulátorokkal tagolt adatokat alakítsuk táblázattá, és formázzuk meg a *Táblázatok, képek…* című részben lévő mintának megfelelően!
6. A *Két évezred…* című rész első bekezdésének kezdőbetűjét alakítsuk iniciálévá! Az iniciálét követő első két szó legyen kiskapitális!
7. Szúrjuk be a tankönyv weblapjáról letölthető *Övezetek* képet az *Agglomeráció* és *Munkahelyöv* mellé – az oldalarányok megtartásával – 5 cm magasságúra átméretezve, a szöveggel négyzetesen körbefuttatva! Ábraszövegként írjuk a *Budapest elméleti szerkezete* magyarázatot, majd a képet és ábraszöveget foglaljuk csoportba!
8. A *Példák az ipari…* című részben lévő bekezdéseket alakítsuk felsorolássá, a felsorolást jelző egyedi szimbólum utaljon a városra!
9. Az *Összefoglaló kérdések…* cím alatti részt a *Fogalmakig* alakítsuk számozott listává!
10. A *Fogalmak* szóval kezdődő szavakat helyezzük el egy 5 cm széles szövegdobozba a főszövegtől eltérő formátumú betűkkel (pl. 10 pontos Calibri), és a szövegdobozt igazítsuk az összefoglaló kérdések mellé!
11. Az *Élhető főváros* címhez tartozó szöveget alakítsuk hasábelválasztó vonallal kéthasábossá!
12. Alkalmazzunk a szövegben automatikus elválasztást!
13. Frissítsük a dokumentum végén lévő tartalomjegyzéket!
14. Mentsük el a szöveget a szövegszerkesztő alapértelmezett formátumában, majd készítsünk belőle egy PDF formátumú fájlt is!

Nagyvárosi övezetesség

A főváros szerkezeti modellje legegyszerűbben koncentrikus körökkel ábrázolható. Ezek persze nem szabályos körök, mert a domborzati és közlekedési viszonyokhoz igazodnak. Ezért az övek a pesti oldalon szabályosabbak, mint a budain.

Agglomeráció

Bármely irányból érkezünk is Budapestre, először a várost körülvevő elővárosok során, az agglomeráción kell áthaladnunk. Családi házas zónák és lakóparkok váltják egymást, amelyek lakói többnyire a fővárosban dolgoznak, az ottani szolgáltatásokat veszik igénybe (vásárlás, szórakozás, oktatás, egészségügy). Ezért e településekről reggelente óriási autóforgalom indul a belvárosba, délután pedig az ellenkező irányba, lakossága naponta ingázik.

Budapest elméleti szerkezete

Munkahelyöv

A munkahelyövben épültek ki a 19–20. században a nagy helyigényű gyárak, raktárak, vasúti létesítmények. Nagy részük a 20. század végére elavult vagy elvesztette a szerepét, így a terület a főváros „rozsdaövezetévé" vált. Már megindult a felújításuk, új funkciójukkal bekapcsolódnak a modern nagyváros életébe. Felújításra szorulnak a belső lakóöv évszázados bérházai is. Ahol ez sikeresen megvalósult, ott a városrész központi helyzete, modern lakásai, változatos vásárlási vagy kikapcsolódási lehetőségei vonzerőt jelentenek.

Városmag

A felújított városrészek népességszáma nő. A pesti oldal belvárosa az V. kerület, a városmag. Közigazgatási, pénzügyi és kereskedelmi szerepe mellett sok turistát is vonz. A budai városrészek központja a Várnegyed, közigazgatási, kulturális és idegenforgalmi szerepe kiemelkedő.

A főváros-központúság problémája

Magyarország településhálózatának egyik problémája a túlzott főváros-központúság. Népességben vagy gazdaságban betöltött túlsúlya aránytalanul nagy. A főváros hazánk GDP-jének 40%-át állítja elő. A hazai tőkebefektetések nagy része is ide irányul, de Budapest a határokon túlról is képes tőkét vonzani, pénzügyi, kulturális és idegenforgalmi szerepe pedig egész Kelet-Közép-Európában meghatározó.

Példák az ipari területek megújulására

- Millenáris kulturális központ (Ganz Villamossági Művek)
- Nemzeti Színház (vasúti raktárak, műhelyek)
- Művészetek Palotája (vasúti raktárak, műhelyek
- Bálna (Közraktár)

Átalakuló világváros sokasodó gondokkal

Hazánk lakosságának egynegyede, két és félmillió ember a fővárosban vagy annak közvetlen környezetében él. Ha Budapest megfelelő életminőséget tud biztosítani lakossága számára, turistákat és tőkét tud magához vonzani, akkor a közép-európai régió vezető központja lehet. Ehhez azonban a leromlott állapotú városrészek rendezése, a közlekedési problémák megoldása és az európai uniós pénzügyi folyamatokba való erőteljesebb bekapcsolódás szükséges.

A főváros népességszáma 1980-ig a természetes szaporodás és a bevándorlás miatt növekedett. Az 1980–90-es években viszont fogyatkozott a népessége, lakói közül sokan a várost körülvevő kisebb településekre költöznek. A 2000-es évtized második felétől azonban újra emelkedni kezdett a főváros lakosságszáma. Hogy megértsük a népességszámváltozásának az okait, meg kell ismerkednünk a város szerkezetével is.

Két évezred a Duna partján

FŐVÁROSUNK TERÜLETÉN már több ezer éve kisebb települések alakultak ki a terület kedvező földrajzi adottságai miatt. A Budai-hegység és a Pesti-síkság találkozásánál (a Margit-sziget és a Gellért-hegy között) lehetett könnyen átkelni a folyón. Így itt kereszteztek egymást a Duna mentén haladó észak–déli és a Kárpát-medencét átszelő kelet–nyugati irányú kereskedelmi útvonalak. Budát, Pestet és Óbudát 1873-ban egyesítették, ettől kezdve beszélhetünk Budapestről. A város az első világháborúig hatalmas növekedésen ment keresztül, igazi világvárossá fejlődött. Ekkor alakult ki mai szerkezete (sugárutak, körutak, terek) és ekkor épült számos, a mai városképet meghatározó épület is. 1950-ben 23 környező település hozzácsatolásával érte el a város mai kiterjedését. 1994 óta* 23 kerület alkotja.

Buda és Pest fejlődése a 19. század közepéig

A római kortól a reformkorig

Időszak	Buda	Pest
Római kor	Aquincum (katonai- és polgárváros)	katonai őrhely
Árpád-kor	13. század: kővár, városi jogok	mezőváros, tatár pusztítás
14–15. század	királyi székhely, kulturális szerep	mezőváros, növekvő gazdasági szerep
16–17. század	1541–1686 török uralom, fűtőlök	török uralom, visszanövés
18. század	lassú fejlődés	gyors fejlődés
Reformkor	lassú fejlődés	az ország gazdasági, kulturális központja

* Ebben az évben vált önálló kerülletté Soroksár.

- Dorottya-udvar (katonai egyenruhagyár)
- Óbudai Gázgyár
- Haller Garden (iparterep)
- Aréna Plázа (Ügető)
- MOM Park (Magyar Optikai Művek)
- WestEnd City Center (vasúti raktárak, műhelyek)
- Sárkány Center (Kőolajipari Gépgyár)

Élhető a főváros?

A főváros által nyújtott előnyökért cserébe a lakosságnak számos nagyvárosi problémával kell megküzdenie. A belső városrészekben európai összehasonlításban rendkívül kicsi a zöldfelületek aránya. A Budai-hegység és a pesti külvárosok erdői, rétjei pedig gyakran a lakóingatlanok terjeszkedésének áldozatául esnek.

Az M0 autópálya legnagyobb részének kiépülésével a városon áthaladó forgalom jelentősen csökkent. A városon belüli és az agglomerációból érkező autóforgalom azonban továbbra is óriási terhet jelent Budapest lakosságának. Ennek kezeléséhez a négy metróvonal mellett a HÉV és a városi vasútvonalak fejlesztésére is szükség van. Az utóbbi időben kiépült P+R parkolók a városba autóval érkezőket a városon belül a tömegközlekedés igénybevételére szeretnék rászoktatni. A belvárosi közlekedési és parkolási gondok enyhítését szolgálhatja a városi kerékpárosút-hálózat kiterjesztése és a behajtási díj („dugódíj") bevezetése is. Ma már elsősorban nem az ipar, hanem a közlekedés okozza a levegő szennyezését, mint ahogy ez a zajszennyezés fő forrása is.

Összefoglaló kérdések, feladatok

1. Milyen szerepe volt a főváros kialakulásában a természeti adottságoknak?
2. Hogyan vált Budapest fővárossá, majd világvárossá?
3. Milyen szerepe van az egyes öveknek a város gazdasági életében?
4. Mely városrészeket mutatnád meg külföldről érkezett barátodnak?

Fogalmak

Buda • Pest • Óbuda • kerület • agglomeráció • elővárosi • ipari öv • külső lakóöv • munkahelyöv • belső lakóöv • belváros • városmag • nemzetközi szerepkör • városi problémák • világváros

Számítógépes grafika

Pixelgrafikus ábrázolás

A külvilágból érkező jelek közül a környezetünkről a legtöbb információt szemünk segítségével – fényérzékeléssel – kapjuk. Nem véletlen tehát, hogy a reklámoknál, termékek csomagolásánál különösen nagy hangsúlyt fektetnek a cégek a külső megjelenésre. Ebben a fejezetben rövid betekintést kaphatunk a pixelgrafikus képszerkesztés, képmanipulálás lehetőségeibe – feliratok, plakátok készítésén keresztül.

Pixelek, képpontok

A digitális képábrázolásnak azt a fajtáját, amikor minden egyes képpontról eltároljuk az információt, pixelgrafikus (raszteres) képábrázolásnak nevezzük. Mobiltelefonok, digitális fényképezőgépek esetében a 12 megapixeles kamera azt jelenti, hogy 12 millió képpontot fog tartalmazni a kép. Amennyiben a kamera 4 : 3-as képarányú képet készít (ez nem más, mint a kép szélességének és magasságának az oldalaránya), akkor 4000 képpont lesz vízszintesen és 3000 képpont lesz függőlegesen. Minél több képpontot tud érzékelni a kameránk, annál részletgazdagabb lesz az általa készített kép.

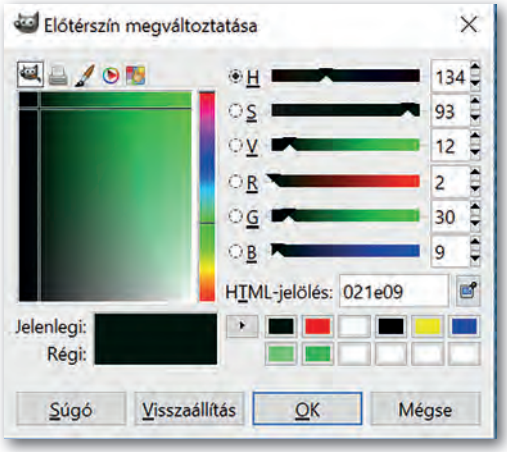
► Egy profi fényképezőgéppel készített kép adatai

Színmélység

Egy képnek fontos jellemzője, hogy mennyire színes. **Az egy képponton ábrázolható színek számát színmélységnek nevezzük.** 2 szín megkülönböztetéséhez 1 bitet elég használnunk, 256 szín megkülönböztetéséhez már 8 bit kell, azaz 1 bájt. Ám ennyi szín nem alkalmas arra, hogy élvezhető módon élethű tájképet mutassunk be. Jellemzően **ma már 24 bites színmélységgel találkozunk,** ahol több mint 16 millió színt különböztetünk meg. A 24 bites színmélységet **true colornak nevezzük.**

Színábrázolás

Milyen tulajdonságokat tárolunk 24 biten? **Az additív színkeverés esetében a színek három alapszínből keverhetők ki: vörös, zöld és kék.** A színek angol nevének rövidítéséből (red, green, blue) adódik az **RGB színkeverés** elnevezés. Korábban szövegszerkesztés esetén a színek meghatározásakor már találkozhattunk is a három színösszetevő megadásával. Ott minden egyes összetevőnél 0-tól 255-ig egy egész számot kellett megadni, ami azt határozza meg, hogy mennyire „sok” legyen az adott alapszínből a keverésnél. Ügyeljünk az összetevők sorrendjére, ne cseréljük fel azokat! **Először mindig a vörös, majd zöld, végül a kék komponenst kell megadni!**



▶ Színek választása palettáról, vagy RGB kóddal (2,30,9)

Létezik olyan színábrázolás is, amikor ezt a **24 bitet kiegészítjük még 8 bittel, amin az adott pont átlátszóságát határozzuk meg** egy 0-tól 255-ig terjedő skálán. **Szakkifejezéssel ezt a 8 bitet szokás alfa-csatornának nevezni,** egyes képszerkesztő programoknál az értékét százalékosan szoktuk megadni. Az alfa-csatornával a későbbiekben, a rétegekkel való ismerkedéskor még találkozni fogunk.

Képek megjelenítése

Hiába van egy 16 megapixeles képünk, ha a megjelenítő eszközünk (monitor, nyomtató) nem tudja visszaadni a szépségét. Egy digitális megjelenítő eszköz **felbontás**án azt értjük, hogy **inchenként (2,54 cm) hány képpont**ot ábrázolunk – angolul **p**ixel **p**er **i**nch, amelynek rövidítéséből származik a ppi mértékegység. A 300 ppi felbontás tehát azt jelenti, hogy 2,54 cm-en 300 képpontot ábrázolunk. **Nyomtatók** esetében a **dot per inch** (inchenkénti pontok száma), azaz **dpi** mértékegységet használjuk, hiszen nyomtatáskor a papíron nem pixelek, hanem pontok keletkeznek. Minél nagyobb a felbontás, annál jobb minőségű a kapott kép.

Fájlformátumok

BMP (bitmap) kiterjesztésű állományok esetén minden képpontnak a színét eltároljuk. Tekintsünk egy 24 bites színmélységű, 16 megapixeles képet. Ennek mérete 48 x 10^6 bájt. (Ettől valójában egy picit – 54 bájttal – több, mert az állomány tartalmaz még egyéb információkat is.) Ezt átváltva megközelítőleg az állomány mérete 46 MB-os lesz, ami egyáltalán nem barátságos, ezért **ritkán használjuk ezt a fajta fájlformátumot.**

▶ Ennek a 36 megapixeles BMP képnek a fájlmérete 103 MB

JPG vagy JPEG (Joint Photographic Experts Group) kiterjesztésű állományoknál **veszteséges tömörítést** alkalmaznak annak érdekében, hogy az állomány mérete csökkenjen. A veszteséges tömörítés elég kézenfekvő, hiszen az emberi szem egyrészt nem tud megkülönböztetni 16 millió színt, másrészt az egymáshoz közel eső pontokat, ha hasonló színekről van szó, egyformának látjuk. **A veszteséges tömörítés azt jelenti, hogy a tömörítés alkalmazása után már az eredeti állományt nem tudjuk visszaállítani bitről bitre.** Például ha megnyitunk egy BMP kiterjesztésű képet egy képszerkesztő programmal, akkor a **Fájl / Mentés másként (File / Save As) parancsot választva** más fájlformátumba tudjuk menteni. A mentés paranccsal **hajtjuk végre a veszteséges tömörítést.** E formátum alkalmazása minőségromlással jár, de akár századára is csökkenthetjük a kép tárolási méretét úgy, hogy még élvezhető maradjon. **Mobiltelefonok, fényképezőgépek ilyen formátumban mentik el a fényképeket.** Az interneten is gyakran találkozhatunk ilyen formátumú képekkel.

► Ennek a 36 megapixeles JPG képnek a fájlmérete 20,4 MB

PNG (Portable Network Graphics) fájlformátum esetében a fájlok mérete szintén jóval kisebb, mint a BMP kiterjesztésnél, a **veszteségmentes tömörítés**nek köszönhetően. **Veszteségmentes tömörítés esetén az eredeti állomány bitről bitre visszaállítható.** Az előző két fájlformátummal ellentétben **alkalmas az alfa-csatorna kezelésére.** A későbbiekben ezt tartsuk szem előtt, és ha a munkánk során használtunk alfa-csatornát, akkor ilyen formátumban mentsünk! A JPG- és BMP- formátumok nem alkalmasak arra, hogy a kép egy jól körülhatárolt, kijelölt területén eltárolják az átlátszatlanságot.

► Ennek a 36 megapixeles PNG képnek a fájlmérete 41,5 MB

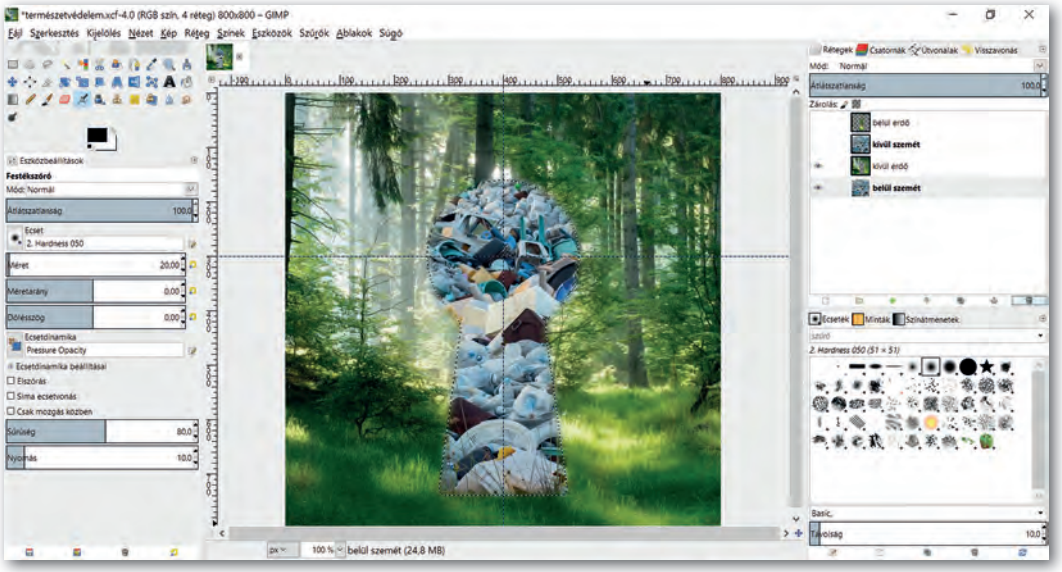
A tervrajzokról, jellemzően vonalas ábrákat, geometriai alakzatokat tartalmazó képekről elmondható, hogy nagyon kevés színt használnak. Az ilyen típusú állományok tárolására használjuk például az SVG- – nem pixelgrafikus – formátumot. Erről a későbbiekben a *Vektorgrafika* fejezetben lesz szó.

Pixelgrafikus képszerkesztő programok

Pixelgrafikus képszerkesztő programok széles választékával találkozhatunk az interneten. A sok közös szolgáltatás mellett mindegyik kínál valami egyedi pluszt, amiért azt a szoftvert érdemes választani.

Professzionális fényképszerkesztéshez, utómunkákhoz az **Adobe Lightroom** és **Adobe Photoshop** mellett érdemes megemlíteni **Pixlr Editort,** aminek kifejezett előnye, hogy online, mobil és asztali alkalmazási lehetőséget is biztosít. A **Corel PaintShop Pro** segítségével tudunk 360°-os kamerával felvett képeket szerkeszteni, és gazdag grafikus sablontárral rendelkezik. A portrék retusálására specializálódott szoftverek közül a **PortraitPrót** érdemes megemlíteni. Ezeket a programokat különböző konstrukciókban lehet megvásárolni.

Egy hétköznapi felhasználó igényeit azonban az ingyenes szoftverek is bőven kielégítik. Ilyen például a **GIMP**, ami az Adobe Photoshop vagy Lightroom alternatívája lehet. A mintapéldák megoldásait ezzel a szoftverrel adjuk meg!



► A GIMP 2.8 (GNU Image Manipulation Program) felhasználói felülete

Kérdések, feladatok

1. A mobiltelefonunk kamerája 16 megapixeles. Mit jelent ez?
2. Nézzünk utána, hány megapixeles a mobiltelefonunk kamerája! Készítsünk vele egy képet, és nézzük meg a keletkezett fájl adatait egy képnézegető szoftverrel! Hány képpontot tartalmaz vízszintesen és függőlegesen? Ezekből az adatokból kiszámolhatjuk, hány megapixeles a kamera! Alkalmazzunk kerekítést!
3. Mit jelent a 16 : 9 képarány? Az előző feladatban készített képnek mi a képaránya?
4. Tudjuk állítani a mobiltelefonunkon a képarányt? Nézzünk utána!
5. Keressünk rá az interneten, hogy mekkora a Full HD felbontás! Milyen eszközök esetében találkozhatunk ezzel a kifejezéssel?
6. A KRESZ-ben vannak veszélyt jelző táblák. Keressük meg az interneten, hogy néz ki az a tábla, ami veszélyes lejtőre figyelmeztet! Pixelgrafikus képszerkesztő program segítségével (pl. Paint) készítsük el a táblát! A kép mérete 400 x 400 pixel legyen! Mentsük el a képet 24 bites BMP-formátumban, majd JPG- és PNG-formátumban `tábla` néven!
7. Mekkora lesz az előző feladatban elkészített `tábla.bmp` állomány mérete? Számoljuk ki! Fájlkezelő segítségével ellenőrizzük magunkat, és nézzük meg az állomány méretét!

Kijelölések, pozicionálás, alakzatok készítése

Kijelölések

Az egyik legalapvetőbb művelet, amire szükségünk van pixelgrafikus képszerkesztéskor, **a kijelölés, mivel minden művelet a kijelölt területre vonatkozik.** Ez nem újdonság, hiszen például szövegszerkesztésnél is így jártunk el a karakter- vagy bekezdésformázáskor. Csakhogy most jóval „színesebb" a kijelölések palettája, amiből választhatunk. A szoftvergyártók szerencsére a kijelölési módokhoz és minden művelethez igyekeztek hasonló ikonokat választani, így minden programnál könnyen el tudunk igazodni.

A kijelölési módszereket azért is fontos ismerni, mert **nem minden pixelgrafikus programban vannak alakzatok.** Ezekben a szoftverekben a **kijelölés körberajzolásával (stroke) tudjuk elkészíteni azokat.** Az alakzatok **kitöltését festékes vödörrel végezzük.**

	Téglalap kijelölés		Mozgatás
	Ellipszis kijelölés		Nagyítás, kicsinyítés
	Lasszó kijelölés		Forgatás
	Szín alapján egybefüggő terület kijelölése		Tükrözés
	Szín szerinti kijelölés		Igazítás
	Útvonal létrehozása		Szöveg elhelyezése

▸ Néhány képszerkesztő eszköz ikonja (GIMP)

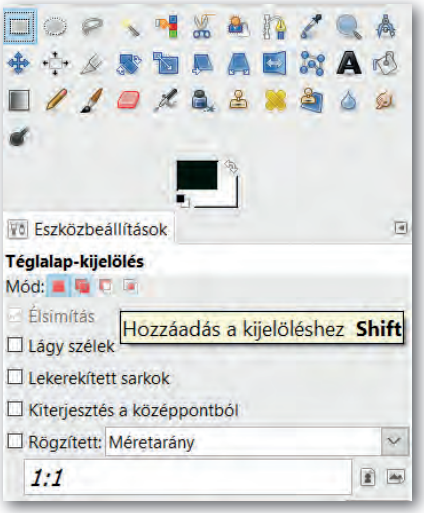
Haladjunk szépen végig a kijelölőeszközökön (szoftverektől függően minimális eltérések lehetnek, az alábbi leírás a mindenki számára ingyen elérhető GIMP szabad szoftver kijelölési műveleteit írja le):

- Az első vagy második ikonra kattintva **téglalap vagy ellipszis alakú területek** kijelölését végezhetjük el. A rögzített méretarány ki-, illetve bekapcsolását kijelölés közben a Shift billentyű lenyomásával tudjuk megtenni. Amennyiben ez az arány 1 : 1 volt, a kijelölés szabályos alakzatúvá változik (azaz négyzet vagy kör lesz).
- A következő jellemző kijelölési mód a **szabadkézi kijelölés,** aminek két változata van. Ha a bal gombot nyomva tartjuk, és úgy mozgatjuk az egeret, akkor a határolóvonal az egér mozgásának megfelelő lesz. Ha csak bal gombbal egyszer kattintunk, akkor két kiválasztott pontot egyenes vonallal köt össze. Ezt a két módszert lehet egy kijelölésen belül kombinálni. A kijelölést befejezhetjük úgy, hogy zárjuk, azaz a kijelölés kezdőpontjába kattintunk. Amennyiben még nem zártuk le a kijelölést, akkor **az egér bal gombjával duplán kattintva – egy összekötő szakasszal – automatikusan megtehetjük azt.** Ennek a kijelölési módnak az ikonja jellemzően egy lasszó minden szoftvernél.

- A következő ikon **a varázspálca, amely egybefüggő területet jelöl ki szín alapján.** Az objektumok hátterének eltávolításához előszeretettel használjuk a törlés műveletével együtt.
- Végül, amiről még szó lesz, az a **szín szerinti kijelölés,** ami hasonló színű területeket jelöl ki a képen.

További kijelölések esetén a következő billentyűkombinációkat alkalmazva változtathatjuk a kijelölési módot:

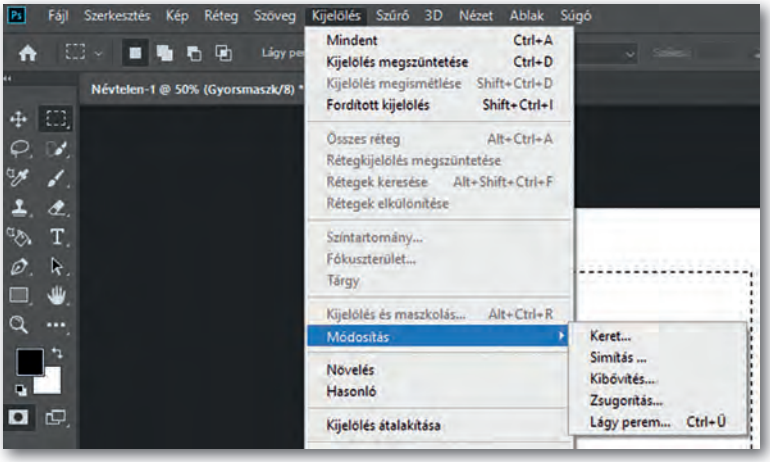
- **Shift** lenyomása estén **hozzáadódik** az új kijelölés az előzőhöz.
- **Ctrl** lenyomása esetén az előző kijelölésből **eltávolítja a közös részeket.**
- **Ctrl + Shift** lenyomása esetén **a metszetet veszi a korábbi kijelöléssel.**



► Váltás a kijelölési módok között (GIMP)

A kijelölésekhez kapcsolódóan hasznos műveletek még a következők:

- Kijelölés megszüntetése.
- Kijelölés megfordítása (invertálása).
- Kijelölés szűkítése megadott pixellel.
- Kijelölés bővítése megadott pixellel.
- Kijelölés keretté alakítása. Ezt úgy tudjuk elképzelni, mintha nem egy vékony vonallal végeznénk a kijelölést, hanem szélesebb sávval. Kiterjedést adunk a kijelölésnek.



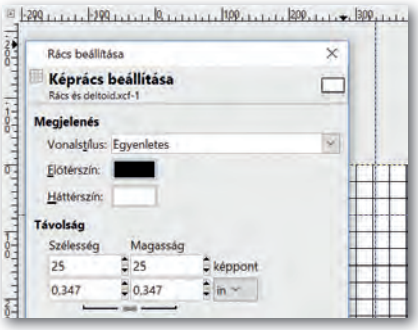
► Kijelölési lehetőségek (Photoshop)

A fenti műveleteket szoftverfüggően menüből vagy billentyűkombináció segítségével érhetjük el!

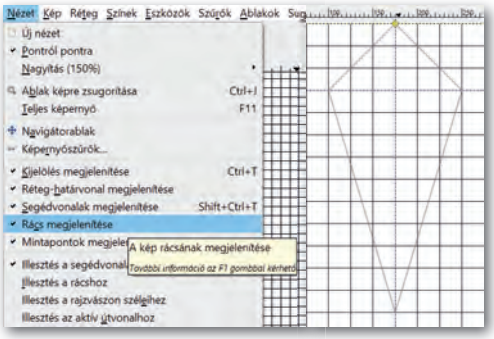
Pozicionálást segítő eszközök

A kijelöléseket és az objektumok pozicionálását rácsok és segédvonalak segítik. Ezeknek a tulajdonságát általában a menün keresztül érjük el, de segédvonalakat a vonalzóról bal gomb segítségével is tudunk „behúzni” a megfelelő pozícióba.

A rácsok és illesztések megjelenítését és a hozzájuk való illesztést bekapcsolva egyszerűen készíthetünk szimmetrikus alakzatokat!



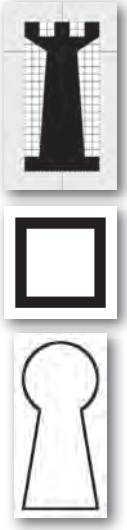
► Az ábrán kék szaggatott vonallal látszanak a segédvonalak és folytonos fekete vonallal a rácsok



► Deltoid alakú terület kijelölése rácsok és segédvonalak segítségével

Kérdések, feladatok

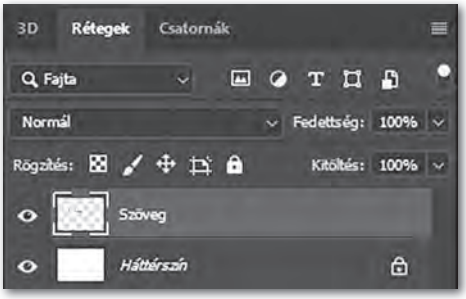
1. Milyen kijelölési módokat ismerünk a képszerkesztő programban?
2. Milyen eszközöket ismerünk a pixelgrafikus szerkesztőprogramokban alakzatok rajzolására?
3. A sakkbábuk közül a bástya egy közkedvelt figura. Speciális helyzetekben lehetőség van arra, hogy a királlyal együtt, egyszerre lépjen. Ezt sáncolásnak nevezzük. Nézzünk utána az interneten, mit jelent a sáncolás, majd készítsük el a minta alapján a fekete figura képét. A kép mérete 200 × 400 px legyen! Az alakzat legyen szimmetrikus! Az ábrán látható rácsok 20 pixelenként vannak, a vízszintes segédvonal 80 pixelnél. Mentsük a képet `3bastya.png` néven.
4. Készítsük el a mintán látható képkeretet! A kép mérete 400 × 400 px legyen! A képkeret vastagsága 40 px. Munkánkhoz jelenítsük meg a rácsokat! Milyen rácsközöket érdemes választani? Mentsük a képet `4kepkeret.png` néven!
5. Készítsük el a minta alapján egy szimmetrikus kulcslyuk körvonalazott ábráját! A kép 200 × 400 px legyen! A kulcslyuk teteje egy 160 px átmérőjű körből legyen, az alja pedig egy trapézból! Milyen kijelölési módot kell használni? Mentsük a képet `5kulcslyuk.png` néven!



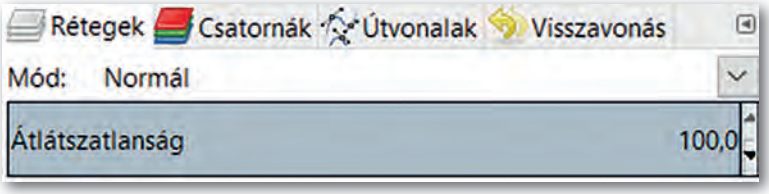
Rétegek, átlátszóság – átlátszatlanság, alfa-csatorna

A réteg

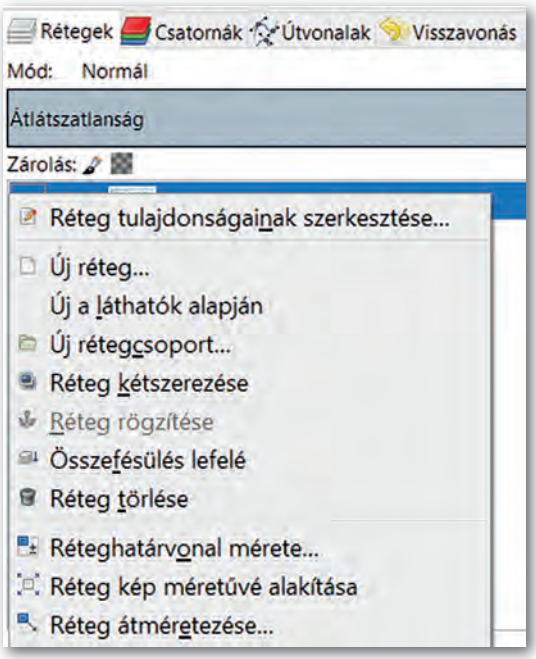
A pixelgrafikus képszerkesztők egy része alkalmas arra, hogy rétegeket (layer) kezeljenek. Az ilyen programok a képeket **egymásra helyezett képekből** – melyek az úgynevezett **rétegeken** foglalnak helyet – hozzák létre. A külön rétegekre elhelyezett képek tulajdonságait bármikor módosíthatjuk. Ezért amikor a vágólap tartalmát beillesztjük, mindig döntenünk kell, hogy a tartalom hová kerüljön: új rétegre vagy a beillesztés műveletekor éppen aktív rétegre (réteg rögzítése). Amíg ezt nem tesszük meg, a beillesztett tartalom egy ideiglenes rétegen foglal helyet.



► Rétegkezelő (Photoshop)



► Réteg- és útvonalkezelő (GIMP)



► Műveletvégzés a rétegkezelőben (GIMP

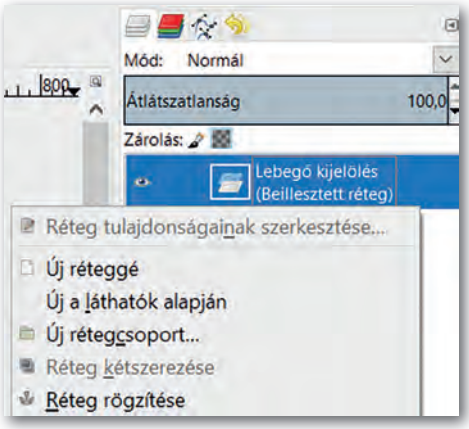
A réteggel kapcsolatos műveleteket (pl. törlés, kétszerezés, rétegek egyesítése) legkönnyebben a rétegkezelőben végezhetjük el. Kattintsunk jobb gombbal a megfelelő rétegen, és válasszuk ki a kívánt műveletet! Ügyeljünk arra, ha egy rétegen kijelölést végzünk, majd egy másik rétegen folytatjuk a munkát, akkor a kijelölés az új rétegen is aktív marad. Ezt majd a feliratok készítésénél fogjuk használni. Továbbá minden réteg megjelenítését egyenként tudjuk állítani. Ha több réteggel dolgozunk, akkor lehetnek olyan rétegek, amelyek munkánkat vizuálisan akadályozzák. Ezen rétegek megjelenítését a szem ikonra kattintva kapcsoljuk ki ideiglenesen. Sorrendjüket bal egérgombbal megfogva és mozgatva tudjuk módosítani a **rétegkezelő**ben.

Átlátszatlanság

A homályos vagy kettős látás lehet fáradtság, kimerültség jele, de utalhat komoly betegségre is, mint például a stroke-ra (agyvérzés). A rétegek és a hozzájuk rendelt átlátszatlanság segítségével könnyen bemutatható a kettős látás jelensége. A feladat végrehajtásához válasszunk egy rétegek kezelésére alkalmas pixelgrafikus programot! A műveletek végrehajtására használhatunk menüt, de a jól bevált billentyűkombinációk is alkalmazhatók.

1. példa: Rétegek átlátszatlansága

1. Nyissunk meg használni kívánt képet új rétegként!
2. Jelöljük ki az egész réteget!
3. A kijelölt területet másoljuk le, és illesszük be új rétegként! Ne feledjük, hogy a beillesztés után egy ideiglenes rétegre kerül a tartalom, amiről dönteni kell, hol helyezkedjen el! Válasszuk az új réteg lehetőséget jobb gombbal kattintva a lebegő rétegen a rétegkezelőben! Jelenleg az újonnan beillesztett kép eltakarja az eredetit, hiszen ez a réteg került legfelülre.
4. Az új rétegen továbbra is kijelölve maradt a tartalom, így az áthelyező eszköz segítségével mozgassuk el a képet tetszőleges irányban néhány pixellel arrébb!



► Művelet lebegő réteggel (GIMP)

5. A legfelső réteg átlátszatlanságát (opacity) állítsuk 50%-ra!
6. Mentsük a munkánkat `kettoslatas` néven a szoftver alapértelmezett formátumában, illetve exportáljuk PNG-formátumban ugyanezen a néven!

Figyeljük meg, hogy a felső réteg alatt átlátszik az alsó, de nagyon zavaró a rétegek találkozásánál az éles határvonal.

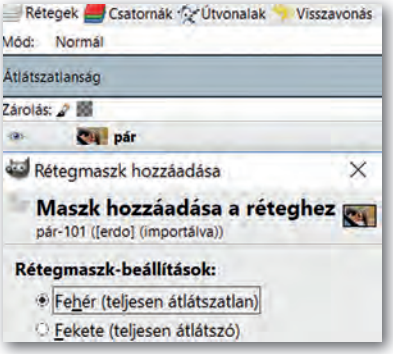

► Éles határvonal a rétegek találkozásánál

Átlátszatlanság állítása rétegmaszk segítségével

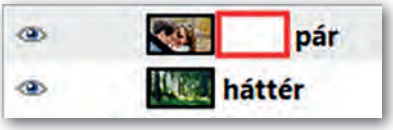
Minden réteghez lehetőségünk van rétegmaszkot hozzáadni, amelyen fehér színnel tudjuk bejelölni a réteg átlátszatlan területeit, fekete színnel az átlátszókat. Fekete-fehér színátmenet segítségével fokozatos áttűnést tudunk elérni az átlátszó és átlátszatlan területek között.

2. példa: Háttér módosítása rétegmaszk segítségével

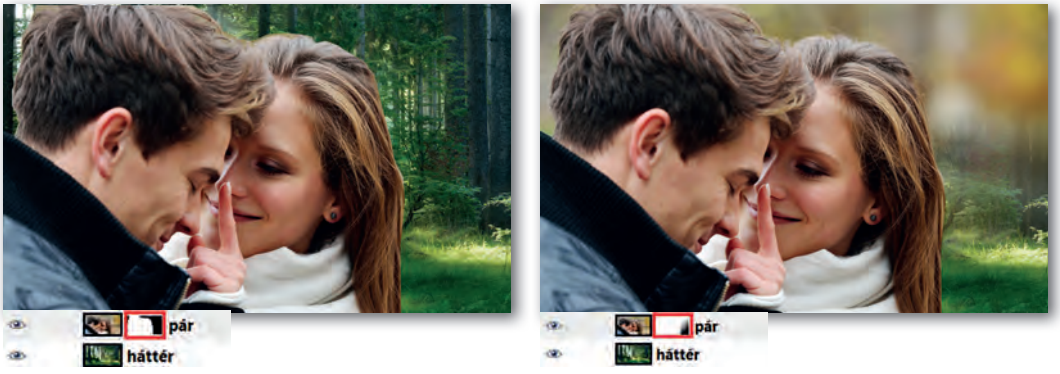
1. Nyissuk meg az `erdo.jpg` képet új rétegként és jobb gomb segítségével nevezzük el *háttérnek* a rétegkezelőben!
2. Nyissuk meg a `par.jpg` képet új rétegként és jobb gomb segítségével nevezzük el *párnak* a rétegkezelőben!
3. Adjunk fehér rétegmaszkot a *pár* réteghez jobb gomb segítségével a rétegkezelőben! Így a *pár* réteg minden pontja átlátszatlan lesz.
4. Válasszuk ki a rétegmaszkot bal gomb segítségével a rétegkezelőben.
5. Jelöljük körbe szabadkézi kijelölési eszköz segítségével az embereket! Fontos tudni, hogy a kijelölést a *pár* réteghez tartozó rétegmaszkon végezzük a 4. lépés miatt!
6. Invertáljuk a kijelölést a CTRL+I billentyűkombináció segítségével!
7. Töltsük ki a kijelölést fekete színnel ( ), vagy alkalmazzunk színátmenetes kitöltést feketéből fehérbe ( ), hogy beállítsuk a réteg átlátszó területeit!
8. Mentsük a munkánkat `szerelmesek` néven a szoftver alapértelmezett formátumában, illetve exportáljuk PNG formátumban ugyanezen a néven!



► Fehér rétegmaszk hozzáadása



► Rétegmaszk a rétegkezelőben (az ábrán pirossal bekeretezve)



► Invertált kijelölés kitöltése a rétegmaszkon fekete színnel, vagy színátmenet alkalmazása

Háttér eltávolítása, szürkeárnyalat (telítettség), alfa-csatorna

Az előző példában a háttér módosításához rétegmaszkot használtunk. Ennek segítségével a réteg bizonyos területeit átlátszóvá tudtuk tenni. (Emlékezzünk vissza: a kettőslátást demonstráló példánál a réteg minden pontjának átlátszatlanságát egységesen kezeltük.) Harmadik példánkban egy autót szeretnénk az utcai környezetből úgy kiemelni, hogy hátterét megtartjuk, de azt szürkeárnyalatossá tesszük. A megoldásához most alfa-csatornát fogunk használni.

3. példa: **Kiemelés**

1. Nyissuk meg az `auto.jpg` képet! Itt nagyon fontos megjegyezni, hogy a JPG-formátumú képek egyáltalán nem alkalmasak az alfa-csatorna kezelésére.
2. A rétegkezelőben kattintsunk jobb gombbal a rétegen, és kétszerezzük meg! Ez sokkal gyorsabb, mint az első feladatnál bemutatott másolás, beillesztés, új réteggé alakítás műveletsorozat.
3. Nevezzük el a rétegeket jobb gomb segítségével! A felső réteg neve legyen *autó*, az alsó réteg neve legyen *utca*! Az *utca* réteg megjelenítését kapcsoljuk ki a réteg neve mellett található szemet ábrázoló ikon segítségével.
4. Válasszuk ki az autó réteget, és a lasszó segítségével (szabadkézi kijelölési eszköz) jelöljük ki a körvonalát!
5. Fordítsuk meg a kijelölést (invertálás), és töröljük az utcaképet! Azt fogjuk tapasztalni, hogy a törlés helyén a képünk háttérszínű lett. (Ha az alapbeállításon nem módosítottunk, akkor ez fehér.) Ennek az az oka, hogy nem adtunk alfa-csatornát a képhez, és mint korábban írtuk, a forrásként használt JPG-képformátum ezt nem támogatja!
6. Vonjuk vissza Ctrl + Z segítségével a törlés parancsot, és adjunk alfa-csatornát az autó réteghez jobb gomb segítségével a rétegkezelőben! Próbálkozzunk ismét a törléssel! Az *autó* rétegen a teljes háttér átlátszó – **átlátszatlansága 0, az ilyen területeket szürkeárnyalatos pepita megjelenítéssel mutatják a szoftverek** –, míg az autó átlátszatlan! Fogalmazhatunk úgy is, hogy a háttér esetében a képpontok „eltűntek”. **Jó tudni, hogy a 0 átlátszatlanságtól eltérő összes pont egyszerűen kijelölhető az alfa-csatorna kijelölése lehetőséggel.**
7. Szüntessük meg a kijelölést!
8. Válasszuk ki az utca réteget, jelenítsük meg, és változtassuk a réteget szürkeárnyalatossá (Színek / Telítetlenné tevés)!
9. Mentsük a munkánkat `utca` néven a szoftver alapértelmezett formátumában, illetve exportáljuk PNG-formátumban ugyanezen a néven!

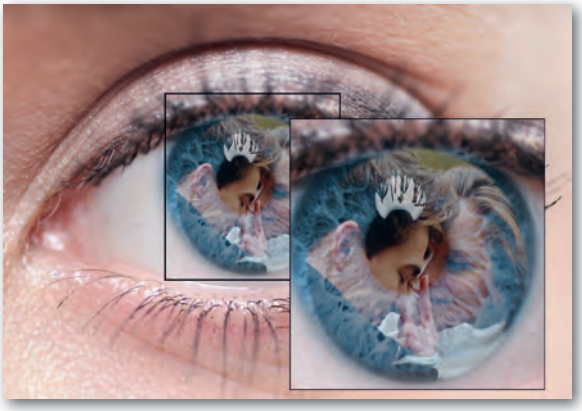
Ezt a feladatot rétegkétszerezés nélkül is megoldhattuk volna egyszerűen úgy, hogy megnyitás után egyből kijelöljük az autót szabadkézi kijelölő eszköz segítségével, majd a kijelölés megfordítása után szürkeárnyalatossá tesszük az autón kívüli területet. Ám ez a módszer nem teszi lehetővé a háttér lecserélését. Ezek után, ha más hátteret szeretnénk az autónk mögé, könnyen megtehetjük, ha az *utca* réteg megjelenítését kikapcsoljuk, és helyette másik képet teszünk be új rétegként.

► Színek telítettségének módosítása a rétegen

Lágy perem, alfa-csatorna

A kettős látást demonstráló feladatnál nagyon zavaró volt az éles határvonal, ami két réteg találkozásánál keletkezett. Ezt az éles határvonalat lágy perem segítségével el tudjuk tüntetni. Készítsünk el egy olyan képet, amely szemben tükröződő szerelmespárt ábrázol!

4. példa: Tükröződés a szemben



► A lágy perem alkalmazása eltünteti az éles határvonalakat

1. Töltsük le munkánkhoz a `szem.jpg` és `par.jpg` képeket!
2. Nyissuk meg képszerkesztő programmal a `szem.jpg` képet, és nevezzük el a réteget *szem*-nek! Ez a réteg lesz legalul, ezért ehhez nem adunk alfa-csatornát. Megjelenítését a szem ikonra kattintva kapcsoljuk ki a rétegkezelőben!
3. Nyissuk meg új rétegként a `par.jpg` képet a Fájl menü segítségével, és nevezzük el a réteget *pár*-nak! Ez a réteg legyen a legfelső!
4. Adjunk a *pár* réteghez alfa-csatornát!
5. Válasszuk az ellipszis kijelölőt, és a kijelöléshez adjunk lágy peremet!
6. Jelöljük ki a szerelmesek fejét a *pár* rétegen úgy, hogy maradjon egy kis felesleges háttér az átmenetnek!

7. Fordítsuk meg a kijelölést, és töröljük a hátteret, majd szüntessük meg a kijelölést!
8. A *pár* réteg átlátszatlanságát állítsuk 55%-ra!
9. Jelenítsük meg a *szem* réteget!

A feladattal már majdnem készen vagyunk, de a szerelmespár nem fér bele a szembe, és rossz helyen is van. Tehát át kell méretezni, és mozgatni is kell.

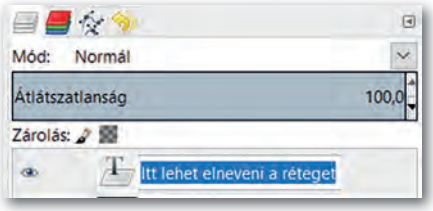
10. Jelöljük ki a *pár* réteget, és az átméretező eszköz segítségével módosítsuk a méretarányok megtartása mellett a réteg méretet úgy, hogy elférjen a szemben (a szivárványhártya által kialakított kör alakú területen)!
11. A *pár* réteget mozgassuk a helyére az áthelyezési eszköz segítségével!
12. Mentsük a munkánkat `tukrozodes` néven a szoftver alapértelmezett formátumában, valamint exportáljuk PNG-formátumban!

Kérdések, feladatok

1. Milyen rétegekkel kapcsolatos műveleteket ismerünk? Hogyan módosíthatjuk a rétegek sorrendjét? Ismételjük át az általunk használt képszerkesztő program segítségével!
2. Mire való az alfa-csatorna? Mi történik akkor, ha egy kép nem rendelkezik alfa-csatornával, és azon radírozni kezdünk? Milyen formátumban kell elmenteni egy képet, ha az alfa-csatorna adatait is tárolni szeretnénk?
3. Hogyan jelölik a szoftverek az alfa-csatornával rendelkező rétegeken a „képpont nélküli” területeket (az olyan képpontokat, melyek átlátszatlansága 0%)? Milyen módon lehet kijelölni az összes olyan pontot egy rétegen, melyeknek átlátszatlansága 0-tól különbözik?
4. Keressünk az interneten speciális képkereső segítségével olyan fekete-fehér (szürkeárnyalatos) képet az 1956. október 23-ai forradalomról, amelyen van zászló! Töltsük le a képet, és módosítsuk úgy pixelgrafikus képszerkesztő segítségével, hogy a zászló kiemelve, színesen jelenjen meg! Ügyeljünk arra, hogy az anyag gyűrődése is megmaradjon valamiképp! Mentsük a munkánkat `okt23` néven a szoftver alapértelmezett formátumában!
5. Az `erdo.jpg` és `szemet.jpg` képeket felhasználva készítsünk el két 800 × 800 px méretű képet a minta alapján! Elvárás, hogy a kulcslyuk szimmetrikus legyen, és vízszintesen a kép közepén helyezkedjen el, mindkét képen ugyanott! A kulcslyuk vonalát lágy peremmel határoljuk! Mentsük a munkánkat `ajto1` és `ajto2` néven a szoftver alapértelmezett formátumában, és exportáljuk PNG-formátumban a neveket megtartva!

► Minta az 5. feladat megoldásához

Szöveg a képen



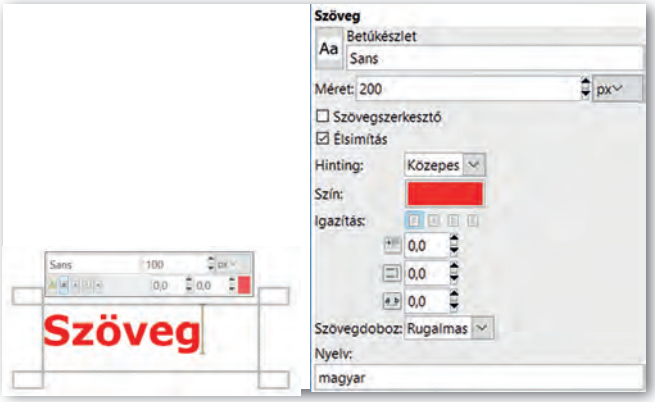
► Szöveg típusú réteg a rétegkezelőben

A pixelgrafikus képszerkesztő programokban lehetőségünk van **szöveg elhelyezésére** is. A későbbi könnyebb szerkesztés érdekében a szöveg jellemzően egy új, **szöveg típusú rétegen** jön létre, amelyre ikon utal a rétegkezelőben. Formázási lehetőségeink korlátozottabbak, mint egy szövegszerkesztőben, és szoftverenként is eltérő. Általában a félkövér, dőlt, aláhúzott, áthúzott formázások, a betűtípus és -méret módosítása mindenütt használható, de vannak olyan szerkesztők, ahol a betűtávolság, sorköz, első sor behúzása, alsó-, felső index stb. is elérhető. Több lehetőségünk van viszont a szöveget valamilyen **útvonal** mentén elhelyezni, vagy **görbe alapján torzítani, szűrők** segítségével grafikusan módosítani. Így egyedi és figyelemfelkeltő feliratokat tudunk készíteni.

Nagyon fontos megjegyezni, hogy **a szöveg tartalmát átírni csak addig tudjuk szövegeszköz segítségével, amíg külön szöveg típusú rétegen megtalálható.** Amennyiben a szövegréteget egy másik réteggel egyesítjük, akkor már csak képpontonként szerkeszthetjük.

5. példa: Szöveg elhelyezése, tulajdonságok állítása

1. Hozzunk létre egy új képet!
2. Válasszuk az eszközök közül a szövegeszközt, aminek az ikonjában általában egy betű van (jellemzően A vagy T).
3. A kép területén kattintva helyezzük el a feliratunk dobozát! Egyes szoftverekben be lehet állítani, hogy a doboznak fix mérete legyen, vagy a tartalomnak megfelelően változzon.
4. A beírt szöveget vagy eszköztárról, vagy felugró ablakról formázhatjuk (pl. félkövér, betűszín, alapvonal állítása, betűköz, szöveg igazítása stb.).

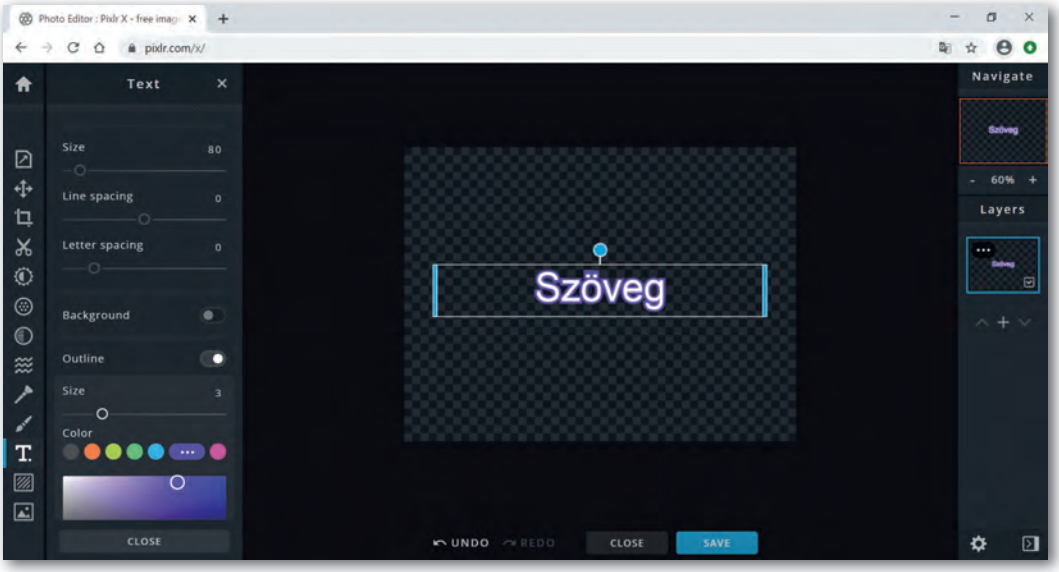


► Szöveg formázása felugró ablakból vagy eszköztárból

A szövegrészlet kijelölése történhet az egér bal gombjával. Egy szót dupla kattintással, a teljes szöveget tripla kattintással tudunk egyszerűen kijelölni.

Szöveg körvonalazása

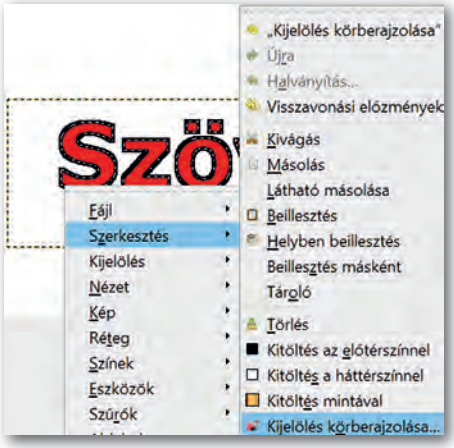
Vannak szerkesztőprogramok, ahol egy kattintással lehet a szövegeszközök közül választva körvonalat (outline) adni a betűknek. Ahol erre nincs lehetőségünk, ott ki kell jelölni a szöveg képpontjait, és a kijelölést körberajzolni, mint ahogy azt az alakzatok létrehozásánál tettük!



▶ Egy online képszerkesztő felülete, ahol a betű körvonalazása könnyen elvégezhető (PIXLR)

6. példa: Szöveg képpontjainak körberajzolása

1. Hozzunk létre egy átlátszó háttérrel rendelkező új képet!
2. Helyezzünk el rajta szöveget, tetszőleges színnel!
3. Válasszuk az eszköztárról a szín szerinti kijelölést a szöveg típusú rétegen, és kattintsunk a szövegben egy képpontra!
4. Változtassuk meg az előtér színét, majd válasszuk menüből, – vagy jobb gomb segítségével a helyi menüből – a kijelölés körberajzolását!



▶ Kijelölés körberajzolása jobb gomb segítségével

Szöveg kitöltése képpel

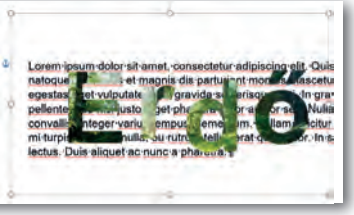
Nagyon látványos megoldás, amikor a betűk nem egyszínűek, hanem valamilyen mintázattal vagy képpel vannak kitöltve.

7. példa: Kép mintázatú szöveg

1. Nyissunk meg egy képet a szerkesztőprogrammal, amelyet a betűk kitöltésére fogunk használni! Nevezzük el a réteget *Háttér*-nek és adjunk hozzá alfa-csatornát!
2. Adjunk szöveget a képhez, és formázzuk meg (méret, stílus, típus stb.) Kísérletezzünk bátran! Módosítsuk az alapvonalat akár betűnként, és állítsuk át a betűk közti távolságot! Most a színnel nem kell foglalkoznunk, hiszen a kitöltést majd a Háttér réteg fogja adni! Nevezzük át a szöveget tartalmazó réteget *Felirat*-ra!
3. Jelöljük ki a *Felirat* rétegen a betűk képpontjait színük alapján!
4. Váltsunk át a *Háttér* rétegre, és fordítsuk meg (invertáljuk) a kijelölést!
5. Töröljük a *Háttér* réteg felesleges részeit!
6. Töröljük a *Felirat* réteget!
7. Mentsük a munkánkat `felirat` néven a szoftver alapértelmezett formátumában!

► Átlátszó háttér pepita megjelenítéssel a képszerkesztőben

► Megtekintés képnézegetővel

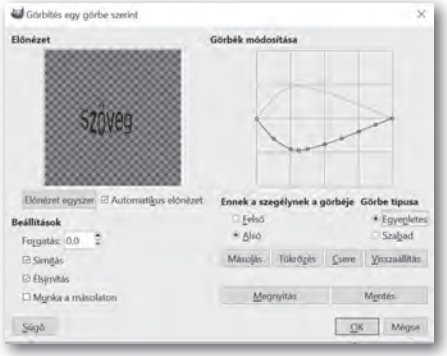


► Kép elhelyezése szöveg elé szövegszerkesztő programban, az átlátszóság érvényesülése

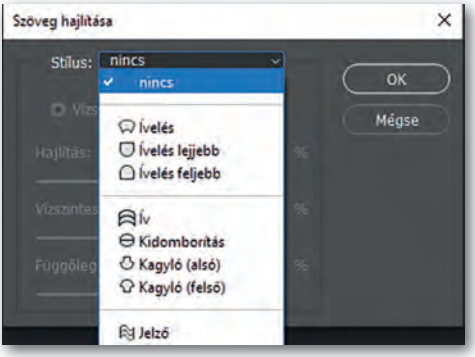
Szöveg torzítása

A szövegszerkesztő programok lehetőséget kínálnak a karakterek grafikus megjelenítésére, de a torzítást (hajlítást) csak néhány lehetőség közül választhatjuk ki, és a formázási lehetőségünk behatárolt. A pixelgrafikus képszerkesztő programok számos, gazdagon paraméterezhető lehetőséget biztosítanak a kijelölt terület (legyen akár szöveg vagy képrészlet) átalakításához, amiket többnyire a Szűrő (Filter) menün keresztül érünk el!

► Szöveg hajlítása szövegszerkesztő programban



► Szöveg görbítése Szűrő/Torzítás/Görbítés egy görbe szerint menün keresztül (GIMP)



► Szöveg hajlítása (Photoshop)

Szöveg elhelyezése útvonalon

Zárt útvonalat könnyen létre lehet hozni kijelölés alapján (szabadkézi, téglalap, ellipszis). Kattintsunk jobb gombbal a kijelölésen és válasszuk a *Kijelölés* menü *Útvonallá alakítás* parancsát!

Tetszőlegesen hajlított nyílt vagy zárt útvonalak létrehozásához útvonal eszközt is találunk az eszköztáron. Létrehozott útvonalainkat az útvonalkezelőben találjuk, melyeket a rétegekhez hasonlóan el is tudunk nevezni.

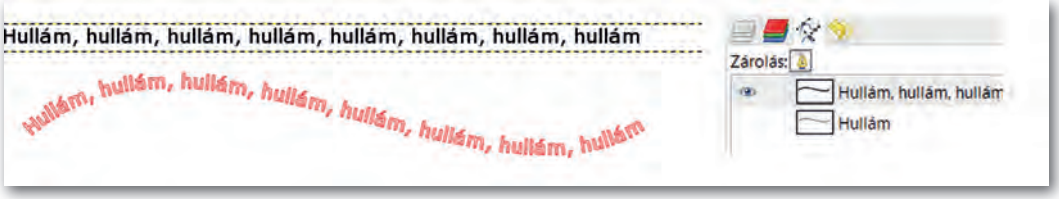
8. Példa: Készítsünk egy olyan szöveget, melynek alapvonala hullámzik!

(Ehhez hasonló feladattal a *Vektorgrafika* témánál is fogunk találkozni.)

1. Hozzunk létre szerkesztőprogram segítségével egy új képet fehér háttérrel! A réteg neve legyen *Háttér*!
2. Adjunk szöveget a képhez! Nevezzük el a szöveg réteget *Szöveg*-nek!
3. Az útvonal eszközt használva, az egér bal gombjával kattintva jelöljük ki a hullám két végpontját, majd a szakasz belső pontjait megfogva végezzük el a hajlításokat! A végpontokban található vezérlőpontokkal állíthatjuk be a görbületet.

▸ A hullámos útvonal: a végpontokban található vezérlőpontokat kis négyzet jelöli

4. Válasszuk ki az útvonalkezelőben a létrehozott útvonalat, és nevezzük el *Hullám*-nak!
5. Válasszuk ki a rétegkezelőben a *Szöveg* réteget, és jobb gomb segítségével illesszük útvonalra! Ezzel egy új útvonal jön létre, amit láthatunk is az útvonalkezelőben, amit a hullámos szöveg alkot.



▸ Az új útvonal pirossal látható

6. A szöveget tartalmazó útvonalat alakítsuk kijelöléssé jobb gomb segítségével, majd váltsunk át a rétegkezelőben a *Háttér* rétegre, és töltsük ki a kijelölést tetszőleges színnel! Kapcsoljuk ki a *Szöveg* réteg megjelenítését!
7. Szüntessük meg a kijelöléseket, és kapcsoljuk ki az útvonalkezelőben az útvonalak megjelenítését! Mentsük a munkánkat `felirat` néven a szoftver alapértelmezett formátumában!

Kérdések, feladatok

1. Az úszás és néptánc oktatását már nagyon sok helyen óvodákban elkezdik, sőt van, ahol játékos módon már idegen nyelveket is tanítanak. A korosztály érdeklődési körét, játékosságát figyelembe véve tervezzünk figyelemfelkeltő plakátokat a három témában, alkalmazva az eddig tanultakat! Dolgozzunk háromfős csapatokban, gondoljuk végig, milyen szöveges információra van szükségük a szülőknek, és az óvodásokra milyen vizuális látvány van hatással! Mindenki egy témát dolgozzon fel, de ügyeljünk arra, hogy a három plakát valamilyen módon egységes legyen! Mutassuk be a plakátokat, ismertessük elképzelésünket!
2. Hozzunk létre egy 1600 × 800 pixel méretű képet, amelyre elhelyezzük az előző lecke 5. feladatában elmentett két képet! Készítsünk feliratot a képre a következő szöveggel: „Az ajtó melyik oldalán szeretnél lenni?” Alkalmazzuk a tanultakat (szöveg útvonalra illesztése, a betűk körvonalazása, betűk hajlítása stb.) Ügyeljünk a szöveg olvashatóságára!

► Minta a 2. feladat megoldásához

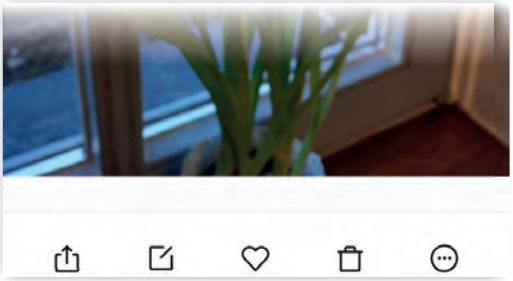
Képszerkesztés mobiltelefonnal

Mobilalkalmazások

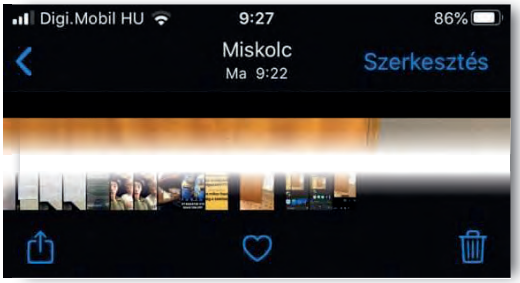
Sokszor nincs időnk arra, hogy a mobiltelefonunkkal készített képeket az asztali számítógépen módosítsuk. Ilyenkor a képek formázására használhatjuk a mobiltelefon **operációs rendszerébe integrált képszerkesztőt**, vagy letölthetünk alkalmazásokat, amelyek többnyire egy-egy speciális feladat megoldására készültek. Ilyen lehet a keretezés, kollázs készítése, képek keverése (photo blender). Akadnak egészen nagy tudású, több réteg kezelését lehetővé tevő alkalmazások, amelyek használatáért már fizetni kell.

Operációs rendszerbe integrált képszerkesztők

Az eddig megismert módosítások közül a vágás, szűrők, telítettség állítása, szöveg elhelyezése képre könnyen elvégezhető mobiltelefonon is. Válasszunk a képgalériában egy képet! A képek szerkesztését vagy ikonra, vagy menüre kattintva tudjuk elkezdeni operációs rendszertől függően!



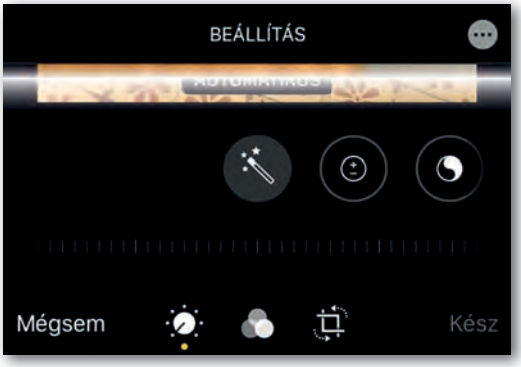
► Képek szerkesztése az ikonra kattintva (Android)



► Képek szerkesztése menün keresztül (iOS)

A szerkesztési lehetőségek változatosak, az ikonok hasonlóak az asztali gépeken használatos szoftverekéhez. Egyszerű képszerkesztési műveleteket az azonnali üzenetküldő szolgáltatást nyújtó alkalmazások is tudnak (Messenger, Skype, WhatsApp stb.).

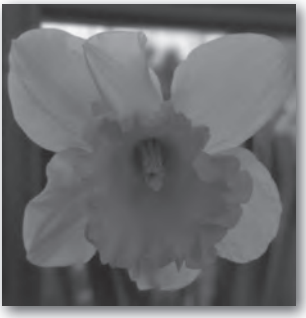
► Szerkesztési lehetőségek (Android)



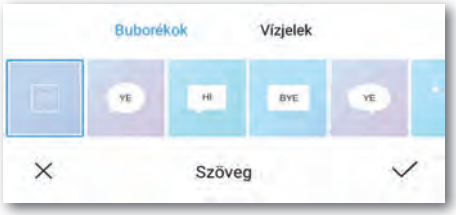
► Szerkesztési lehetőségek (iOS)

9. példa: A telítetlenség állítása és a szöveg elhelyezése képen

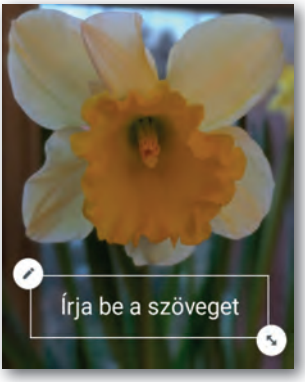
► Telítetlenség beállítása 0-ra



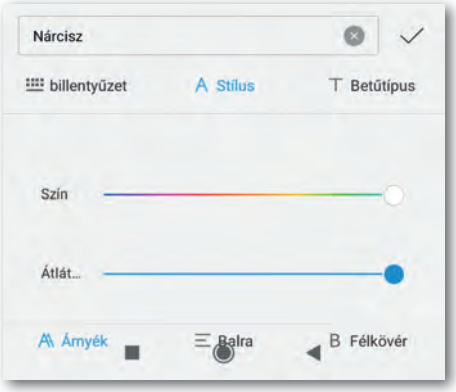
► A művelet eredménye



► Szöveg elhelyezése



► A művelet eredménye



► Szöveg formázása

► A művelet eredménye

Rétegek kezelése mobiltelefonon

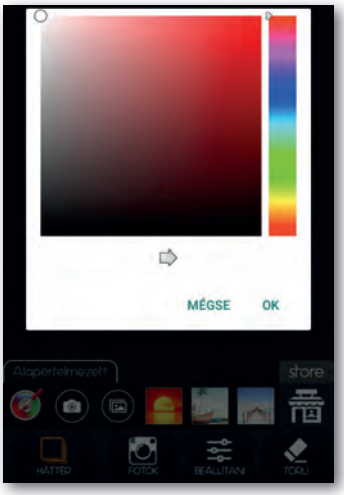
A mobilalkalmazásokra jellemző, hogy kevesebb vagy kevésbé testre szabható funkciót kínálnak a felhasználók számára, de azokat jóval gyorsabban és egyszerűbben érjük el. A képszerkesztés, különösen nagy felbontású képek esetén kifejezetten erőforrás-igényes feladat, ilyen esetekben célszerűbb az asztali alkalmazásokat választani. Meg kell jegyezni, vannak nagy teljesítményű, kiváló kamerával rendelkező telefonok, amelyek ilyen jellegű

feladattal is megbirkóznak, de a megjelenítő mérete miatt kényelmesebb az asztali verziót választani, nem is beszélve arról, hogy jóval könnyebben és pontosabban tudunk pozicionálni, kijelölni.

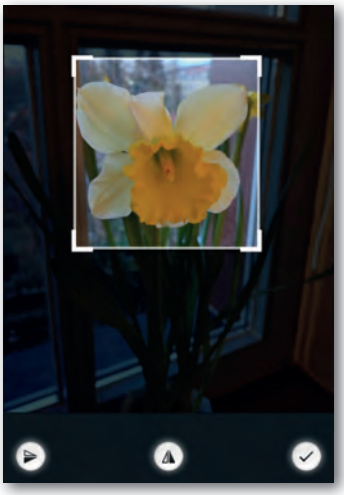
A munka teljesen hasonlóan folyik mobiltelefonon, mintha asztali gépen dolgoznánk.

1. Háttér kijelölése (szín vagy háttérkép állítása)
2. Rétegek hozzáadása, méretezése, vágása, forgatása, átlátszatlanság beállítása
3. Szűrők, keretek, szöveg hozzáadása

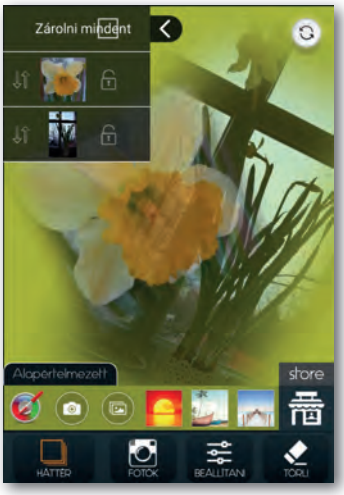
10. példa: Munka rétegek kezelésére alkalmas applikációval (Ultimate Photo Mixer)



▶ Háttér hozzáadása

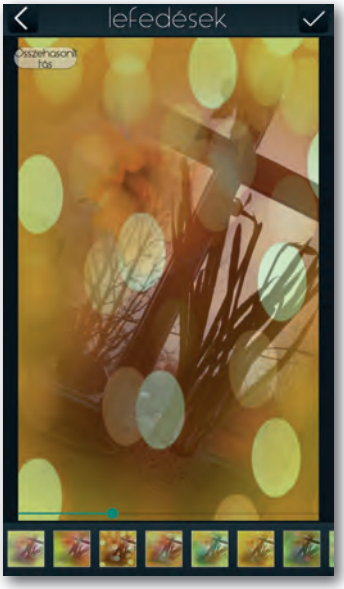


▶ Réteg beszúrása, vágása, mozgatása, forgatása



▶ Műveletek rétegekkel, balra fent a rétegkezelő

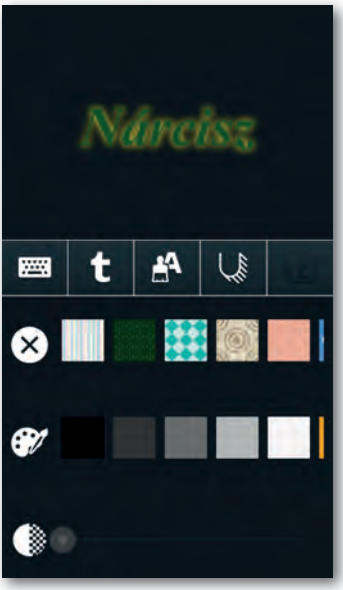
▶ Kép vágása



▶ Lefedések, matricák

▶ Keret hozzáadása

► Beépített szűrők



► Szöveg beszúrása, formázása

► Szöveg elhelyezése

Kérdések, feladatok

1. Sokszor előfordul, hogy mobiltelefonnal fényképezve utólag jövünk rá: képünkön felesleges képrészletek vannak. Milyen lehetőségeink vannak ezek eltüntetésére mobilalkalmazások segítségével? Milyen lehetőségek és eszközök vannak, ha nem kívánt részlet a háttérben van, illetve abban az esetben, ha a szélén helyezkedik el?
2. Keressünk a telefon galériájában vagy az interneten álló alakos ember képét! Készítsünk belőle egy szürkeárnyalatos igazolványképet, ami csak a fejet ábrázolja!
3. Nézzünk szét az interneten, milyen képszerkesztő applikációk vannak mobiltelefonra! Próbáljuk kategorizálni azokat több szempont alapján!
4. Készítsünk szülinapra invitáló képet mobiltelefonnal! Alkalmazzunk ehhez keretet, matricákat, szöveget! Mentsük a képet a mobiltelefonunkra!
5. Nézzünk utána, hogyan lehet képernyőképet készíteni a mobiltelefonunkkal! Keressünk rá lakcímünkre műholdas vagy utcakép nézetben egy online térképen! Készítsünk képernyőképet a találatról, majd rajzoljunk nyilat, ami az otthonunkra mutat, és készítsünk rá feliratot: „Ide kell jönni!”! Mentsük a képet a mobiltelefonunkra!

Elméleti fogalmak, adatvédelem

GDPR – General Data Protection Regulation

A **GDPR**, magyarul általános adatvédelmi rendelet (a hivatalos elnevezése kicsit hosszabb) azoknak a természetes személyeknek a személyes adatait védi, akik az Európai Unió területén tartózkodnak.

Az adatvédelem mindig is fontos volt, törvényi szabályozások korábban is voltak, ám az információs technológiai fejlődés új kihívások elé állították azokat. Ennek hatására 2016. május 24-én lépett hatályba a GDPR, és 2018. május 25-től kell alkalmazni. Miért fontos ez számunkra?

▶ GDPR – 2018. május 25-től kötelező alkalmazni

„A **személyes adat** minden olyan információ, amely valamely azonosított vagy azonosítható élő személlyel kapcsolatos. Mindazon információk, amelyek összegyűjtése egy bizonyos személy azonosításához vezethet, ugyancsak személyes adatnak minősülnek” – olvasható az Európai Unió hivatalos portáljához tartozó ec.europa.eu weboldalon.

▶ Az arckép, ujjlenyomat, hang mind személyes adatnak minősül

Ezek után teljesen érthető, hogy amikor fényképezünk vagy videót készítünk, nagyon körültekintőnek kell lennünk, hiszen az emberi képmásunk, hangunk – személyiségünk fontos része – személyes adatnak minősül, a felvételekhez kapcsolódó személy hozzájárulása nélkül nem hozható nyilvánosságra, világhálón nem publikálható! A hozzájárulásnak több módja van: szóban, írásban, illetve ráutaló magatartással (például a kamerába mosolygunk).

Hogyan kezdjünk neki?

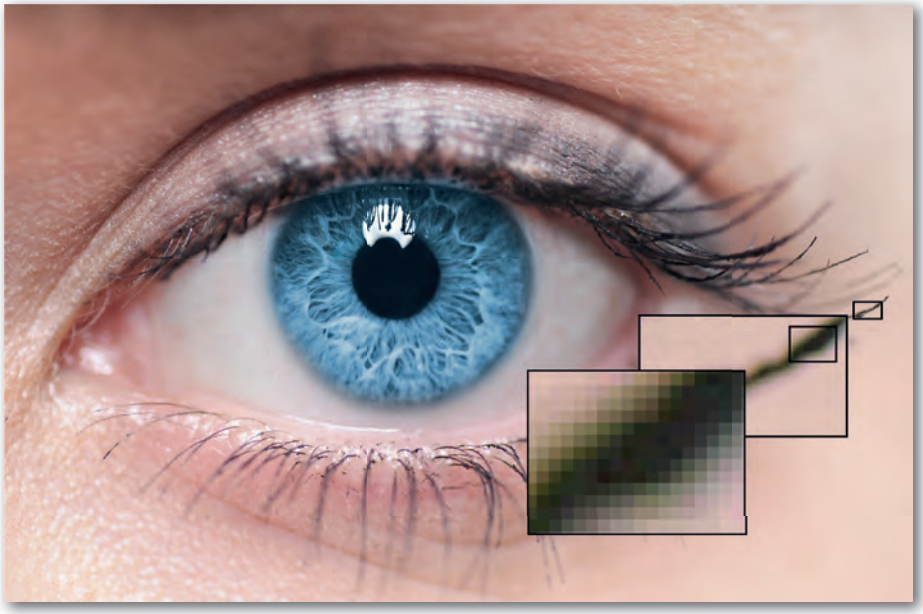
Egy jó videót elkészíteni nem egyszerű dolog. Gondoljunk bele, hogy egy film elkészítésénél hány szakma képviselője dolgozik együtt. Ha nem is leszünk filmesek, van néhány alapszabály és fogalom, amelyeket célszerű ismernünk.

Függetlenül attól, hogy mobiltelefonnal, fényképezőgéppel vagy kamerával filmezünk, törekedni kell arra, hogy a készülék ne mozogjon – ha van lehetőségünk, használjunk állványt. Mobiltelefonnal kerüljük az álló tájolású videók készítését, ugyanis a megjelenítők fekvőnek felelnek meg, és a megtekintéskor nagyon zavaró a széleken maradt üres terület. Események filmezésénél több szemszögből készítsünk felvételeket, a különböző részesemények felvételeit majd a vágóprogram segítségével tudjuk egyesíteni.

Digitális vagy optikai zoom (nagyítás)

Nagyon fontos tudnunk, hogy az **optikai zoom** növelése esetén a fényérzékelőre (CCD szenzor: Charge-coupled Device, azaz töltéscsatolt eszköz) a teljes **látótérnek csak kisebb része vetül**. A 36 megapixeles kamera esetén a kisebb területet ábrázoló kép ugyanúgy 36 millió képpontból fog állni, ezáltal jóval részletgazdagabb lesz a megjelenés.

Digitális zoom esetében a látótér nem módosul, hanem annak egy része lesz digitálisan felnagyítva. A nagyítás során **a kép minősége romlik**. A képpontok mérete „megnő”, ezért jelentősebb digitális zoom esetén a pixelek már nagyobb kiterjedésű négyzetekké alakulnak, ezt nevezzük **pixelesedésnek**. A fényképezőgépek ezt a jelenséget szoftveres eljárással próbálják csökkenteni, de a hatás nem küszöbölhető ki. Kerüljük, és csak elkerülhetetlen esetben használjunk digitális zoomot!



► Digitális nagyítás – pixelesedés

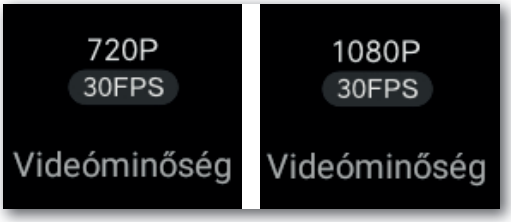
Képkockaszám (Frame Rate)

Az **FPS** (Frame Per Second) azt mutatja meg, hogy másodpercenként hány képkockát rögzítünk. Az emberi agy a **24 FPS**-sel rögzített videót már **folyamatosnak** érzékeli. Ha egy ilyen felvételt lassítani kezdünk, akkor hasonló dolog fog történni, mint a digitális zoom esetében. Míg a képeknél pixelesedést tapasztaltunk, a videóknál szaggatni kezd a kép. Gondoljunk csak arra, hogy amikor egy mozgást szeretnénk folyamatában látni – természetfilmekben sokszor találkozhatunk ilyen felvételekkel –, akkor a 24 FPS nem lesz elég. Ezekben az esetekben muszáj magasabb képkockaszámmal dolgozni (pl. 60 FPS).

Felbontás és FPS

A videó háttértáron elfoglalt méretét egyrészt meghatározza, hogy hány képkockát mentünk el másodpercenként, azaz az **FPS,** másrészt, hogy a képnek mekkora a **felbontása**. Ezek ma már jellemzően

- **HD Ready**, ami 1280 × **720** pixelt jelent,
- **Full HD,** ami 1920 × **1080** pixelt jelent,
- **4K**, ami 3840 × **2160** pixelnek felel meg.



► Felbontás és FPS beállítása mobiltelefonon

A fenti felbontások **16 : 9-es képarányt** eredményeznek, ezek a legelterjedtebbek az interneten. Telefonon a felbontás kiválasztásánál jellemzően a sorok számát adják csak meg.

Tömörítés, fájlformátumok

Azt már láttuk, hogy képek esetén milyen drasztikusan csökken az állományok mérete, ha valamilyen tömörítést alkalmazó fájlformátumot használunk. Videók esetében is így van ez. Szinte mindegyik formátum **veszteséges tömörítési eljárást** alkalmaz. A tömörítéseknél kihasználható, hogy a mozgókép egymás utáni képek sorozata, ahol az egymás utáni képek között általában kicsi az eltérés, így elég a változásokat tárolni.

Jellemző fájlformátumok videók tárolására az **AVI, MPEG, MP4, MOV, 3GP.**

Kérdések, feladatok

1. Nézzünk utána az interneten a *ráutaló magatartás* fogalmának!
2. Soroljunk fel *személyes adatokat*, amelyeket a telefonunkon tárolunk!
3. Milyen minőségben tud a telefonunk videót rögzíteni? Nézzünk utána!
4. Nézzünk utána a videómegosztó portálokon, mi az az *FPS*! Keressünk olyan videót, ami szemlélteti, mi történik, ha a képkockaszámot növeljük!
5. Keressünk videómegosztó portálon 1920-as évekből származó Charlie Chaplin-némafilmet! A képkockaszám *24 FPS* alatt vagy felett van? Válaszunkat indokoljuk!

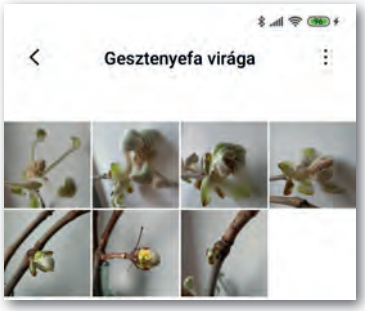
Videók készítése és szerkesztése mobiltelefonnal

Alkalmazások

Videók szerkesztését könnyen elvégezhetjük asztali vagy online alkalmazás segítségével, de akár mobiltelefonon is. A mobil- és online alkalmazások esetében kevesebb eszköz áll a rendelkezésünkre, de azok is elegendőek egy gyors, egyszerű, esztétikus munka elkészítéséhez. Az asztali alkalmazások között találunk fizetős és profi munka kivitelezésére alkalmas szoftvereket.

► Műveletek a Windows 10 videószerkesztőjében

A képszerkesztésnél már volt szó arról, hogy mennyire gazdag a fényképszerkesztéssel kapcsolatos mobilapplikációk választéka az interneten. Ugyanez a videószerkesztéssel kapcsolatban is elmondható. Sőt vannak olyanok, amelyekkel egyszerre lehet videót, fényképet, vagy akár kollázst is készíteni, szerkeszteni. Ilyen applikáció például a magyar nyelven is elérhető InShOt.



► Képek tárolása albumban

Videó szerkesztése mobiltelefonnal

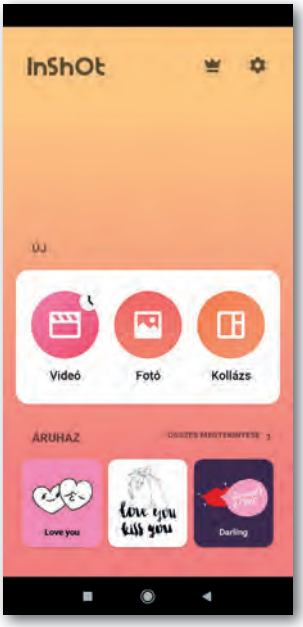
Az egyik legfontosabb, hogy a munkánkhoz használt képeket, videókat, hanganyagokat rendszerezetten tároljuk a telefonunkon. A következő lépés, hogy töltsünk le egy applikációt, ami alkalmas videók szerkesztésére. Az internetről letöltött fényképek és hanganyagok esetében ügyeljünk a szerzői jogokra!

A következő lépés, hogy megtervezzük a munkánkat. Amennyiben videót dolgozunk fel, a vágással célszerű kezdeni. Ez igen időigényes feladat, mert már a vágás előtt tudnunk kell, mire szeretnénk a hangsúlyt helyezni. Ehhez ismerni kell a teljes videóanyagot. Munkánk közben további videókat, képeket szúrhatunk be. Utána jöhet a háttérzene és/vagy hanganyag hozzáadása, illetve ezek megvágása. Szükség esetén szöveget is elhelyezhetünk, amihez animációt lehet rendelni. Az összetevők közül asztali alkalmazásoknál lehetőségünk van egyszerre többet is megjeleníteni, mivel a kijelző mérete ezt megengedi. Mobilapplikációk esetén a különböző összetevőket megjelenítő sávok csak akkor látszanak, ha az összetevő beszúrására koppintunk. A szerkesztési lehetőségeket hasonló módon érhetjük el.

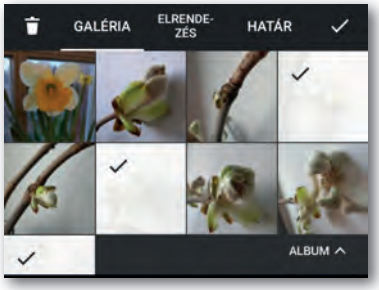
1. példa: Gesztenyefa virága

Ebben a példában egy gesztenyefa virágjáról készült képsorozatból fogunk videót készíteni, majd az elkészített videót megvágjuk.

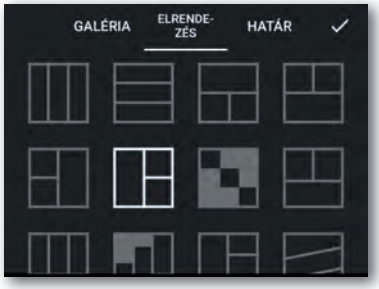
Kollázskép létrehozása



► Az InShOt alkalmazás kezdőképernyője



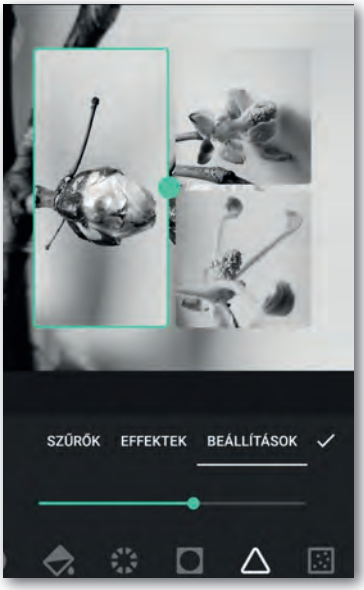
► Képek tárolása albumban



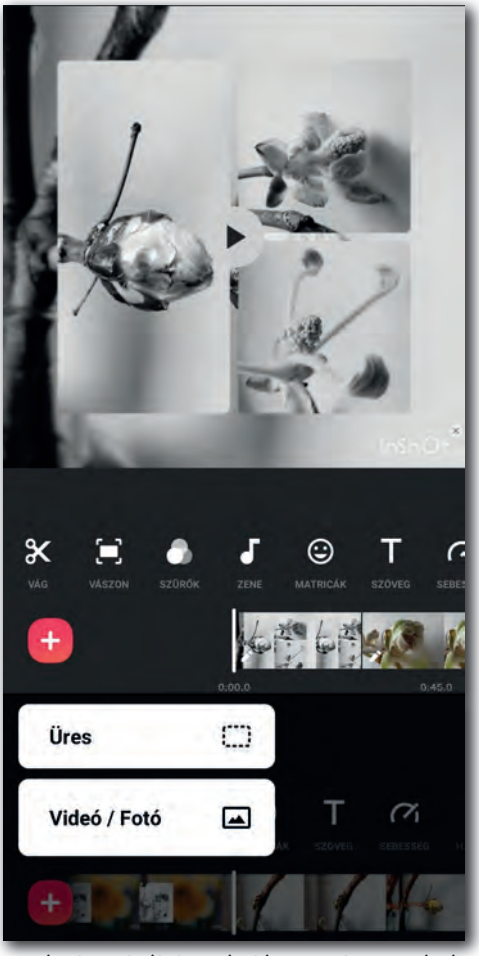
► Kollázs elrendezésének beállítása

► Kollázs határvonalának részletes beállítása

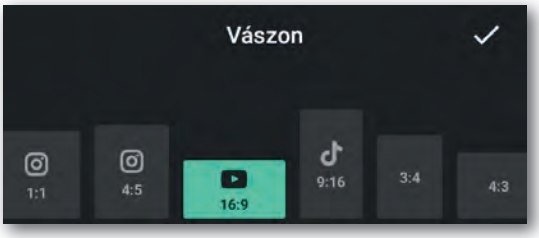
► Háttér beállítása (kép vagy szín)



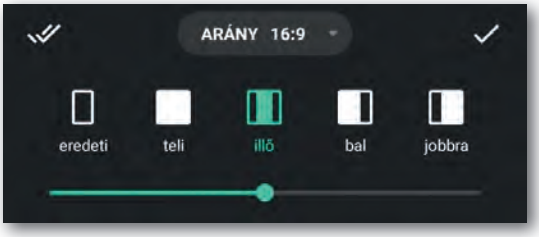
► Szűrők: telítettség és élesítés beállítása



► Képek, videók hozzáadása a + ikon segítségével



► A leggyakrabban használt képarány 16 : 9

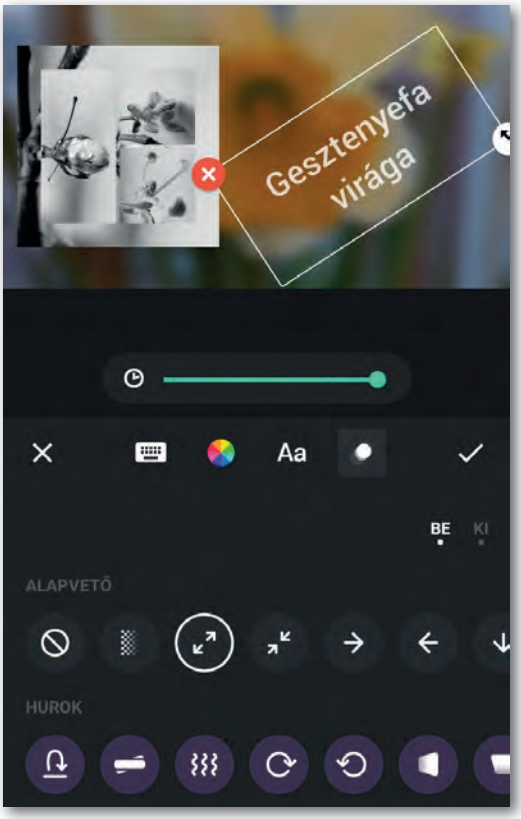


► A videó elhelyezése a vásznon

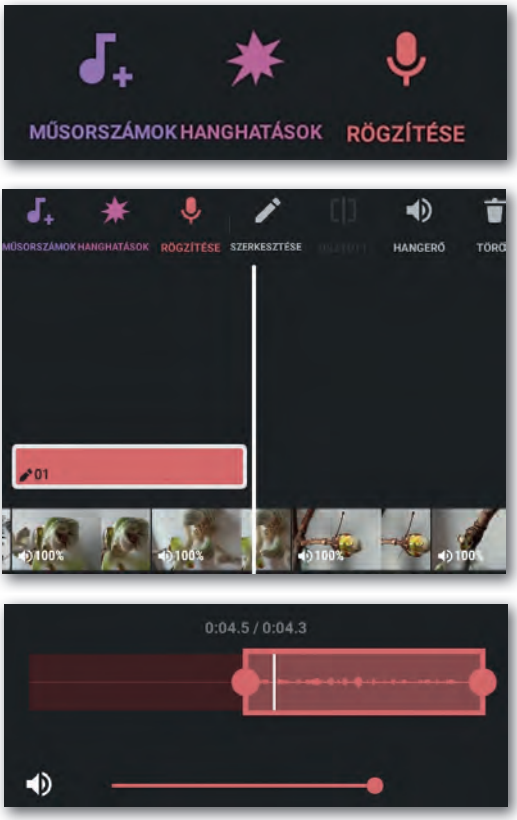
► Üresen maradt területek kitöltése képpel vagy színnel

A következő lépésekben szöveggel és hanganyaggal bővítjük a videót. A képszerkesztésnél tanultakat alkalmazzuk a szöveg beszúrásánál. Az animáció alkalmazásával bánjunk körültekintően, ne zavarja az olvashatóságot! A videóhoz hozzáadott háttérzene, hanghatás vagy hangrögzítés hanganyagát a videó képsávjához hasonlóan tudjuk vágni, illetve a hangerejét be tudjuk állítani. Nagyon fontos, hogy a hang, háttérzene dinamikája és a képi megjelenítés között összhang legyen. Jó, ha publikálás előtt megtekintik mások is a videót, mert így kaphatunk visszajelzést arról, elérte-e célját a kisfilmünk. Videók készítésénél és publikálásánál tartsuk szem előtt a személyiségi és szerzői jogokat!

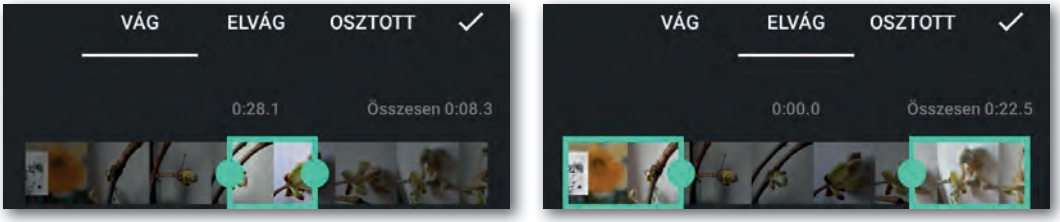
Felirat és hanganyag hozzáadása a videóhoz, vágás



► Szöveg elhelyezése és vágása



► Hangfelvétel hozzáadása és vágása

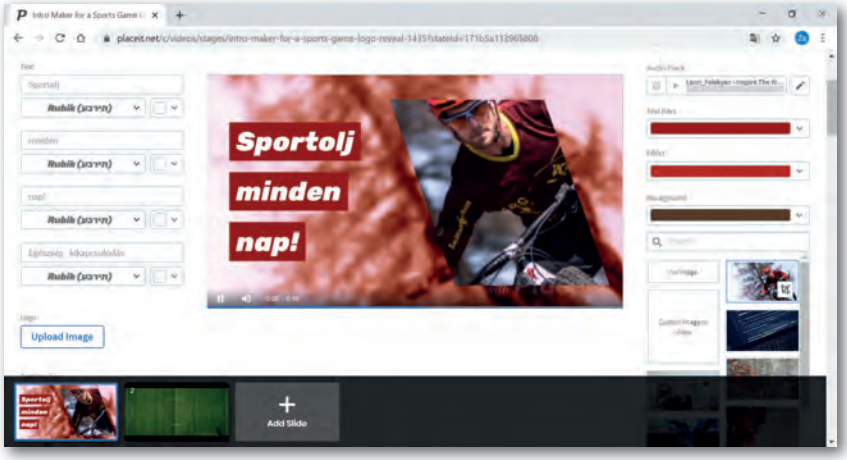


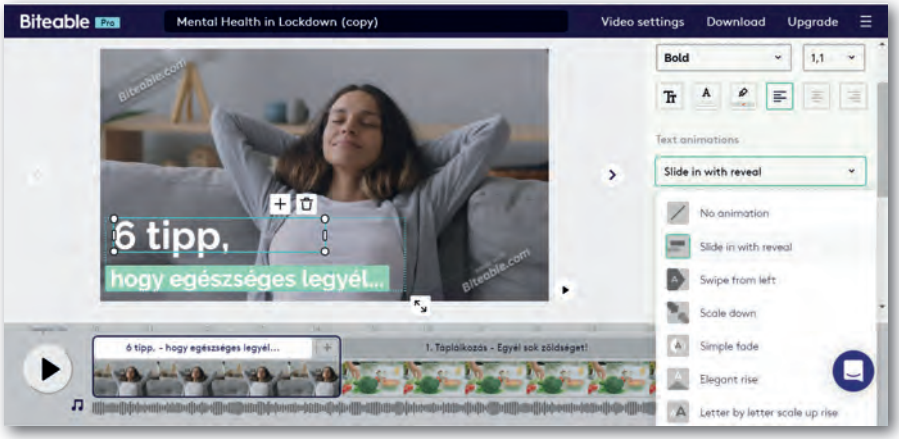
► Videó vágása (kivágás és szétvágás)

Kérdések, feladatok

1. Nézzünk utána, mit jelent a Time-lapse fogalom!
2. Milyen videószerkesztési feladatokra találunk online és mobilalkalmazásokat az interneten?
3. Alakítsunk háromfős csoportokat érdeklődési körünk, hobbink alapján, és készítsünk maximum 60 másodperces bemutatóvideót! A videóhoz felhasznált segédanyagokat (képek, videók, hanganyag) rendszerezve gyűjtsük úgy, hogy a csoport tagjai bármikor hozzá tudjanak férni. Az összegyűjtött anyagok alapján közösen hozzuk létre a videót, vágjuk meg, adjunk hozzá háttérzenét, szöveget! Nézzük meg egymás munkáit!

4. Amikor interneten böngészünk, gyakran találkozhatunk 10-20 másodperces reklámokkal. Ezeknek a videóknak rövidségük miatt az első pillanattól kezdve figyelemfelkeltőnek kell lenniük. Egy reklám elkészítése sok időbe kerül, azon kívül a professzionális videószerkesztő szoftverek ára sem olcsó. Erre nyújtanak áthidaló megoldásokat az ingyenes videósablonok (free video templates), melyek gazdagon paraméterezhetőek és módosíthatók, így könnyen igényeink szerint átalakíthatjuk. Sok esetben logószerkesztésre is van lehetőségünk. Alakítsunk 3-4 fős csoportokat és készítsünk egy figyelemfelkeltő, rövid videót, amiben az egészséges életmódra hívjuk fel a figyelmet (táplálkozás, sport, kirándulás, lelki egészség, stb…). Keressünk ingyenes online szerkeszthető videósablont az interneten és azt felhasználva oldjuk meg a feladatot! Egészítsük ki a videót saját felvételeinkkel! Tervezzünk logót is!





D Vektorgrafika

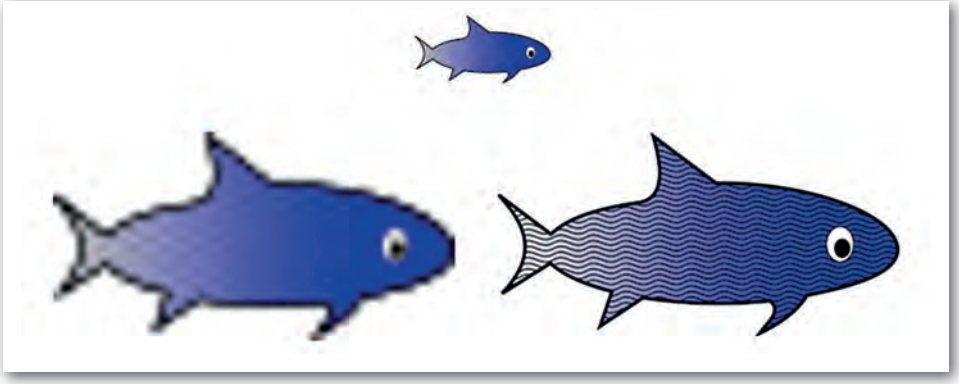
A vektorgrafika alapfogalmai és szerkesztőprogramjai

A vektorgrafika a számítógépes grafikának az a része, amelyben alakzatokat, geometriai elemeket használunk fel az ábra vagy kép elkészítéséhez, és nem egymás mellé, alá helyezett színes képpontokat. Az alakzatoknak általában vannak alaptulajdonságai, mint például méret, kitöltőszín, átlátszóság, szegély stb. Az ábrát éppen ezeknek a beállításával, módosításával hozzuk létre.

A vektorgrafikai programok az ábrák elkészítését alakzatok létrehozásával, tulajdonságainak módosításával, célszerű átalakításával és együttes felhasználásával teszik lehetővé.

A vektorgrafikus ábrák felhasználása széles körű, mert a környezetünkben a tájékozódást segítő információkat legtöbbször ezek segítségével készítik. Ilyen ábrák például az emblémák, logók, a pólók rajzai, a közlekedési és információs táblák, a különböző mobileszközök ikonjai, de még a karakterek megjelenítése is. Használják a tervező- és a mérnöki munkában, a tudomány különböző területein, ahol a vizualizáció fontos.

A vektorgrafikus ábrán az alakzatok (szakaszok, csillagok, sokszögek, ellipszisek, ívek, paraméteres görbék stb.) helyét koordinátákkal, alakját egyenletekkel adja meg és tárolja a szerkesztőprogram. A szerkesztési lehetőségek mellett fontos tudni, hogy a vektoros rajzolási módszer a tárolási formátumot is meghatározza. Megjelenítéskor a képet rasztergrafikus formátumba alakítja a képnézegető program, mert a megjelenítő eszközök, a monitorok, a nyomtatók pixelekből állítják elő a képet.



► A rasztergrafikus ábrák minősége nagyításkor romlik, a vektorgrafikusoké nem

A tárolási mód egyben meghatározza a vektorgrafikus kép legfontosabb előnyét a pixelgrafikussal szemben: a minőségromlás nélküli nagyíthatóságot. Az ábrát alkotó alakzatokat leíró egyenletek átalakításai nem függenek a felbontás értékétől.

Vektorgrafikai szerkesztőprogramból többféle áll rendelkezésre. Egy részük az irodai programokba integráltan érhető el, például a Microsoft Wordben és PowerPointban a Rajzeszközök alkalmazásával vagy a LibreOffice Writerben és Impressben a Draw eszközzel. (A Draw önálló programként is elérhető a LibreOffice-ban.) Másik részük önálló vektorgrafikai szerkesztőprogram. Ezek közül ingyenesen letölthető és használható az *Inkscape* program, ezért ezzel foglalkozunk a tankönyvben. Nagy tudású, de igen költséges a CorelDraw, az Adobe Illustrator és a Xara Designer. Harmadik részük online webes felületen használható, például: Drawser, Boxy SVG, Vectr stb., vagy mobiltelefonon például: Pluma Vector SVG.

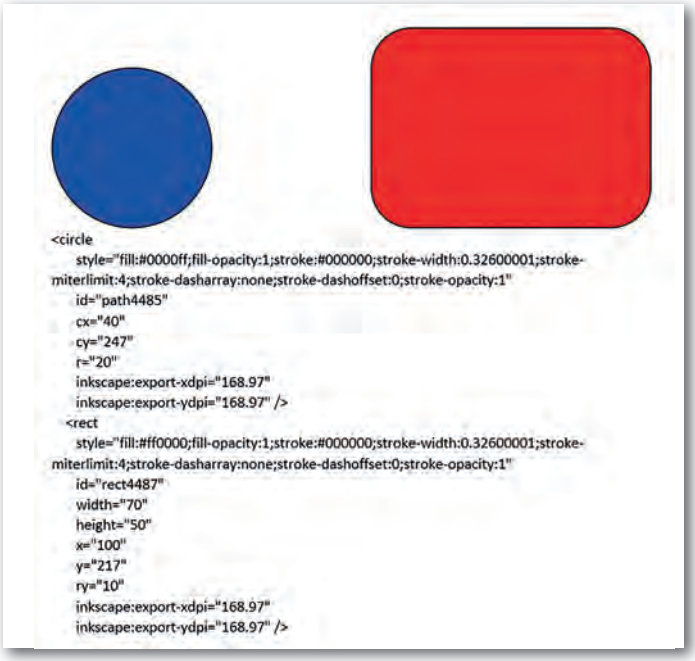
► A Boxy SVG Editor felhasználói felületének részlete

Az alkalmazói programok, például a szöveg- és prezentációszerkesztők integrált vektorgrafikai eszközeinek használatát a megfelelő fejezeteknél tárgyaljuk.

Az Inkscape program

Az Inkscape **nyílt forráskódú** vektorgrafikus szerkesztőprogram. A 2000-es évek elején jelent meg, és még ma is folyamatosan fejlesztik. Feladataink megoldásához a magyar nyelvű változatát használjuk. Elérhető többféle operációs rendszer alatt, ezért **keresztplatformos**nak hívjuk. Alapértelmezett képformátuma az SVG.

Az **SVG (Scalable Vector Graphics)** magyarul skálázható vektorgrafikának fordítható. Egy leírónyelvet jelent, amely a vektorgrafikus alakzatok megadására szolgál. Az állóképek mellett animált ábrák készítésére is alkalmas, de ezt az Inkscape még nem támogatja. Az SVG-formátumot nyílt szabvány határozza meg a W3C nemzetközi szervezet irányításával, hasonlóan a HTML-hez. A weblapszerkesztéshez hasonlóan **az SVG-állományok egyszerű szövegszerkesztőkkel olvashatók és szerkeszthetők,** de az alakzatok paramétereinek számsorából elég nehéz elképzelnünk, hogy milyen ábrát is írnak le. A mai webböngészők nagy része az SVG-formátumú képeket meg tudja jeleníteni, azaz az SVG-képek weblapokba beágyazva is felhasználhatók. Az SVG-állományok szövegeket, bitképet és hivatkozásokat is képesek tárolni XML formátumban.



```
<circle
    style="fill:#0000ff;fill-opacity:1;stroke:#000000;stroke-width:0.32600001;stroke-
miterlimit:4;stroke-dasharray:none;stroke-dashoffset:0;stroke-opacity:1"
    id="path4485"
    cx="40"
    cy="247"
    r="20"
    inkscape:export-xdpi="168.97"
    inkscape:export-ydpi="168.97" />
  <rect
    style="fill:#ff0000;fill-opacity:1;stroke:#000000;stroke-width:0.32600001;stroke-
miterlimit:4;stroke-dasharray:none;stroke-dashoffset:0;stroke-opacity:1"
    id="rect4487"
    width="70"
    height="50"
    x="100"
    y="217"
    ry="10"
    inkscape:export-xdpi="168.97"
    inkscape:export-ydpi="168.97" />
```

► Kört és lekerekített sarkú téglalapot bemutató SVG-kódrészlet

Kérdések, feladatok

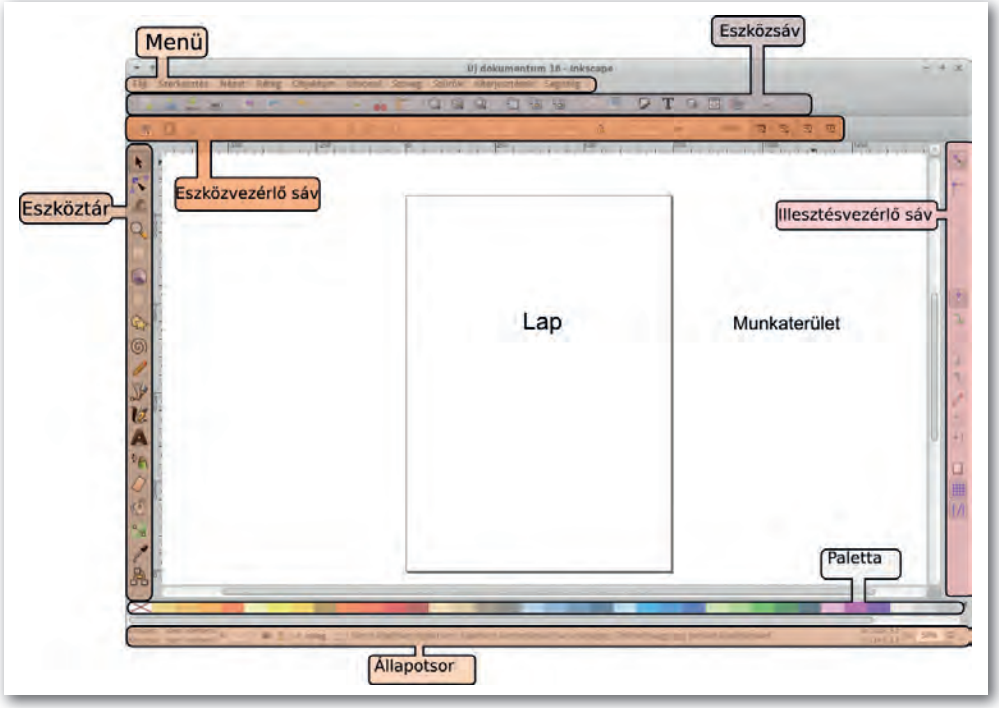
1. Keressük meg az előző kódrészletben az alakzatok kitöltésének színét!
2. Milyen tulajdonságok meghatározása olvasható ki a kódrészletből?
3. Keressünk a mobiltelefonok operációs rendszerének megfelelő applikáció-adatbázisban vektorgrafikus szerkesztőprogramokat, és olvassuk el rövid ismertetésüket!
4. Soroljunk fel olyan szakmákat, ahol speciális vektorgrafikus szerkesztőprogramot használnak!
5. Milyen típusú képszerkesztő programot és tárolási formátumot javasoljunk az alábbi képek átalakításához?

► Képek a tizedikes földrajzmunkafüzet (FI 506011002_1) 40. oldaláról

6. Nézzünk utána, hogy az XML-alapú adattárolásnak milyen előnyei vannak!
7. Keressünk és mentsünk le egy egyszerű geometriai alakzatokat tartalmazó SVG-ábrát az internetről! A lementett képfájlt nyissuk meg egy szövegszerkesztővel, és a kódban azonosítsuk az alakzatokat!

Felhasználói felület

A vektorgrafikus szerkesztőprogramok felhasználói felületei funkcionálisan nagyon hasonlóak egymáshoz. A továbbiakban az Inkscape keretrendszerét vizsgáljuk és használjuk.



► A felhasználói felület részei

A professzionális szerkesztőprogramokhoz hasonlóan a felhasználói felület az egyéni igényeknek, a kézreálló munkakörülményeknek megfelelően átrendezhető, egyes összetevők megjelenítése ki-, illetve bekapcsolható.

Az **Eszköztár** azokat a rajzeszközöket tartalmazza, amelyekkel az ábra alakzatait, objektumait hozzuk létre.

Az **Eszközvezérlő** sávon az éppen használatba vett eszközhöz tartozó beviteli mezők és gombok vannak. Például spirál rajzolásakor itt lehet megadni a fordulatok számát, a tágulást és a belső sugár méretét. Használatával precízebb munka végezhető, mint az egérrel történő módosításoknál.

A **Menü** sorában a program vezérléséhez tartozó parancsok érhetők el. Az általános, általában minden alkalmazás vezérléséhez használt menüpontok – *Fájl, Szerkesztés, Nézet, Segítség* – mellett speciálisan a vektorgrafikához tartozó parancsok – *Objektum, Útvonal, Réteg* stb. – is elérhetők.

Az **Eszközsáv** a legfontosabb, közvetlenül elérhető parancsok ikonjait tartalmazza. A parancsok egy része közvetlenül, azonnal végrehajtódik, ilyen például a *Kijelölt objektum kettőzése*. Néhány gomb hatására a rajzvászon jobb szélén dokkolható párbe-

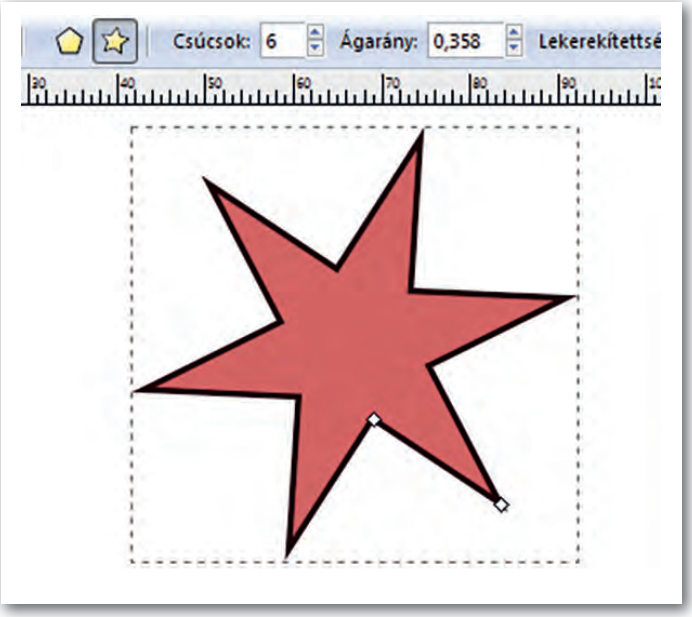
szédablak jelenik meg. A párbeszédablakok vagy más elnevezéssel panelek megjelenítése ki-be kapcsolható, de használatuk után automatikusan nem záródnak be.

Az **Illesztésvezérlő sáv** eszközeivel az objektumok vagy azok részeinek egymáshoz képesti elhelyezését, annak pontosságát lehet szabályozni.

A rajzvászon gördítősávja alatti **Paletta** az alakzatok kitöltéséhez és a szegélyük színének beállításához használható. A színminták fölött az egér buboréksúgójában azok neve és RGB-kódja jelenik meg.

Az **Állapotsor** tartalma a szerkesztés aktuális lépéséhez ad különböző információkat. Itt jelenik meg, hogy az egyes váltóbillentyűket (CTRL, SHIFT, ALT) használva hogyan végezhetjük el a műveleteket. A rétegekről és a rétegműveletekről állapotleírást kapunk.

Válasszunk ki az Eszköztárról egy alakzatot, például a *Csillagok és sokszögek rajzolása* ikont. Hatására egy csillag vagy egy sokszög jelenik meg. Közöttük a bal oldalon látható két ikonnal lehet váltani.



Kérdések, feladatok

1. Milyen tulajdonságokat állíthatunk be az Eszközvezérlő sávon a csillag alakzat beszúrása után? Készítsünk 5, 6 és 24 ágú csillagot!
2. Vizsgáljuk meg az Eszköztár alakzatainak beszúrása után az Eszközvezérlő sávon állítható tulajdonságokat! Az alakzaton megjelenő csomópontok mozgatásával azonosítsuk a tulajdonságokat!
3. A *Menü* > *Fájl* > *Mentés másként...* segítségével mentsük ábránkat az alapértelmezett SVG-formátumba! Készítsünk rasztergrafikus képet a *Menü* > *Fájl* > *PNG kép exportálása...* menüponttal a szerkesztéskor még vektorgrafikus eszközökkel készült ábrából. Ezzel a funkcióval a kép tárolási módját változtatjuk meg.

Alakzatok

Az **Eszköztáron** kiválasztott alakzatot a munkalapra helyezhetjük az egér segítségével. Az, hogy milyen állapottal (mintázat, kitöltőszín, szegélyvastagság, szegélyszín stb.), az az előző használatnál beállított értéktől függ. Az alapvető alakzatok az ikonjuk alapján könnyen kiválaszthatók: *Téglalap, Ellipszis, Csillag, Spirál.*

Tipp: Ha ellipszis helyett kört, téglalap helyett négyzetet akarunk rajzolni, akkor tartsuk lenyomva CTRL billentyűt. Persze ha valamelyik irányban jobban meghúzzuk, akkor nem lesz szabályos az alakzat.

Az Eszköztáron a legfelső ikon a Nyíl. A leggyakrabban használt eszközök közé tartozik, mert ezzel jelöljük ki az alakzatokat.

Az alakzatok, például a kör méretét nem a sugár hosszával, hanem az objektumot kijelölő szaggatott vonalú négyzet oldalhosszával adjuk meg. A kör esetén ez éppen az átmérővel azonos, de más alakzatnál lehet, hogy semmilyen nevezetes szakaszának hosszával nem egyezik meg.

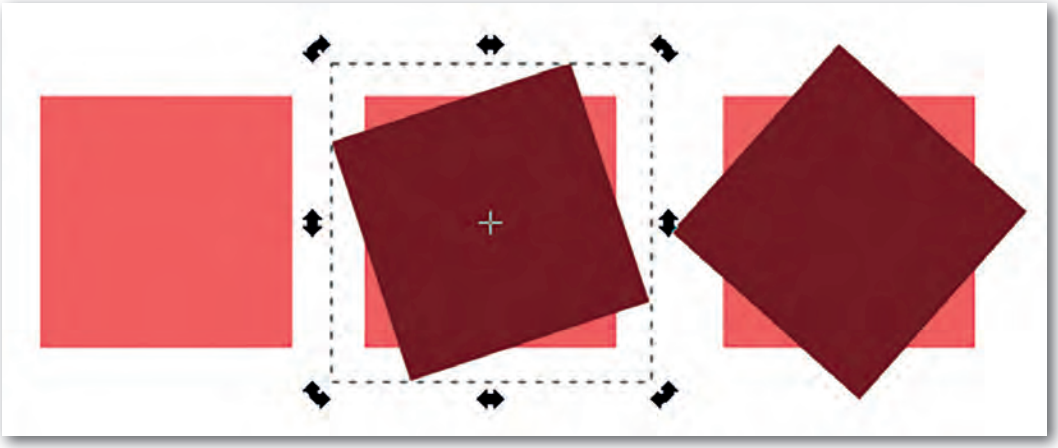
1. példa: Négyzetek

Készítsük el az alábbi ábrát!

A bal felső négyzetekből álló rész másolata a jobb alsó. Ezért az első elkészítése után a másodikat majd duplázással és igazítással fogjuk elhelyezni. Vizsgáljuk meg, hogy az első ábra hány négyzetből épül fel, és azok között hány azonos méretű!

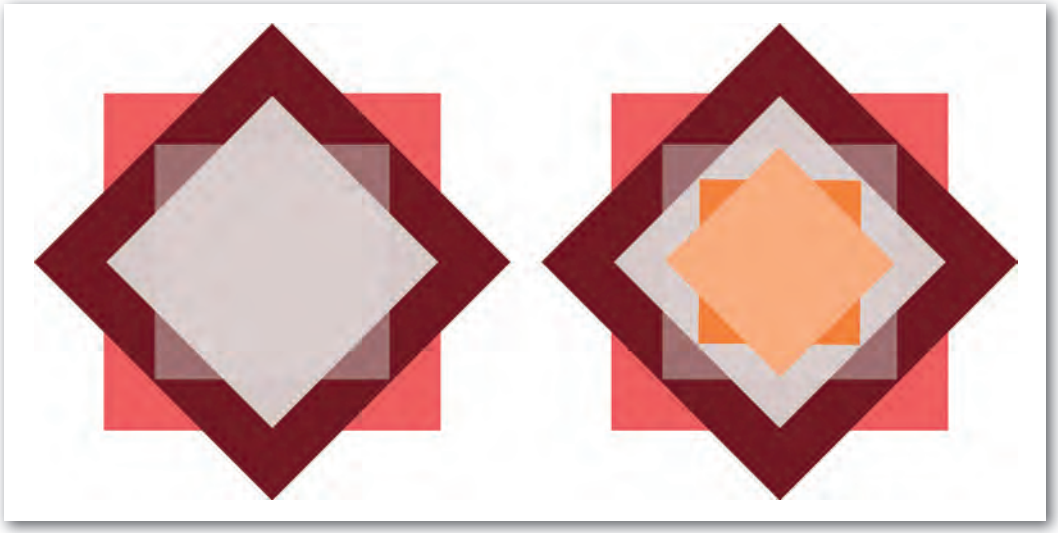
Először a leghátsó négyzetet rajzoljuk meg! A Téglalap eszköz kiválasztása után a CTRL gomb folytonos nyomva tartása mellett szabályos alakzatot, négyzetet rajzolhatunk. Kitöltő színét a palettáról válasszuk meg tetszőlegesen! Az alakzatok szegélyének színét a palettára a SHIFT gomb nyomva tartása mellett tudjuk kiválasztani. Most ne legyen a négyzetünknek szegélye, ezért a SHIFT gomb lenyomásával kattintsunk a paletta bal oldalán található jelre.

Jelöljük ki az Eszköztár nyíl kijelölő eszközével a négyzetet, majd készítsünk róla másolatot a *Menü > Szerkesztés > Kettőzés paranccsal*, vagy a CTRL + D (D: Duplicate, jelentése dupláz, megkettőz) billentyűkombinációval vagy az Eszközsáv ikonjával. A másolat az eredetivel megegyező helyen jön létre, így azok egymást takarják. Az alakzatra még egyszer rákattintva azt forgatni tudjuk. Ha a forgatást a CTRL egyidejű nyomva tartásával végezzük, akkor a forgatás 15°-onként történik. Így a szükséges 45°-ot könnyen beállítjuk.



► A négyzet méretezése és forgatása CTRL lenyomása mellett

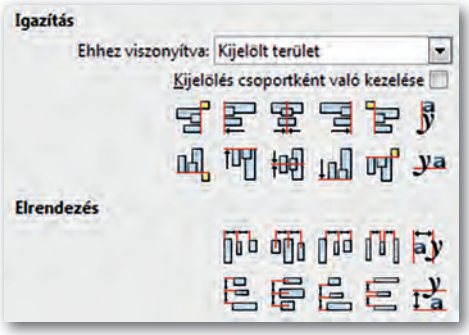
Az elforgatott négyzetet színezzük át tetszőlegesen. Az egymáson lévő két négyzetet foglaljuk az ikonnal csoportba, hogy véletlenül se mozdítsuk el őket egymáshoz képest, illetve együtt tudjunk róluk másolatot készíteni. A másolatot méretezzük át a CTRL + SHIFT váltóbillentyűk használatával együtt! Ezek a méretezést annyiban befolyásolják, hogy a folyamat középpontos kicsinyítéssel játszódik le. A két kisebb négyzetet ismét színezzük át, majd foglaljuk csoportba az összes négyzetet!



A rajzolás közben szükség lehet arra, hogy az eddig csoportosított négyzeteket egymástól elkülönítsük, azaz a ikonnal a csoportot szétbontsuk, a színeket beállítsuk, és a részeket újra csoportosítsuk.

Igazítás

A műveletek során szükség lesz az alakzatok igazítására. Általában az objektumokat egymáshoz képest igazítjuk, ezért jelöljük ki az összes alakzatot, amin végre szeretnénk hajtani a műveletet. Az igazítás *ikonra* kattintva az *Igazítás és elrendezés* panel nyílik meg.



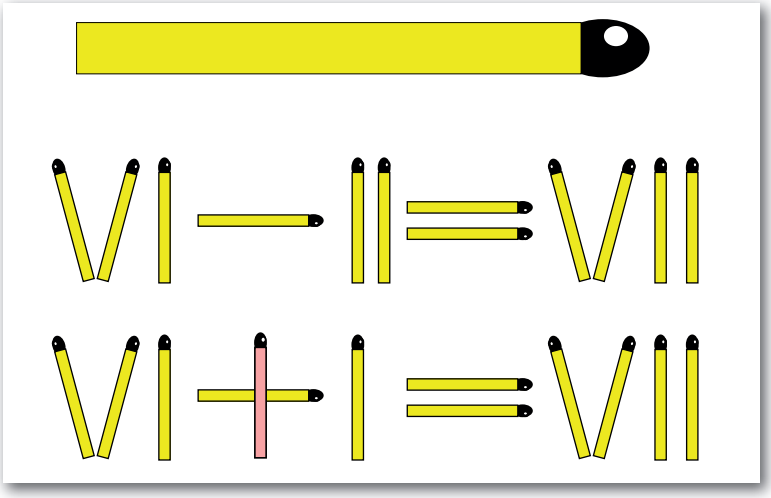
Az ikonokon a piros vonal jelzi, hogy az igazítás milyen rögzítéshez történik. A felső sorokban a vízszintes, az alsóban a függőleges irányú eszközök vannak.

Feladatok

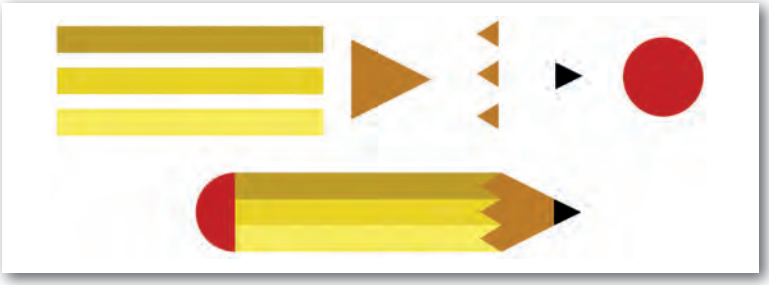
1. Készítsük el egy csiga, majd a csigacsalád rajzát! Milyen geometriai alakzatokból rakjuk össze a csiga rajzát?

2. Rajzoljunk néhány közlekedési táblát! A táblákhoz vagy az ábrázolt szimbólumokhoz a *Csillagok és sokszögek rajzolása* eszközre lesz szükség. Figyeljük meg az Eszközvezérlő sáv tartalmát ennek használatakor! Itt lehet váltani a csillag és a sokszög rajzolása között, illetve itt adhatjuk meg a csúcsok számát.

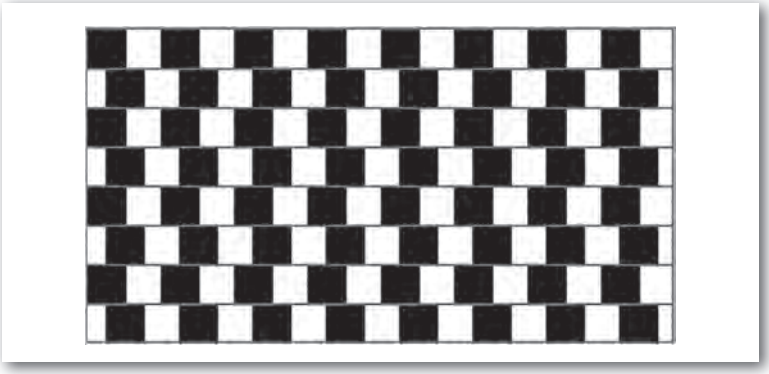
3. Rajzoljunk gyufaszálat, majd annak felhasználásával gyufarejtvényt és megoldását! Az áthelyezett gyufa színét változtassuk meg!



4. Rajzoljunk egyszerű alakzatokból ceruzát! A hatékony munkavégzéshez érdemes a ceruza részeit előre elkészíteni, és csak utána összerakni az ábrát.

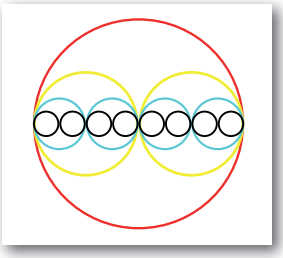


5. Készítsük el az optikai csalódást bemutató ábrát! Az optikai csalódások a látási folyamat részeinek tévedéseiből jönnek létre, amikor az agy olyan jeleket kap a látóidegektől, amelyek számára ellentmondóak. A párhuzamos fekete-fehér négyzetek alsó és felső élei egymáshoz képest elhajlani és a szélek felé összetartani látszanak.



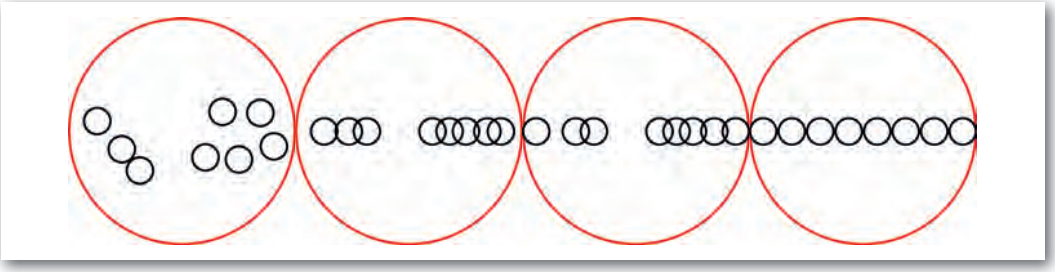
Elrendezés

2. példa: Körök



Színes körökből álló ábrát készítünk. Milyen összefüggés van az egyre kisebb körök sugara között?

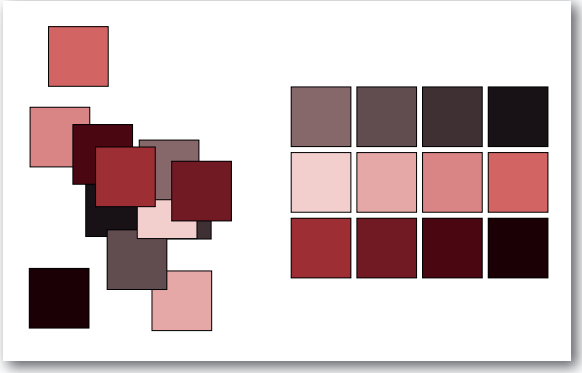
Ha a legnagyobb kör átmérőjét, ami egyben a **befoglaló négyzetének** oldalhossza is, 16-tal osztható egységnek választjuk, akkor a többi kör átmérőjét könnyen ki tudjuk számítani. Ezeket a méreteket az eszközsávon beállíthatjuk. Érdemes például 640 px oldalhosszt választani a piros körnek. Ekkor a sárga 320 px, a türkiz 160 px és a fekete 80 px oldalhosszú lesz. A szükséges számú és színű körökből kettőzéssel a példányokat előállítjuk. Most csak a fekete körök elrendezését kövessük, hiszen a többiét hasonlóan kell elvégezni. A 8 kört függőlegesen középre igazítjuk, majd a fekete körök közül a bal szélsőt a piros körrel balra, a jobb szélsőt jobbra igazítjuk.



Ezután jelöljük ki az összes fekete kört, és az *Igazítás és elrendezés* panelen az elrendezés funkciót használjuk. *A közepek egyenletes távolságban való vízszintes elrendezése* ikon hatására a két szélső közé a többi kört azonos távolságokra – most érintkezve – elhelyezi.

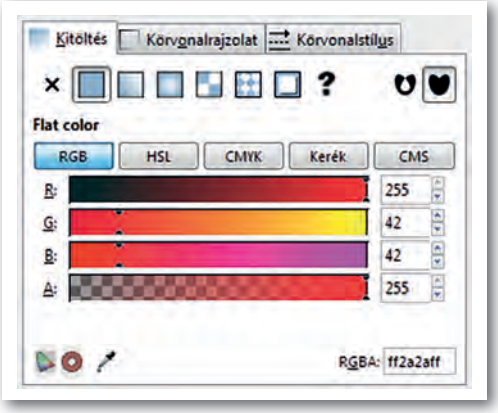
Feladat

Az *Igazítás és elrendezés* eszközeivel készítsük el az ábrát! *Véletlenszerű és rendezett* elrendezést válasszunk!



Színek, kitöltés, szegélyek

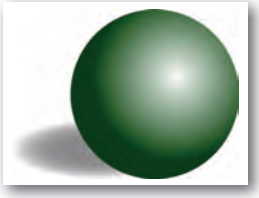
Az alakzatok szegélyét és színét már eddig is állítottuk a méret és elhelyezés mellett, most részletesebben vizsgáljuk meg a beállítási lehetőségeket. A *Kitöltés és körvonal* panel az ikon lenyomására nyílik meg.



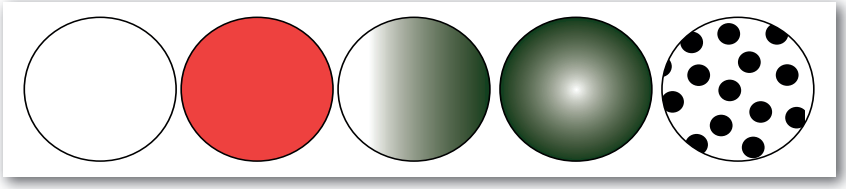
▸ Az alakzatok kitöltését és a szegély beállításait a Kitöltés és körvonal panelen állítjuk

3. példa: Színes golyók

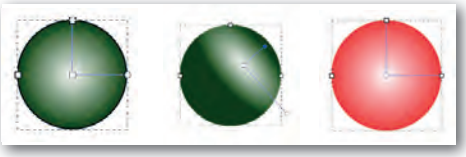
Ebben a példában színes golyókat rajzolunk, melyek térhatását csillogással és árnyék beállításával fogjuk szemléltetni. Mindezt a kitöltés módosításával érjük el.



Az alakzatok kitöltésénél a *Nincs*, az *Egyenletes*, a *Lineáris*, *Sugár irányú színátmenetes* és a *Mintával* való kitöltést választhatjuk, a lenti öt ábra ezekre mutat példát.



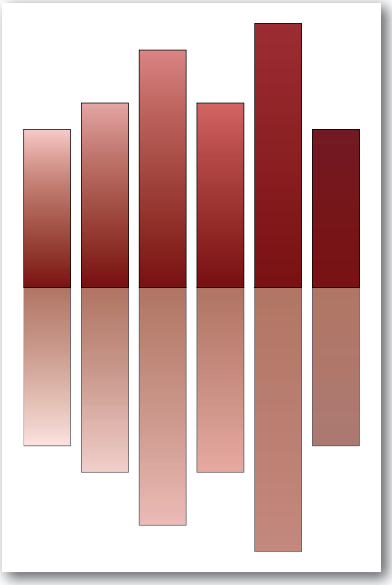
A golyó térhatását sugár irányú színátmenet alkalmazásával és annak további módosításával érhetjük el. A megvilágítás csillogását el kell mozgatnunk középről, és meg kell adnunk, hogy mely színek között legyen átmenet. Ehhez a Csomópont-kijelölő eszközzel a körre kattintunk. A rajzon a szerkesztést elősegítő csomópontok jelennek meg.



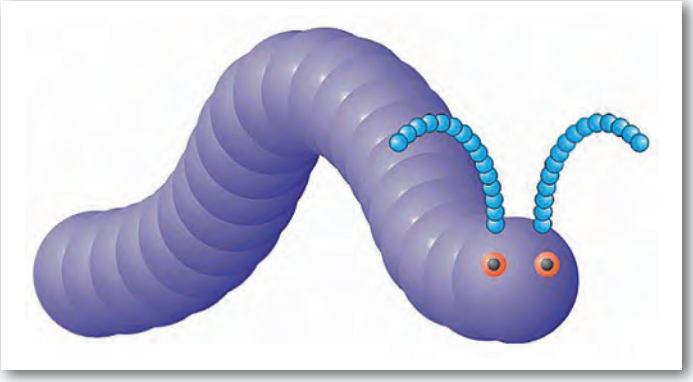
A kör közepén lévő négyzet alakú csomóponttal a belső szín indulási helyét mozgathatjuk más helyre, a két vele összekötött kör alakú csomóponttal a színátmenet ütemét, lefutását szabályozhatjuk. A négyzet és a kör csomópontokkal külön-külön beállíthatjuk a kiindulási színeket és átlátszóságukat.

Kérdések, feladatok

1. Mutassuk be egy színátmenetes téglalapokból álló ábra tükröződését egy tükröző felületen! Használjuk az Átlátszatlanság és Elmosás százalékos értékének csökkentését!



2. Rajzoljunk színátmenetes körökből ábrákat, például kukacot!



Unió, metszet, különbség

A matematikában megismert halmazműveleteknek megfelelő eszközöket alakzatokra is használhatunk. A *Menü* > *Útvonal* > Unió , Metszet és Különbség műveletek ikonjai szemléletesen mutatják hatásukat. Az ábrakészítés során az alakformáláshoz használjuk ezeket az eszközöket.

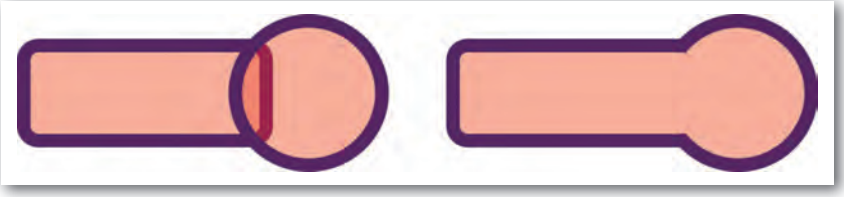
4. példa: Megálló

Készítsük el a közlekedési vállalat járműveinek megállótábláját a halmazműveletek segítségével!

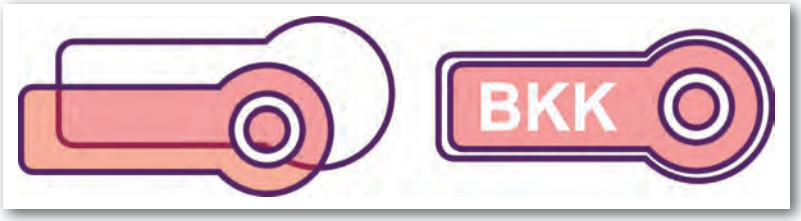
A tábla alapelemeit egy lekerekített sarkú téglalap és egy kör uniójával készíthetjük el. A fehér színű karikát és a feliratot később is elhelyezhetnénk, de az egyesítés után a fehér kört vízszintesen már nem tudnánk középre igazítani. Ezért a fehér kör elkészítését a lila kör megrajzolása után tegyük meg. Ezt elkészíthetnénk egy átlátszó kitöltésű, vastag fehér szegélyű körből, de most készítsük két eltérő sugarú kör különbségével!

Az alakzatok jobb láthatósága miatt a lila RGB(88, 39, 94) kódú szín beállítását a tábla összeállításának végére hagyjuk. A fázisrajzokban részben átlátszó alakzatokat használunk. Az első két fázisrajz szerint a két kört egymáshoz képest középre igazítjuk, majd a harmadik rajznak megfelelően a különbségükkel létrehozzuk a tábla fehér körgyűrűjét.

A táblát a téglalap és a kör uniójával rakjuk össze a fázisrajznak megfelelően.



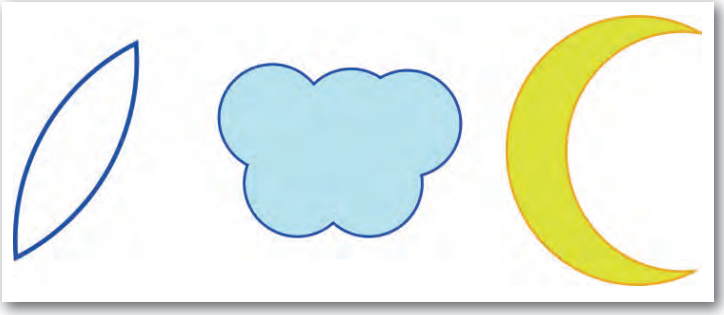
A következő lépésben a táblát átmásoljuk, a másolat méretét megnöveljük, kitöltését fehérre változtatjuk és átlátszatlanságát megszüntetjük. Végül a tábla részeit összerakjuk.



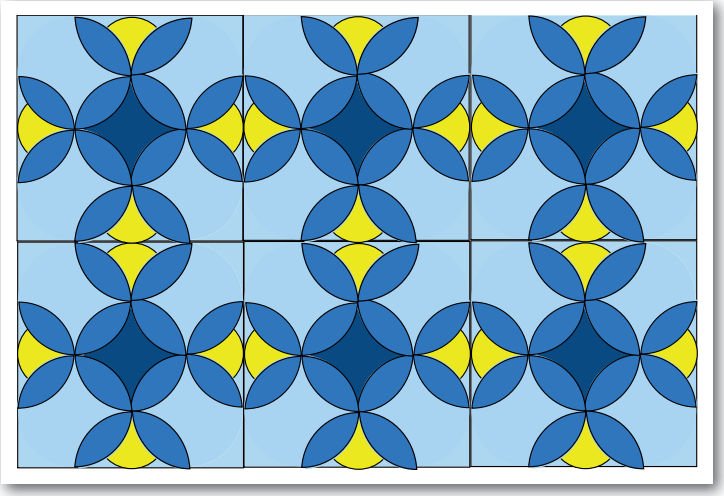
A tábla feliratát az *Eszköztár* *Szöveg objektum létrehozása és szerkesztése* funkciójával írjuk le. A szöveg *Arial* betűtípusú és *félkövér* stílusú. Méretét a tábla arányaihoz állítjuk. A megfelelő részeket a minta szerint igazítjuk egymáshoz, kitöltésük színét, a szegélyeket beállítjuk, és a rajzunk elkészült.

Feladatok

1. Készítsük el a következő szirom-, felhő- és Hold-ábrákat a körök halmazműveleteivel!



2. Készítsük el a következő díszes ábrát, miután végiggondoltuk a készítés menetét!



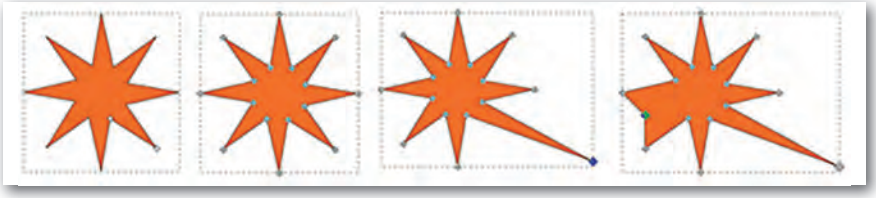
Útvonal

Ebben a fejezetben arról ejtünk szót, hogyan tudjuk az Inkscape által biztosított alapalakzatok jellemzőit módosítani, illetve hogyan tudunk vonalakból, görbékből saját készítésű, bonyolultabb ábrákat rajzolni.

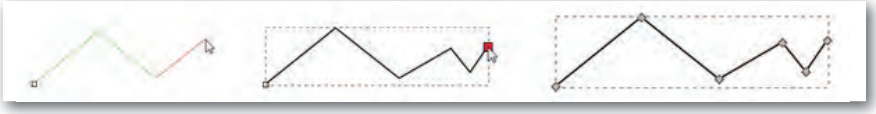
Először azt vizsgáljuk meg, hogy az *Eszköztár* rajzobjektumait hogyan tudjuk testre szabni. Legtöbbször az objektum alakját szeretnénk módosítani. A méretet a *Nyíl* eszközzel történő kijelölés hatására megjelenő nyilakkal módosíthatjuk. Az alak módosításához a ***Csomópont-kijelölő*** eszközt kell használnunk. Például a *Csillag alakzatnál* két csomópont jelenik meg, amelyek egyikével a méretet és elforgatást, a másikkal a csúcsok hegyességét tudjuk állítani. Ha csak az egyik csúcs méretét akarjuk növelni, de a többiét nem, új eszközt kell használnunk.

A *Menü* > *Útvonal* > *Objektum átalakítása útvonallá* parancs hatására a sokszög minden töréspontja csomóponttá alakul. **Az objektumot alkotó csomópontok sorozatát nevezzük útvonalnak.**

A csomópontok a többitől függetlenül mozgathatók el, és ezzel az objektum alakját meg tudjuk változtatni.



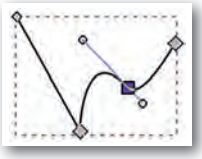
Saját magunk is létrehozhatunk útvonalakat, például vonalas ábrák átalakításával. Vonalas ábrát a *Bézier-görbék és egyenes vonalak rajzolása* eszközzel tudjuk elkészíteni. (A továbbiakban a rövidség kedvéért *Bézier-görbe* eszközként hivatkozunk erre.) Egyenes szakaszokból álló törött vonalat rajzolunk, és ezt a vázlatot fogjuk finomítani. A Bézier-görbe ikon kiválasztása után szakaszokból álló sorozatot rajzolunk úgy, hogy minden kattintás egy szakaszvégpontot hoz létre. A sorozatot dupla kattintással tudjuk lezárni.



► Törött vonal rajzolása, befejezése és útvonala

A csomópontokat mozgatni és szerkeszteni lehet. *Csomópont-kijelölést* használva az alakzat útvonala jelenik meg a rajzterületen, az Eszközvezérlő sávon pedig a csomópont-átalakító ikonok.

A rajzon és az ikonokon kétféle csomópontot látunk. A csúcs csomópont olyan csomópont, ahol az útvonal élesen megtörik. Az ív csomópontnál az átmenet görbe mentén történik. Az utóbbinál iránypontok segítik a görbe ívének beállítását.



► Három csúcs- és egy ívcsomópont az útvonalon

A csomópont eszköz vezérlőikonjaival a csomópontok típusát, az összekötő szakaszok alakját lehet változtatni.

5. példa: Alma

Rajzoljunk almát vázlat segítségével, majd annak finomításával, végül színátmenetes kitöltésével!

► Alma rajzának finomítása csomópont-műveletekkel

Az alma vázát a *Bézier-görbe* eszközzel szakaszokból rajzoljuk meg. A csomópont-kijelölő eszköz a bal oldali ábrán mutatja, hogy csúcs csomópontok jönnek létre, amelyeket a középső ábrának megfelelően ívesekké alakítunk. A csomópontok közötti szakaszokat ívesekké tesszük, ha szükséges, akkor a görbültségüket állítjuk. Végül színátmenetes kitöltést állítunk.

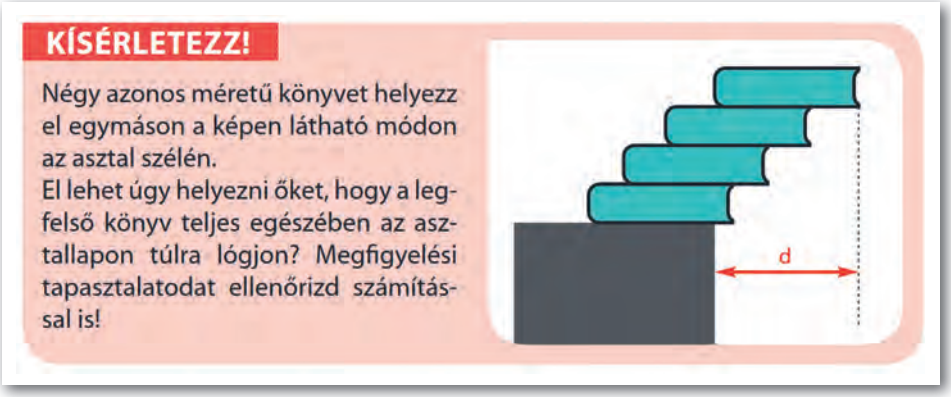
Érdekesség: Az eszköztárban van *Szabadkézi vonalak rajzolása* ikon. Ennek segítségével művészi érzékkel rendelkezők almához hasonló zárt görbét rajzolhatnak egy vonallal. Ezt csomópont-kijelölővel megnézve azt láthatjuk, hogy nagyon nagy számú csomópont jön létre, ami elsőre kezelhetetlennek tűnik. A *Menü* > *Útvonal* > *Egyszerűsítés* a nevének megfelelően egy grafikai algoritmus alapján csökkenti a csomópontok számát, és a továbbiakban úgy járhatunk el, mint az előző megoldásnál.

Kérdések, feladatok

1. A következő négyszögek közül melyek kialakításához kell a Téglalap alakzatot útvonallá alakítani?

2. Rajzoljunk biliárdasztalt! A golyók színátmenetesek legyenek úgy, hogy belül a térhatás érzékeltetésének megfelelően fehér, kívül tetszőleges más színűek legyenek! A golyók elrendezése, takarása a mintának feleljen meg!

3. Készítsük el a fizikatankönyv alábbi ábráját! Figyeljük meg, hogy a hátteret adó téglalapok három sarka lekerekített, de a negyedik nem!



► Kép a kilencedikes fizikakönyv (FI-505040901_1) 117. oldaláról

4. Készítsük el csomópontműveletekkel körből az alábbi ábrákat! Első lépésként a kört alakítsuk át útvonallá, majd a megfelelő csomópontok típusát állítsuk át. Ha szükséges, akkor a megfelelő helyen a csomópontok számát növeljük meg!

Szövegek

Készülő ábráinkhoz szöveget is hozzáadhatunk, az *Szövegobjektumok létrehozása* ikonnal. Formázási beállításokkal bőven ellátott a szövegszerkesztési funkció, de alapvetően nem a szövegbevitel a vektorgrafikai szerkesztőprogramok feladata. A beállításokhoz szükséges *Szöveg és betűtípus* panel a ikonnal jeleníthető meg. Itt a szokásos beállítások érhetők el: betűtípus, méret, stílus és igazítás. A szöveg gépelése közben az *Eszközvezérlő sáv* további érdekes beállítási lehetőségeket ad.

A felső és alsó index beállítása mellett sorközt, betűközt, a betűk vízszintes és függőleges alávágását (betűpárok távolságát), valamint az írás irányát állíthatjuk be.

6. példa: Póló

Rajzoljuk meg az Irka Iskola diákpólóját, illetve készítsük el annak feliratát!

Az üres pólót rajzoljuk meg (unió és különbség műveletekkel), vagy töltsük le a könyv weblapjáról! A görbített, körre illesztett szöveg elkészítése az újdonság. Kör alakú szöveget készítünk úgy, hogy a szöveg egyik része a körvonal külső, másik része a belső felére illeszkedjen.

Az alapgörbének választott kört és a kívül megjelenő szöveget a munkalapra tesszük. A szövegen érdemes néhány formázási beállítást, például a betűtávolság-állítást kipróbálni.

A szöveget és a kört együtt kijelöljük, és alkalmazzuk rájuk a *Menü > Szöveg > Útvonalra való illesztés* parancsot. A szöveg állását a kör elforgatásával tudjuk beállítani.

A kör méretét középpontosan (SHIFT + CTRL lenyomása mellett) növeljük meg az első szöveg betűméretével. A második szövegrészt is a körre görbítjük, és a szöveget belülre helyezzük a *H* billentyű (horizontális tükrözés) megnyomásával vagy a függőleges méretezőnyilak lefelé mozgatásával.

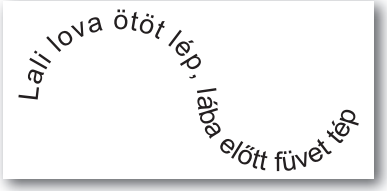
Következő lépésként a két szöveget útvonallá alakítjuk a *Menü* > *Útvonal* > *Objektum átalakítása* útvonallá paranccsal. Ezek után a kör elmozgatható és törölhető a szövegektől függetlenül.

A két szövegrészt egymáshoz képest igazítjuk, és szövegkör belsejébe valamilyen külön elkészített logót teszünk.

Utolsó lépés a póló összeállítása. Kitöltését zöld színűre állítjuk, és az elkészített feliratot csoportba foglalva rátesszük. Érdemes a felirat színét kontrasztosra, most pl. fehérre állítanunk.

Feladatok

1. Készítsük el a következő képverset!



2. Keressünk olyan érdekes képverseket, amelyeket el tudunk készíteni!
3. Állítsuk be a betűk közötti távolságot a minta létrehozásához!

TALPASI IRKA ISKOLA
TALPASI IRKA ISKOLA
TALPASI IRKA ISKOLA

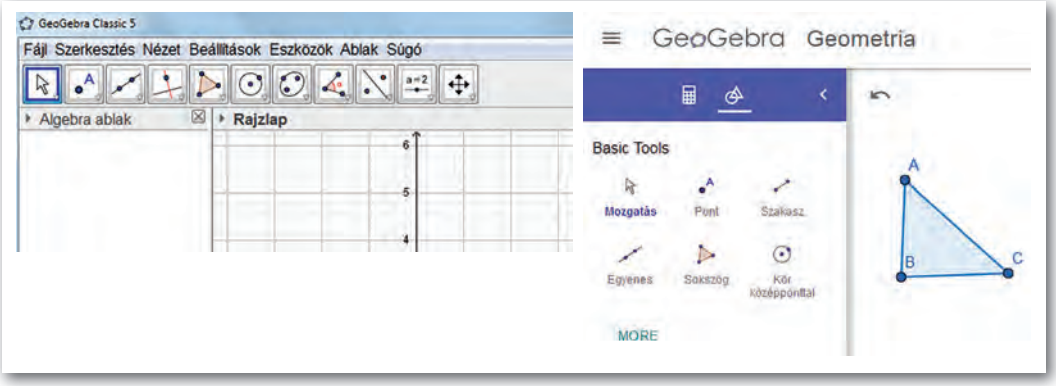
GeoGebra

Az általános célú Inkscape vektorgrafikus szerkesztőprogrammal végzett ábrakészítés után témaorientált, más funkciójú alkalmazást vizsgálunk meg. A matematikai problémák több területének – geometria, algebra, táblázatkezelés, függvényábrázolás, statisztika és az analízis – feladatai interaktív módon oldhatók meg a GeoGebra szoftverrel.

A GeoGebra ingyenesen letölthető, nyílt forráskódú **matematikai szoftver,** amely asztali alkalmazásként szinte minden operációs rendszerre elérhető. Futtatható online és mobilrendszereken (pl. okostelefonokon) is.

Most kizárólag az interaktív geometriai szerkesztésre nézünk példát, ezzel az alkalmazás használatának kezdő lépéseit, alapozását tesszük meg. A matematika- és fizikaórán folytatódhat a GeoGebra használata.

A vektorgrafikus szerkesztés többlete, hogy az alakzatok alkotóelemeinek helyzetét tetszés szerint megváltoztatva a szerkesztések követik az új helyzetet.



▶ A GeoGebra asztali és online felhasználói felülete

A GeoGebra szerkesztések különböző alakzatokat, matematikai objektumokat tartalmaznak. A geometriai szerkesztést eszközök és parancsok segítségével végezzük el. Az egyszerűbb szerkesztéseket leírások (idegen szóval tutorialok) segítik az interneten.

A GeoGebrában az *Eszköztárban* a szerkesztéshez használható geometriai alakzatok ikonjai szerepelnek.

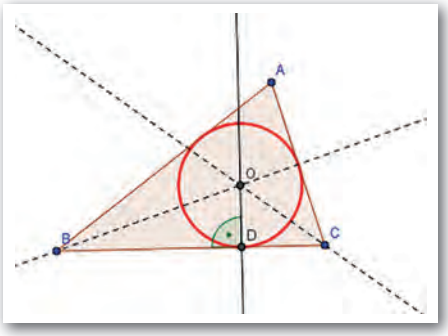
Az ikonok eszköztárakat nyitnak ki, rajzaik elég jól kifejezik a mögöttük álló tartalmakat: pont, egyenes, merőleges, sokszög, kör, ellipszis, szög, tengelyes tükrözés, csúszka és mozgatás.

A GeoGebrával készített szerkesztések, eljárások saját vektorgrafikus formátumba menthetők, és később újra felhasználhatók. A szerkesztés publikálható a világhálón, és exportálható pixelgrafikus formátumba is.

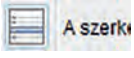
7. példa: Háromszög beírt köre

A háromszög szögfelezőinek metszéspontja a beírt kör középpontja. A beírt kör az egyetlen olyan kör, ami a háromszög mindhárom oldalát belső pontban érinti. Az érintési pontokba húzott sugarak merőlegesek a megfelelő oldalakra.

A szerkesztés menete lépésenként (1–8.) és annak bemutatása (9–13.).

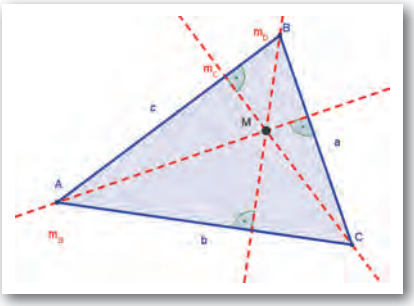


► Háromszög beírt körének szerkesztése GeoGebrával

	Eszköz	Megadás	Megjegyzés
1.		Sokszög: ABC háromszög felvétele	Záródnia kell
2.		Szögfelező: Csúcsok megadásával	
3.		Metszéspont: Szögfelezők metszéspontja	Jobb klikk: Átnevezés O-ra (Szabad és függő alakzatok)
4.		Merőleges: O-ból az egyik oldalra	
5.		Metszéspont: Érintési pont (D)	
6.		Kör középponttal és kerületi ponttal: középpont: O, kerületi pont: D	
7.		Szög: ODB szög felvétele	
8.		Tulajdonságok változtatása	Tengelyek eltüntetése Szín, vonalstílus Felirat mutatása
9.		Jobb klikk / Navigációs eszköztár	Ne legyen kijelölve objektum! *Lépésenkénti lejátszás*
10.		*Nézet > A szerkesztés lépései*	
11.		Töréspont jelölése	Törőpontok megadása
12.		Csak a töréspontok mutatása	
		Fájl > Mentés, exportálás	*ggb* kiterjesztés használata

Szerkesztés után a háromszög bármely csúcsát megfoghatjuk és elmozgathatjuk. Hegyesszögű háromszögben szerkesztettünk, de megnézhetjük, mi lesz tompaszögű háromszög esetén.

8. példa: Egy háromszög magasságvonalai és magasságpontja



Vegyünk fel egy tetszőleges állású ABC háromszöget! A szerkesztés talán könnyebben követhető, ha a háromszög hegyesszögű. A háromszöget középkékkel színezzük, és 5 pont vastag vonallal szegélyezzük. Az oldalak mellett megjelenítjük feliratként a neveket: a, b, c.

Megszerkesztjük a háromszög magasságvonalait:

Egy háromszög magasságvonala: a csúcsból a szemközti oldalra bocsájtott merőleges.

A magasságvonalakat m_a, m_b, m_c-vel jelöljük, és piros szaggatott vonallal rajzoljuk meg.

A keletkezett derékszögeket jelöljük! Ha szükséges, felvehetünk további segédpontokat is, de azokat ne jelenítsük meg! Csak a derékszög jelét jelenítjük meg, de a szög nagyságát nem.

A három magasságvonal egy pontban metszi egymást, ez a magasságpont. A magasságpontot M-mel jelöljük.

Feliratozzuk a következő objektumokat:

– a háromszög a oldalát a hosszával,
– az a oldalhoz tartozó magasságot a hosszával,
– a háromszöget a területével.

A feliratokhoz nem használunk külön címkéket!

A csúcsok mozgatásával itt is kísérletezhetünk az elkészült szerkesztésen.

Feladatok

1. A háromszög valamely csúcsának mozgatásával vizsgáljuk meg, hogy milyen helyzetbe kerül a magasságpont hegyesszögű, derékszögű, illetve tompaszögű háromszög esetében!
2. Mentsük az ábrát úgy, hogy a háromszög tompaszögű helyzetben álljon!

Mi az a programozás?

Mi a program?

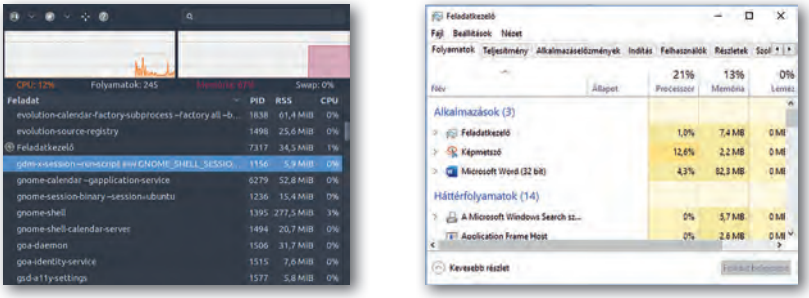
A számítógép, a telefon, az összes olyan eszközünk, amiben valamilyen számítógép van, önmagában képtelen ellátni azt a feladatot, amire készült. Csak egy darab „vas” – azaz hardver. Csak akkor képes igazán működni, ha fut rajta egy (vagy sok) program, alkalmazás – azaz szoftver. Szoftver, program, alkalmazás – nagyjából ugyanazt jelenti: azt a programozó, szoftverfejlesztő által megírt valamit, ami elmondja a hardvernek, hogy mikor mit csináljon.

Program mondja el a kenyérsütőnek, hogy meddig gyúrja a tésztát, meddig hagyja dagadni, és mikor kezdje sütni, mennyire legyen meleg a fűtőszál, és hányat sípoljon a sütő, amikor kész a kenyér. Program mondja el a mosógépnek, hogy mikor és mennyi vizet szívjon be, mennyire melegítse meg, meddig forogjon benne a dob, és meddig kell centrifugáznia. Ezek a számítógépek egyetlen programot futtatnak.

Az informatikaórán bennünket jobban érdekelnek a hagyományos értelemben vett számítógépek (laptopok, asztali gépek, szerverek) és a mobileszközök. Ezek csak bekapcsoláskor futtatnak egyetlen programot, ami azt mondja el nekik, hogy honnan és hogyan kell betölteniük a „fő” programjukat: az operációs rendszert. A többi program (a böngésző, az üzenetküldő, a játék, a képszerkesztő, a szövegszerkesztő, a filmvágó stb.) pedig az operációs rendszerből, annak felügyelete alatt indul el, akár úgy, hogy rákattintunk az egérrel vagy rábökünk az ujjunkkal az indítóikonjára, akár automatikusan.

Hol vannak a programok?

Az elindított, azaz futó programok a számítógép memóriájában vannak. Nemcsak a program van itt, hanem az általa éppen használt adatok is: a szövegszerkesztő által szerkesztett szöveg, a képszerkesztőbe betöltött kép. Az operációs rendszerünk feladatkezelőjében megnézhetjük az épp futó programokat. Látjuk, hogy a legtöbbet nem mi indítottuk el, sőt nem is látjuk őket – a háttérben futnak.

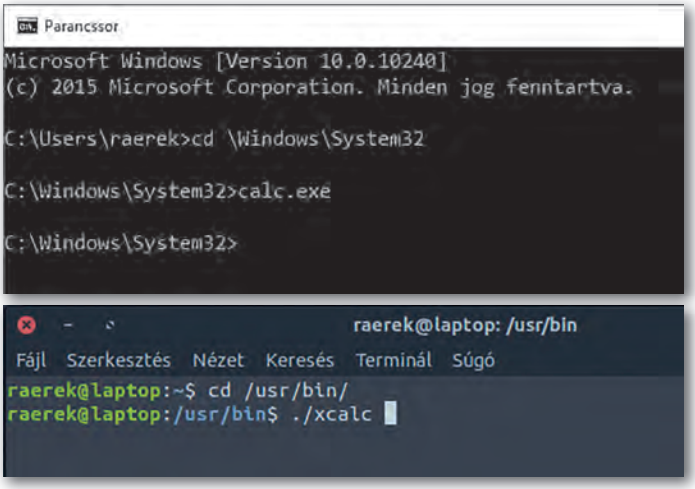


▶ Egy Linuxon futó feladatkezelő és a Windows 10 feladatkezelője – Indítsunk el egy programot, keressük meg a feladatkezelőben, és állítsuk meg innen!

Amíg a programot nem indítjuk el, a számítógép háttértárán, azaz a laptop SSD-jén, az asztali gép vagy szerver winchesterén, vagy a telefon memóriakártyáján van, ugyanolyan fájlként, mint a képek, zenék vagy szövegek. Az egyszerű programok egyetlen fájlból állnak, az összetettek sokszor nagyon sokból.

A grafikus felületű operációs rendszerek elterjedése előtt (az 1990-es évekig) a programokat úgy indítottuk el, hogy a parancssoros felületben beléptünk abba a mappába (könyvtárba), amelyikben a programunk volt, és beírtuk a program nevét. A módszer ma is működik, bár többnyire csak a számítógépekhez jobban értő emberek, rendszergazdák, rendszermérnökök és szoftverfejlesztők használják. Lévén e fejezet célja épp az, hogy kicsit mi is szoftverfejlesztők legyünk, ismerkedjünk meg ezzel a módszerrel!

1. Nyissunk a gépünkön parancssoros felületet: Windowson indítsuk el a Parancssor nevű alkalmazást, macOS-en és Linuxon pedig valamelyik terminált!
2. „`cd`” parancsokkal lépkedjünk abba a mappába, ahol a programfájl van (minden sor begépelése után Entert nyomunk)!
3. Írjuk be a program nevét, és nyomjuk meg az Entert!



► Program indítása parancssorból Windows 10-en és Ubuntu Linuxon

A program ilyenkor betöltődik a számítógép memóriájába, és a benne lévő utasítások végrehajtódnak, azaz a program futni kezd.

Mi van egy programfájlban?

Ha már úgyis a parancssorban, az imént elindított programunk mappájában vagyunk, adjuk ki

- Windowson a `type`
- macOS-en és Linuxon a `cat`

parancsot, és írjuk utána a programfájl nevét (például `type calc.exe`, `cat xcalc`)! Rengeteg krikszkrakszot ír a parancssori ablakba a gép. Ha némileg hihetetlen is, a számítógép ezt érti, ebből tudja, hogy mit kell csinálnia. Ez a program egyik alakja, az úgynevezett *gépi kódú* program, amely most a képernyőn karakterek formájában jelenik meg.

Szerencsére a legtöbb szoftverfejlesztőnek nem így kell megfogalmaznia a gép teendőit. Rendelkezésünkre állnak programozási nyelvek, azaz az angol nyelv szavait használó magasabb szintű nyelvek. Ilyen például a C, a C++, a C#, a Pascal, a Ruby, a Go, a Perl, a JavaScript, a Java, és még sorolhatnánk. Ilyen a mi könyvünkben használt **Python** is. A szoftverfejlesztő többnyire valamelyik ilyen programozási nyelvben készíti el a program **forráskódját**.

Egy egyszerű forráskódot többé-kevésbé már most is értelmezni tudsz. Mit csinál az alábbi, Python nyelvű program?

```
print('Üdv néked!')
évek_száma = input('Hány éves vagy?')
évek_száma = int(évek_száma)
if évek_száma < 14:
    print('Jé, hogyhogy már középiskolás vagy?')
else:
    print('Egy év múlva', évek_száma+1, 'éves leszel.')
```

Nos, ilyen és ehhez hasonló programokat fogunk mi is írni az elkövetkezendő órákon. Látjuk, hogy az angol szavak mellett van még a programban írásjel, műveleti jel, zárójel – ezek mind a program részei, nem hagyhatók el. A Pythonban szerepe van annak is, hogy a sor elején kezdődik-e egy programsor, vagy bentebb.

Természetesen ezt a programkódot a számítógép ebben a formában nem érti, és nem tudja futtatni. A fenti forráskódot egy másik program előbb gépi kóddá alakítja, és a gép processzora a gépi kódot értelmezve futtatja a programunkat.

Feladatok

1. Az alábbi, C++-nyelvű kód pontosan ugyanazt csinálja, mint a fenti Python nyelvű. Keressük meg a hasonlóságokat, mutassunk rá a különbségekre!

```
#include <iostream>
using namespace std;
int main()
{
    cout << "Üdv néked!" << endl;
    int evek_szama;
    cout << "Hány éves vagy?";
    cin >> evek_szama;
    if (evek_szama < 14 ) {
        cout << "Jé, hogyhogy már középiskolás vagy?";
    } else {
        cout << "Egy év múlva " << evek_szama+1 << "éves leszel." << endl;
    }
    return 0;
}
```

2. Rakjunk össze egy másik programot az alábbi részletekből!

Csak ennyiből áll egy szoftverfejlesztő munkája?

Nos, igen, a fejlesztői munka legismertebb része az új programok írása.

Sokkal többen vannak azok, akik meglévő programokat alakítgatnak át az új követelményeknek megfelelően (nekik köszönhetők például a telefonjainkra letöltendő frissítések).

Van, aki azzal foglalkozik, hogy egy már meglévő program fusson másféle gépen is – ez néhány esetben gyorsan megoldható, más programoknál nehéz és kimerítő feladat.

Vannak, akik azért dolgoznak, hogy egy meglévő program legyen gyorsabb.

Van, aki programokat tesztel: megnézi, hogy biztosan jól működnek-e mindenféle helyzetben.

Van, aki azzal foglalkozik, hogy programot, szoftverrendszert tervez. Ő már nem ír kódot, hanem azért felel, hogy a szoftver különböző részei minél jobban tudjanak együtt dolgozni.

Van, aki biztonsági ellenőrzést végez programokon, például azért, hogy számítógépes bűnözők ne tudják a banki szoftverekkel átutaltatni a pénzünket másik számlára.

A következő leckében mi is megírjuk első programunkat.

Kérdések

1. Mik azok a számítógépes vírusok?
2. Milyen egyéb feladatok merülhetnek fel egy szoftver elkészítésekor? Milyen képzettségű munkatársai vannak a szoftver fejlesztőjének?
3. Milyen kép él benned a programozókról? Milyen előítéletek kapcsolódnak hozzájuk?
4. Milyen világszerte ismert oldalakon foglalkoznak programozási kérdések megválaszolásával? Hány programozással kapcsolatos videó készül naponta?

Első programjaink

A programozási környezet

A programozási környezet arra való, hogy benne írjuk meg programjainkat, használatával a programot gépi kódúvá alakítsuk, és tesztelhessük, dokumentálhassuk a kész programot. Ezek közül a legfontosabb a gépi kódúvá alakítás, a többit vagy meg tudjuk oldani egy adott környezetben, vagy nem.

A programozási környezetet telepítenünk kell a gépünkre. A Python nyelv környezete a `python.org` webhelyről tölthető le, méghozzá – lévén a Python szabad szoftver – ingyenesen, bárki számára. Egészen biztosan találunk gépünknek és operációs rendszerünknek megfelelő változatot. Gépigénye nagyon kicsi, azaz bátran telepítsük öregebb, kisebb teljesítményű gépekre is.

Legelső programunk

Az első programunkat egy egyszerű szerkesztőben írjuk meg, ilyen például a Windows Jegyzettömbje. Indítsuk el, és gépeljük bele ezt az egyetlen sort:

```
print('Szia.')
```

Ha Linuxon vagy macOS alatt dolgozunk, akkor valamennyi Python programunk legeslegelső sora kötelezően

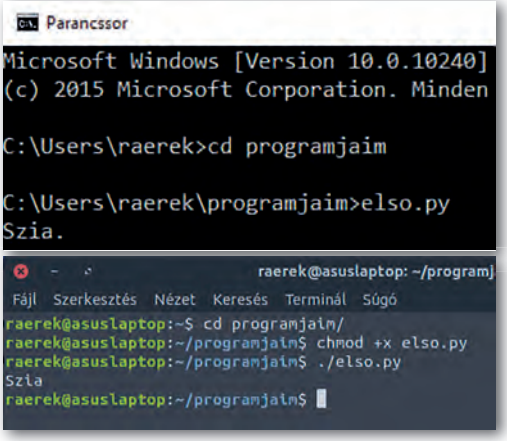
```
#!/usr/bin/env python3
```

legyen. Erre Windows alatt nincs szükség.

Mentsük el a fájlunkat – érdemes a programjainknak létrehozni egy mappát –, és figyeljünk rá, hogy a fájl kiterjesztése `.py` legyen, azaz a fájl teljes neve legyen például `elso.py`. Általában egy szóból álló és ékezetmentes neveket szoktunk programnévként használni. Egyes rendszerek a szóköz és az ékezetes betűk használatára érzékenyek. A `.py` kiterjesztés az állomány Python-forrásállomány jellegére utal.

A programokat mindig egyszerű szerkesztőkben írjuk, sohasem „igazi”, sok formázási és más funkciót tartalmazó szövegszerkesztőben. Ennek az a magyarázata, hogy a Word vagy a LibreOffice Writer a fájlba nemcsak azt menti, amit beleírtunk, hanem sok egyebet is, például a formázásokat.

Nyissunk parancssort, és a múlt alkalommal használt `cd` paranccsal lépjünk abba a mappába, ahova a fájlt mentettük. Linuxon és macOS-en még futtathatóvá kell tennünk a fájlt a `chmod +x elso.py` parancs kiadásával. Ezt követően futtathatjuk programunkat a program nevét beírva.



► Első programunk futtatása

Ha mindent jól csináltunk, fut a programunk, pontosabban, mire idáig jutunk az olvasásban, alighanem véget is ért a végrehajtása. Megírtuk az első programunkat!

Ha valamit nem írtunk be jól, akkor hibaüzenetet kapunk. Például:

```
C:\Users\raerek\programjaim>elsi.py
'elsi.py' is not recognized as an internal or external command,
operable program or batch file.
```

► Elgépeltük a fájlnevet. Ez nem programozási hiba, az operációs rendszer jelzi a hibát.

```
C:\Users\raerek\programjaim>elso.py
  File "C:\Users\raerek\programjaim\elso.py", line 1
    print('Szia.)
                 ^
SyntaxError: EOL while scanning string literal
```

► Valamit a programban írtunk rosszul. Ez a Python hibaüzenete.

Figyeljük meg, hogy miből áll egy hibaüzenet, alighanem sokat látunk még ilyet. A Python

- megmondja, hogy melyik sorban van a hiba (line 1),
- mutatja, hogy szerinte hol a hiba (néha pontatlanul),
- meg is fogalmazza a hibát (utolsó sor).

A programunkat nem kellett külön gépi kódúvá alakítanunk. Az átalakítást a Python automatikusan végzi a háttérben, mert a Python úgynevezett interpretált (értelmezett) nyelv. Megjegyezzük még, hogy programunk a parancssori felületen fut, és még jó darabig nem foglalkozunk grafikus felületű szoftverekkel, mert készítésük lényegesen bonyolultabb.

Az IDE

A szövegszerkesztővel történő programírás után más módszert, más fejlesztési környezetet használunk. Az IDE (Integrated Development Environment – integrált fejlesztői környezet) egy olyan alkalmazás, amely segít nekünk a program megírásában. A Python saját, alapértelmezett fejlesztői környezetének neve: IDLE. A név szóvicc: a szó hasonlít az IDE-re, de angolul tétlent jelent, holott mi épp itt fogunk sokat ténykedni.

Amikor az IDLE elindult, egy úgynevezett Python Shellt látunk, amiben szintén lefuttathatók a Python programok, bár mi ebben a könyvben mindig a parancssorban futtatjuk őket – ez csak egyéni ízlés kérdése.

A *File* menüből tallózva nyissuk meg az előbb elkészült programunkat, és látjuk, hogy az IDLE színesen jeleníti meg a programunkat – ebben segít nekünk az IDLE a Jegyzettömbhöz képest. Mostantól itt készítjük a programjainkat – az IDLE megnyitása után ne felejtsünk majd mindig új fájlt kérni, ne a Python Shellben akarjunk programot írni.

Szöveg és szám

Módosítsuk a programunkat úgy, hogy „Szia” helyett írja ki születésünk évét! Ez az adat egy szám, azaz nem kell aposztrófok közé tennünk. A programozási nyelvekben az a szokás, hogy a szöveges adatokat idézőjelek vagy aposztrófok közé kell tenni. A Pythonban mindkettőt használhatjuk, a megkötés az, hogy amelyiket a szöveg elejére írjuk, azt kell a végére is. Mi a könyvben a következő példakód kivételével az aposztrófoknál maradunk.

A számokat is írhatjuk idézőjelbe, ilyenkor a Python szövegként kezeli őket. Mit jelent ez? A legegyszerűbb, ha kipróbáljuk ezt a programot (mostantól számozzuk a programsorokat, úgy könnyebb beszélni róluk):

```
1. print('Hú' + "Ha')
2. print(2 + 3)
3. print("2" + '3')
```

Aposztróf és idézőjel is jó!

A számokat értelmezi és összeadja.

A szövegeket meg sem próbálja értelmezni, csak egymás mellé írja.

Változók

Írjunk új programot!

```
1. állat = 'ló'
2. print('állat')
3. print(állat)
```

Értéket adunk a változónak.

Kiírja a változó ÉRTÉKÉT.

Az első sor új elemet tartalmaz. Az idézőjelek nélküli szöveg itt egy *változó*, aminek azt adtuk értékül, hogy „ló”. Legegyszerűbb, ha a változókra olyan dobozként gondolunk, amibe bármit tehetünk – itt épp egy szöveget tettünk bele. Ezt a programunk megjegyzi, és ha legközelebb a doboz (változó) nevét írjuk le (idézőjelek nélkül), akkor behelyettesíti oda a doboz *tartalmát*. Ha a változó nevét aposztrófok közé írjuk, akkor a Python egyszerű szövegként tekint rá, amit meg sem próbál értelmezni.

A változókat azért hívjuk változóknak, mert az értékük változhat. Ha új értéket adunk nekik, a régi egyszer s mindenkorra nyomtalanul eltűnik. Gondoljuk végig az alábbi program kimenetét, aztán futtassuk a programot, hogy kiderüljön, jól tippeltünk-e.

```
1. állat = 'ló'
2. print(állat)
3. állat = 'nandu'
4. print(állat)
5. állat = 'cickány'
6. print(állat)
```

Szabály, hogy a Python változónevei

- betűvel vagy alávonással (_) kezdődhetnek;
- betűvel, számmal vagy alávonással folytatódhatnak (írásjel és szóköz nem lehet bennük), azaz `anyu kora` helyett használjuk az `anyu_kora` alakot (ez számít pythonosnak), vagy írjuk egybe a szavakat;
- a kis- és a nagybetű használatára figyelnek, azaz `Majom`, `majom` és `majoM` három külön változó;
- nem egyezhetnek meg az úgynevezett „foglalt szavakkal” – ilyen például a *for*, az *if*, vagy a *while*.

Nem szabály, de érdemes akként tekinteni rá: a programnak mindegy, hogy miként nevezzük el a változókat. Ha a fenti programban az „állat” helyett *mindenhol* „növény” szerepelne, a program hibátlanul működne. A *programozónak* fontos a jó változónév, hogy ha holnapután előveszi a programját, még mindig el tudjon igazodni rajta. A változónevek választásával a program értelmezését segítjük.

Adat bekérése a felhasználótól

A legtöbb program kér adatokat a felhasználótól. A telefonunkba be kell írni az új telefonszámot, vagy egy listából kiválasztani a már rögzítettet. A böngészőnkbe beírjuk, hogy melyik webhelyet nyissa meg. A gépünknek megadjuk a jelszavunkat.

Pythonban a felhasználótól az `input` utasítással kérhetünk adatot. Az utasítás legegyszerűbb formája az `input()`, így, két zárójellel. Írjunk be ennyit egy programba, és futtassuk le! Azt látjuk, hogy a program vár. Ha nyomkodjuk a billentyűket, akkor amit lenyomtunk, kiíródik, ha Entert nyomunk, a program futása befejeződik.

Ha a programunkat úgy módosítjuk, hogy a zárójelek között megadjuk, hogy mit kérdezünk a felhasználótól: `input('Hogy hívnak?')`, akkor ez kiíródik. A programunk azonban nem jegyzi meg, amit válaszolunk, mert nem mondtuk neki.

Így tudjuk erre „megkérni”:

```
1. név = input('Hogy hívnak?')
2. print(név)
```

Amit a felhasználó mond, azt betesszük egy változóba.

A print utasítás több dolgot is ki tud írni egymás után. Amit ki akarunk íratni, azt a zárójelen belül, vesszővel elválasztva kell felsorolnunk. Például: `print('Ezt', 'egymás mellé', 'írom.')`. Bármelyik szöveg helyett írható változó. Próbáljuk meg kiegészíteni a fenti programot úgy, hogy a Python nevünkön szólítva bennünket, köszönjön nekünk!

Változók, kiíratás, adat bekérése

Kérdések, feladatok

1. Mit írnak ki az alábbi programok? Gondoljuk végig, aztán próbáljuk ki a programokat, nézzük meg, hogy igazunk volt-e!

```
1. állat = 'ló'
2. ló = 'Ráró'
3. állat = 'macska'
4. print(állat)
```

```
1. gyümölcs = 'alma'
2. gyümölcs = 'körte'
3. alma = 'dinnye'
4. dinnye = 3
5. gyümölcs = alma
6. print(gyümölcs)
```

```
1. autó = 'Trabant'
2. autó = 'Renault'
3. autó = 3 - 5
4. print(autó)
```

2. Rajzoljuk ki az alábbi ábrát karakterek használatával!

A visszaper („\”, backslash) különleges szerepű, az utána lévő karakter úgynevezett vezérlőkarakter, és a Python értelmet tulajdonítana neki, nem kiírná. Ha azt akarjuk, hogy a visszaper kiíródjon, két visszapert kell írnunk – a `print('\\')` parancs csak egy visszapert ír ki.

3. Rajzoljuk ki az alábbi ábrákat karakterek használatával!

4. Kérdezzük meg a felhasználótól (a program használójától) a vezetéknevét! Kérdezzük meg a keresztnevét is, és köszönjünk neki, a teljes nevén szólítva!
A 3. sor teljesíti a feladatot, de picit csúnya: a felkiáltójel elkülönül az utolsó szótól, ami nem szép és nem is szabályos – mármint a helyesírás szabályai szerint. A 4. sor ezt oldja meg. A print utasítás alapértelmezés szerint szóközt tesz a vesszővel felsorolt kiírandók közé, ezt bíráljuk fölül a `sep=''` utasítással. A `sep` a separator (elválasztó) rövidítése, a két aposztrófunk között pedig semmi sincs: ne legyen elválasztó. Ha ezt beállítjuk, nekünk magunknak kell figyelnünk a szóközökre, ezért alakul át a negyedik sor többi része is.

```
1. vezeték = input('Mi az Ön becses vezetékneve?')
2. kereszt = input('Érdeklődhetek a keresztneve felől is?')
3. print('Üdvözlöm,', vezeték, kereszt, '!')
4. print('Üdvözlöm, ', vezeték, ' ', kereszt, '!', sep='')
```

Számok és karakterláncok a programunkban

Eddig még csak karakterláncot (szöveget) tároltunk a változóinkban. Ebben a leckében változtatunk ezen, és számokat is használunk.

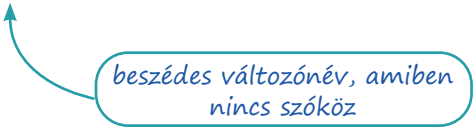
Hány éves a felhasználó?

Olyan programot fogunk írni, amely választ ad a fenti kérdésre. Az első részfeladat egy olyan program megírása, amely megkérdi, hogy mikor születtünk, és ezt ki is írja nekünk. Ez még nem tartalmaz új tudáselemet, úgyhogy készítsük el egyedül, majd olvassunk tovább!

A következő részfeladat a felhasználó korának meghatározása. Ehhez a programunknak tudnia kell, hogy melyik évben futtatják. A jelenlegi év megkérdezhető az operációs rendszertől, de egyelőre megelégszünk azzal, hogy változóként felvesszük a programunkba.

Azokat az értékeket, amelyek később nem változnak, konstansnak nevezzük, és vannak olyan programozási nyelvek, amelyeknél külön jelöljük. A Pythonban úgy *szokás*, hogy az ilyen értéket tároló változóknak csupa nagybetűs nevet adunk. A konstansnak tekintett változókat szokás a program elején megadni, azaz a programunk mostanra nagyjából így néz ki:

```
1.  IDEI_ÉV = 2021
2.  felhasználó_kora = input('Hány éves vagy?')
3.  print('Te most', felhasználó_kora, 'éves vagy.')
```



A negyedik sorunk alighanem egy kivonás lesz, például

```
születési_év = IDEI_ÉV - felhasználó_kora
```

vagy hasonló. Ha ebben az állapotban lefuttatjuk a programunkat, a kérdésre még megvárja a választ, de utána hibaüzenettel leáll.

Hányadik sorra vonatkozik a hibaüzenet?

A hibaüzenet az újonnan beírt sorban van, és átböngészve találunk benne olyat is, hogy `int` és `str`, előttük meg egy kivonásjel. Ez a három dolog a lényeg.

A Python azt igyekszik elmagyarázni, hogy nem tud egy egész számból (angolul: integer, röviden `int`) kivonni egy karakterláncot, más szóval szöveget (angolul: string, `str`).

Ha úgy gondoljuk, hogy itt valami tévedés lesz, mi rendes felhasználó módjára a programban feltett kérdésre számmal válaszoltunk, akkor igazunk van, de el kell fogadnunk, hogy a Python meg óvatos. Nem tudhatja, hogy amit válaszoltunk a kérdésére, azt biztosan számnak, tízes számrendszerbeli számnak gondoltuk. Ezért minden, amit az `input` a programnak átad, szöveg, azaz `str` marad. Ha 15-öt válaszoltunk az előző kérdésre, a program lát egy 1-es és egy 5-ös karaktert, de csak mint két egymás utáni karaktert, nem pedig egy számot.

Mik is azok a műveleti jelek?

Még iskoláskorunk legelején megtanultuk, hogy mik azok, de ha meg kell fogalmazni, esetleg elbizonytalanodunk. Végül talán arra gondolunk, hogy a műveleti jel olyan jel, ami a mellette álló adattal, adatokkal egy műveletet végez.
Futtassuk az alábbi egyszerű programot, és gondolkodjunk el a kimenetén!

```
1. print(10 + 3)
2. print('10' + '3')
3. print('Ej' + 'nye!'
4. print(10 * 3)
5. print(10 * '3')
6. print(10 * 'Abc')
```

Az aposztrófok közé írt szám NEM szám!

Fogalmazzuk meg a tanulságokat:

Az összeadásjel:

- két számot összead,
- két karakterláncot egymás mellé ír.

A szorzásjel:

- két számot összeszoroz,
- számot észlelve a karakterláncot egymás mellett a számnak megfelelően megismétli.

Az, hogy a műveleti jel pontosan milyen műveletet végez, *attól is függ, hogy az adat milyen* ***típusú****.* Ezért nem találgat a Python, hanem azt várja, hogy pontosan adjuk meg az adat típusát.

Típusátalakítás

Térjünk vissza a felhasználó születési évét kiíró programunkhoz. Azt már tudjuk, hogy az `input` utasítás karakterláncot ad vissza, és azt is, hogy nekünk számra van szükségünk. A típusátalakítást, típuskonverziót a Pythonban a céladattípus nevével megegyező utasításokkal végezzük el: az `int('2021')` utasítás eredménye 2021, számként. Ennek figyelembevételével programunk így alakul:

```
1. IDEI_ÉV = 2021
2. felhasználó_kora = input('Hány éves vagy? ')
3. print('Te most', felhasználó_kora, 'éves vagy.')
4. felhasználó_kora = int(felhasználó_kora)
5. születési_év = IDEI_ÉV - felhasználó_kora
6. print('Ekkor születtél: ', születési_év, '.', sep='')
```

„Kozmetikai” szóköz, hogy szebb legyen a kimenet.

A **típuskonverzió** a negyedik sorban van. Próbáljuk ki a kész programot!
Szeretnénk *pontosan* érteni, hogy mit csinál a típuskonverziót végző utasítássor, úgyhogy most mondjuk ki fennhangon, hogy mit csinálnak az alábbiak!

- szám = 2 + 4517
- majmok = orangutánok + cerkófok

Remélhetőleg nagyjából ezeket mondtuk:

- Adjuk össze a két számot, és az eredményt tegyük a „szám” változóba!
- Olvassuk ki az orangutánok és a cerkófok változó tartalmát, és az eredményt tegyük a „majmok” változóba!

Ha megfigyeljük a mondatainkat, látjuk, hogy előbb foglalkozunk a fenti sorok egyenlőségjeltől jobbra álló oldalával, és csak utána a bal oldallal.

Eddig három műveleti jelünk (operátorunk) volt, a „+”, a „–” és a „*” jel, de mostanra el kell fogadnunk, hogy programozáskor az egyenlőségjel is műveleti jel. Alapvetően mást jelent ilyenkor az egyenlőségjel, mint matematikaórán. Ott állításokat, kijelentéseket fogalmaztunk meg vele (Kettő egyenlő: háromból egy. Hatszor hat egyenlő harminchattal.). Programozáskor az egyenlőségjel egy művelet elvégzésére való *felszólítás*, nézzük csak meg az előbbi számos és majmos mondatot!

Programíráskor az egyenlőségjel az **értékadás** műveletének műveleti jele, olvashatjuk „legyen egyenlő” formában. Értékadáskor mindig az egyenlőségjel jobb oldalát értékeli ki a program elsőként, majd a kapott eredményt értékül adja az egyenlőségjel bal oldalán lévő változónak.

A `felhasználó_kora = int(felhasználó_kora)` sor tehát a következőt jelenti:

1. Olvassuk ki a `felhasználó_kora` változó tartalmát (ez az egyenlőségjel jobb oldalán lévő `felhasználó_kora`)!
2. A kapott értéket adjuk át az `int` utasításnak (ami majd egész számmá alakítja)!
3. Az `int` utasítás eredményét adjuk értékül a `felhasználó_kora` változónak (felülírva ezzel a régi értéket)!

Szemléletes, ha úgy képzeljük el, hogy a dobozból kivesszük a benne lévő szöveget, átgyúrjuk számmá, aztán visszatesszük ugyanabba a dobozba.

És amit nem lehet átalakítani számmá?

Abból bizony hibaüzenet lesz.
Nézzük a programunk alábbi két futtatását!

```
C:\Users\raerek\programjaim>hanyeves.py
Hány éves vagy?
Te most  éves vagy.
Traceback (most recent call last):
  File "C:\Users\raerek\programjaim\hanyeves.py", line 4, in <module>
    felhasználó_kora = int(felhasználó_kora)
ValueError: invalid literal for int() with base 10: ''

C:\Users\raerek\programjaim>hanyeves.py
Hány éves vagy? csillijómillijó
Te most csillijómillijó éves vagy.
Traceback (most recent call last):
  File "C:\Users\raerek\programjaim\hanyeves.py", line 4, in <module>
    felhasználó_kora = int(felhasználó_kora)
ValueError: invalid literal for int() with base 10: 'csillijómillijó'
```

Nem adunk meg értéket, csak Entert nyomunk.

A semmit a Python nem tudja számmá alakítani.

A szöveget sem.

Természetesen kezelhetők az ilyen jellegű hibák, csak mi még nem tanultuk meg a módját. Még sokáig abból indulunk ki programkészítéskor, hogy a felhasználótól érkező bemenetet nem kell ellenőriznünk, validálnunk. Egy „igazi” alkalmazás esetében a hibákra való felkészülés (felhasználótól kapott rossz bemenet, elfogyó háttértár, megszakadó hálózati kapcsolat stb.) a programnak igen jelentős része.

Karakterlánccá alakítás

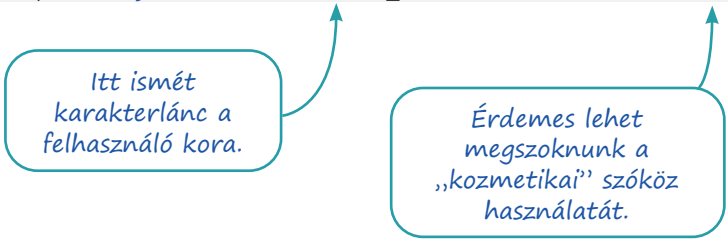
Láttuk már, hogy két karakterlánc összeadható, és azt is láttuk, hogy a `print` utasításnak több kiírnivaló is átadható, és ezeket vesszővel választjuk el. De lehet olyat írni, hogy `print('alma'+ ' ' + 'körte' + ' ' + 'dió')`? Hogyne! Ilyenkor előbb „összeadódnak” a karakterláncok, és ezt követően egyetlen karakterláncot kap meg a `print`. Persze itt éppen megvagyunk e módszer nélkül, mert a vesszővel való felsorolás remekül működik (lásd a programunk 6. sorát), de van, ahol ez probléma.

Egészítsük ki a programunkat: kérdezze meg, hogy „És milyen N évesnek lenni?”, ahol N természetesen a felhasználó korábbi válaszában szereplő szám!

Logikusnak tűnik egy `ilyen = input('És milyen', felhasználó_kora, 'évesnek lenni?')` megoldás, de sajnos az `input` nem ismeri a `print` vesszős összefűzési módszerét. Ha a „+” operátort használjuk, akkor egy másféle hibába csöppenünk, de azért próbáljuk csak ki, könnyű lesz korrigálni!

A hibaüzenet ismét `int`-ről és `str`-ről beszél, és ekkor már gyanítjuk, hogy az lehet a baj, hogy a `felhasználó_kora` a 4. sor óta egész szám, amit nem tud a Python „összeadni” egy karakterlánccal. Amikor egész számmá akartunk alakítani, az `int` utasítást használtuk. Amikor karakterlánccá alakítunk, az `str` utasításra van szükség. Programunk utolsó két sora ezt a formát ölti:

```
7. felhasználó_kora = str(felhasználó_kora)
8. ilyen = input('És milyen ' + felhasználó_kora + ' évesnek lenni? ')
```



Számok és karakterláncok

Feladatok

1. Mit írnak ki az alábbi programok?

```
1. szám = 4
2. szám = szám + 2
3. print(szám)
```

```
1. edény = 'bögre'
2. edény = 'sárga' + edény + 'görbe' + edény
3. print(edény)
```

Természetesen tudunk tizedestörtekkel is dolgozni programjainkban. A tizedestörteket a programozók lebegőpontos számnak is nevezik, ami angolul *floating point number*, innen származik a típus neve: `float`. A megfelelő típuskonverziót a `float` parancs hajtja végre (használni úgy kell, mint az `int` és az `str` utasítást). A tizedesvessző helyett tizedespontot használunk.

2. Kérjünk be egy kilométerben mért távolságadatot a felhasználótól, és írjuk ki tengeri mérföldre átváltva! (Egy tengeri mérföld 1852 méter.)
   Ha elkészültünk, megírhatjuk a feladat megfordítását.
3. Kérjünk be a felhasználótól két számot, tároljuk őket egy-egy változóban!
   a. Adjuk össze őket, és írjuk ki az eredményt!
   b. Írjuk ki az eredmény elé, hogy „Az összegük:”!
   c. Írjuk ki magukat a számokat is! Ha például 2-t és 3-at adott meg a felhasználó, akkor a kimenet legyen ilyen:
   `2 és 3 összege: 5.`
4. Írjuk át a programot szorzásra! (A szorzás jele a *, az osztásé a /, a hatványozásé a **. A gyökvonásnak nincs külön jele, de tört kitevővel megoldható.)
5. Vegyük elő az előző gyakorló feladatsor 4. feladatának megoldását. Ebben a feladatban már megoldottuk a vezeték- és a keresztnév kiírását. Bővítsük úgy a programot, hogy kérdezze meg a születési évünket is, és írja ki a nevünkkel egy sorban: `Kék Blamázs, 2011.` A születésiév-megállapító programunkkal egybegyúrva készíthetünk olyan programot is, amelyik megkérdezi a neveinket, a születési évünket, és a bulvárcikkekben szokásos formában, a korunkkal együtt írja ki a nevünket: `Fehér Karnis (21)`.
6. Adott óra, perc, másodperc hármassal megadott időt váltsunk át másodpercre! (Az adatokat nem kell mindenképp a felhasználótól kérni, beírhatjuk őket a programba is.) Sikeres megoldás után készítsük el a feladat megfordítottját!
7. Kihívást jelentő feladat: Milyen szöget zár be egymással a kis és a nagy mutató adott időpontban?

Az egyik utasításnak közvetlenül is átadhatjuk a másiktól visszakapott értéket, így nem kell külön sorba írnunk a típuskonverziót:

8. Milyen típusú adat lesz a „szám” változóban? Lefut-e rendben a második sor?

```
1. szám = int(input('Hányat ugrik a nyuszi? '))
2. print('A nyuszi ' + str(szám) + '-at ugrik.')? ')
```

Elágazások

Eddig csupa olyan programot írtunk, ami elkezdődött az elején, sorban egymás után végrehajtott minden utasítást, aztán kilépett. Ebben a leckében ezen változtatni fogunk.

Gondolunk egy számra

A témakör első leckéjében már láttunk egy olyan programot, amelyikben elágazás van. Az a program mást csinál, ha nem vagyunk még tizennégy évesek, és mást, ha már betöltöttük ezt az életkort. Hasonló működésű az ugyanott megismert „Mi a neve Mátyás királynak?” program.

Az alábbi **folyamatábra** is egy hasonló programot ír le. Nincsenek benne konkrét utasítások, mert fontosabb, hogy gondold először át, hogy mit csinál a programod, és ráérsz utána azon elmélkedni, hogy miként **kódolod** a programodat.

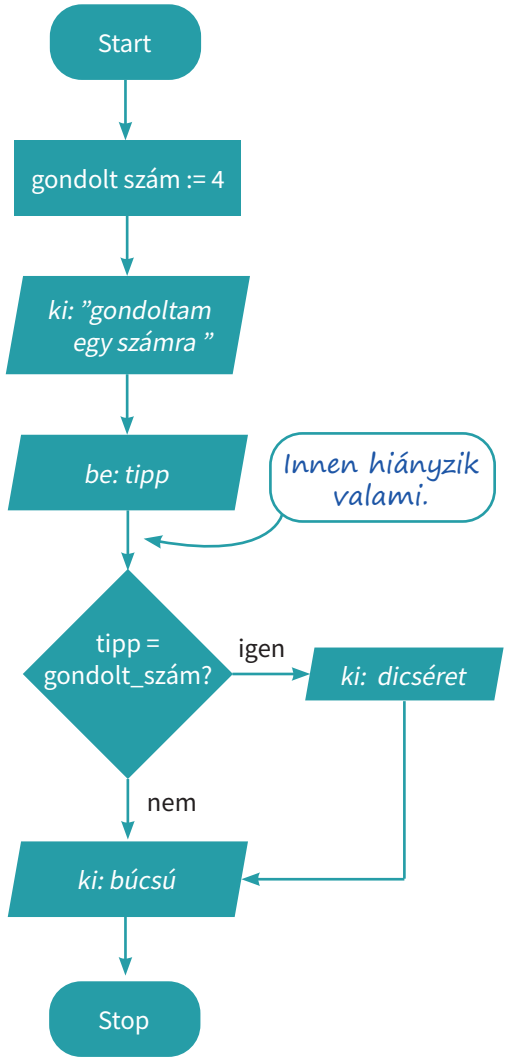
1. Mit csinál a program?
2. Mi az a lépés, amit nem tüntettünk fel? (Ha most nem jövünk rá, nem probléma: hamarosan úgyis megírjuk a programkódot, akkor szólni fog a Python.)
3. Hogyan olvassuk ki a := jelet? Mi a megfelelője Pythonban?

Miközben a programunkat folyamatábrával megfogalmazzuk, **algoritmust** (receptet) adunk a bennünket érdeklő probléma megoldására. A folyamatábrák elég látványosak, de hamar lelógnának a könyvlapról, így aztán a gyakorlatban gyakrabban használunk egy másik algoritmusleíró eszközt, a **mondatszerű leírást.**

A leírás szabályaira rá fogunk érezni.

4. Vessük össze ezt a leírást a folyamatábrával!
5. Melyik az a művelet, ami a mondatszerű leírásban már megvan, de a folyamatábrában még hiányzik?

```
gondolt_szám := 4
ki: „gondoltam egy számra”
be: tipp
tipp átalakítása egésszé
elágazás
ha tipp = gondolt_szám:
      ki: dicséret
elágazás vége
ki: búcsú
```



Készítsük el a fenti két algoritmusleíró eszközzel megadott programkódot!

```
1. gondolt_szám = 4
2. tipp = input('Gondoltam egy számra. Tippeld meg! ')
3. tipp = int(tipp)
4. if tipp == gondolt_szám:
5.     print('Ügyes!')
6. print('Pápá.')
```

A kettőspont hatására a következő sor bentebb kezdődik.

Ez tényleg 2 egyenlőségjel.

Egy TAB vagy 4 szóköz KELL!!!

Teszteljük a programunkat: adjuk meg a helyes megoldást, de próbáljuk ki helytelennel is!

A program következő változatában, más szóval verziójában kicsit bővebben dicsérünk, illetve a hibásan tippelő felhasználókat kicsit ugratjuk. A mondatszerű leírás a következő:

6. Módosítsuk ez alapján a folyamatábrát (segítség: a rombuszból lefelé nem lesz nyíl, de balra igen, és a két nyíl a rombusz alatt összetalálkozik)!

```
gondolt_szám := 4
ki: „gondoltam egy számra”
be: tipp
tipp átalakítása egésszé
elágazás
ha tipp = gondolt_szám:
      ki: kétsoros dicséret
különben:
      ki: ugratás
elágazás vége
ki: búcsú
```

A dicséret második sora újabb print utasítást jelent az előző alatt. Írjuk be az új sort:

```
print('Gratulálok.')
```

behúzva és behúzás nélkül! Mindkét esetben teszteljük a programunkat, és vessük össze a viselkedését. Fogalmazzuk meg a behúzás szerepét!

Ha megvagyunk, írjuk meg az ugratós részt is. A „különben” szó angolul „else” – ezt kell használnunk kódoláskor. A kész program:

Ez itt az elágazás FELTÉTELE.

```
1. gondolt_szám = 4
2. tipp = input('Gondoltam egy számra. Tippeld meg! ')
3. tipp = int(tipp)
4. if tipp == gondolt_szám:
5.     print('Ügyes!')
6.     print('Gratulálok.')
7. else:
8.     print('Hosszan gondolkodtál rajta?:)')
9.     print('Nem érte meg.;)')
10. print('Pápá.'))
```

Ez a két utasítás a ,,ha''-ág.

Ez a két utasítás pedig a ,,különben''-ág.

A programunk tanulságai:

1. Ami az `if` után következik, az az elágazás feltétele.
2. A feltételvizsgálatban két egyenlőségjel kell. A programozásban az egy egyenlőségjel egy felszólítás (ismerjük már, értékadáskor használjuk), a két egyenlőségjel kérdés: A tipp egyenlő a gondolt számmal?
3. Ami az elágazás ágaiban van (a fenti program 5–6. és 8–9. sora), az bentebb kezdődik. A Python onnan tudja, hogy mikor kezdődik egy ág, hogy a programsor bentebb kezdődik, és onnan tudja, hogy hol van vége az ágnak, hogy az újabb programsor már nem kezdődik bentebb.

Összetett feltétel

Szeretnénk úgy bővíteni a programunkat, hogy ha csak eggyel tippelt mellé a felhasználó, akkor ezt eláruljuk neki. Elsőként az algoritmus mondatszerű leírását módosítjuk. Az új, „különbenha” ágat megvalósító Python-utasítás az `elif`.

Akár neki is foghatnánk a kódolásnak, de hogy programozandó a „csak egyet tévedett”? Ilyen utasítás nincs, ezért az algoritmusunkat egy-két lépéssel tovább kell finomítanunk. Észrevesszük, hogy kétféleképp lehet egyet tévedni: vagy eggyel nagyobbat, vagy eggyel kisebbet tippelve. Az „eggyel nagyobbat tippelt” így írható le:

```
tipp = gondolt_szám+1.
```

```
gondolt_szám := 4
ki: „gondoltam egy számra”
be: tipp
tipp átalakítása egésszé
elágazás
ha tipp = gondolt_szám:
      ki: kétsoros dicséret
különbenha csak egyet
tévedett:
      ki: csak egyet tévedtél
különben:
      ki: ugratás
elágazás vége
ki: búcsú
```

A másik feltétel megfogalmazása nem okozhat gondot, de ezek szerint két feltétel lett az egyből. Megírhatjuk úgy az algoritmust (és a programot), hogy két „különbenha” ágat adunk meg, ugyanazzal a kiírandó üzenettel, de ez nem szerencsés – például azért, mert ha módosítani kell az üzenetet, két helyen is módosítanunk kell, és az egyiket előbb-utóbb elfelejtjük megcsinálni.

Alakítsuk inkább a két feltételünket egy összetett feltétellé:

```
különbenha tipp = gondolt_szám+1 vagy tipp = gondolt_szám-1:
```

A két feltételből így lett egy. Lévén a „vagy” szó kapcsolja őket össze, elég, ha az egyik teljesül. Ha „és” kapcsolná őket össze, mindkettőnek teljesülnie kellene, hogy a teljes összetett feltétel igaz legyen. A kód most így néz ki:

```
1. gondolt_szám = 4
2. tipp = input('Gondoltam egy számra. Tippeld meg! ')
3. tipp = int(tipp)
4. if tipp == gondolt_szám:
5.     print('Ügyes!')
6.     print('Gratulálok.')
7. elif tipp == gondolt_szám + 1 or tipp == gondolt_szám - 1:
8.     print('Ó' csak eggyel tévedtél.')
9. else:
10.    print('Hosszan gondolkodtál rajta?:)')
11.    print('Nem érte meg.;)')
12. print('Pápá.')
```

Kérdések

1. Hány `elif`-ág és hány `else`-ág szerepelhet egy elágazásban?
2. Melyiket kell utolsóként megadni?
3. Melyiknek nincs feltétele?
4. Létezhet olyan elágazás, amelyikben nincs egyik sem?
5. Kihívást jelentő feladat: Az alábbi sor nem jó (bár a program nem jelez hibát): `elif tipp == gondolt_szám + 1 or gondolt_szám – 1:` Miért nem jó?

Véletlenszám-előállítás

Meglehetősen unalmas lehet, hogy a programunk mindig a négyre gondol. A legtöbb programozási nyelvben van valamilyen módszer véletlen számok előállítására. A Pythonban több is van, ezek közül a `random.randint` (randint: random integer, azaz véletlen egész) az, amelyik véletlenszerű egész számot állít elő, más szóval generál. Ha ezt az utasítást használni akarjuk, akkor előbb be kell töltetni a programmal a `random` nevű modult. A programunk első sorai a következőképp alakulnak:

```
1. import random
2.
3. gondolt_szám = random.randint(1,6)
4. print('Súgok:', gondolt_szám)
5. tipp = input('Gondoltam egy számra. Tippeld meg! ')
```

Itt töltjük be a „random” modult. Az importálásokat követően szokás egy sort kihagyni.

1 és 6 közötti egész számot állítunk elő.

A súgás a program tesztelésekor hasznos, a végső változatból vegyük ki!

Feladatok

Az „==” operátor nemcsak számok, hanem szövegek egyezésének vizsgálatára is használható.

1. Kérjünk be jelszót a felhasználótól, és hasonlítsuk össze a programban tárolttal! Ha a felhasználó eltalálta a jelszót, írjuk ki, hogy „Helyes jelszó.”, különben „Hozzáférés megtagadva.”.

A feltételek megfogalmazásakor használható a „>”, a „<”, a „>=” és a „<=” operátor is. Azt, hogy „nem egyenlő”, a „!=” operátorral fejezzük ki, a matematikában megszokott áthúzott egyenlőségjelünk nincs.

2. Kérjünk be két számot a felhasználótól! Írjuk ki a nagyobbat!
3. Állítsunk elő két véletlen számot, és kérdezzük meg a felhasználótól az összegüket! Ha helyesen válaszol, dicsérjük meg!

```
1.  import random
2.
3.  egyik = random.randint(1,10)
4.  másik = random.randint(1,10)
5.  egyik_szövegként = str(egyik)
6.  másik_szövegként = str(másik)
7.  tipp = input('Mennyi ' + egyik_szövegként + ' és ' + másik_szöveg-
        ként + ' összege? ')
8.  tipp = int(tipp)
9.  összeg = egyik + másik
10. if tipp == összeg:
11.     print('Valóban annyi, ez igen!')
```

4. Írjunk olyan programot, amelyik bekér két csapatnevet és két pontszámot, majd kiírja a mérkőzés eredményét (a felhasználó válaszai vastagabbal szedve):

```
Mi az egyik csapat neve? Tóparti királyok
Hány pontot szerzett? 78
Mi a másik csapat neve? Talpasi csodatévők
Hány pontot szerzett? 54
Az összecsapás eredménye:
Tóparti királyok - Talpasi csodatévők
78 : 54
Tóparti királyok nyert.
```

5. Vegyük elő azt a programunkat, amelyik a felhasználó nevét és korát kezeli. A felhasználónak javasoljunk életkorának megfelelő olvasnivalót!
   a. 0–3 év: „Totyogóknak a kettes számrendszerről”
   b. 4–6 év: „Hackeljük meg az óvodát!”
   c. 7–14 év: „Felhőtechnológia a menzán”
   d. 15–18 év: „Big data a középiskolában”

6. Ha bonyolultabb összetett feltételeket fogalmazunk meg, érdemes lehet zárójelek használatával egyértelműsíteni a szándékunkat. Melyik feltételmegfogalmazás helyes az alábbiak közül?
   a. Ha (van tollunk és van ceruzánk) vagy van egy papírlapunk:
      tudunk rajzolni valamit.
   b. Ha (van tollunk vagy van ceruzánk) és van egy papírlapunk:
      tudunk rajzolni valamit.
   c. Ha van tollunk vagy (van ceruzánk és van egy papírlapunk):
      tudunk rajzolni valamit.
7. Kihívást jelentő feladat: A kistesód szülinapi banános nyaflatyát csinálod. A kistesód szerint a banános nyaflaty akkor jó, ha:
   a. nincs benne egyszerre vaníliás cukor és tortareszelék,
   b. ha van benne fahéj, akkor kell rá tejszínhab is,
   c. ha nincs benne fahéj, nem kerülhet rá tejszínhab sem.

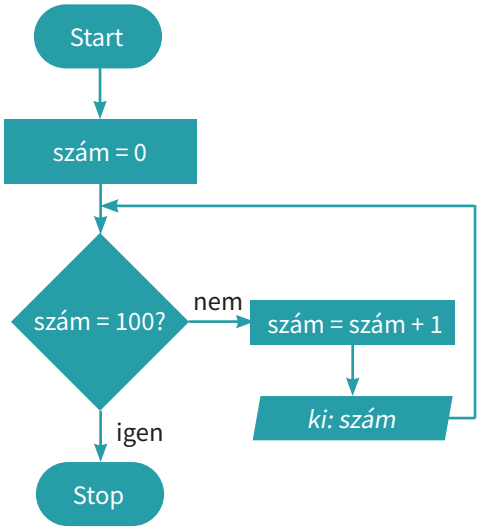
   Írj programot a nyaflaty jóságának eldöntésére!

Ciklusok

A ciklus eredetileg valamilyen ismétlődő dolgot jelent, gondolhatunk holdciklusra, választási ciklusra, árapályciklusra. Mi ebben a könyvben egy olyan *programrészletet* értünk rajta, amely valahányszor megismétlődik. Sok elterjedt programozási nyelvben a ciklus lehet számlálós vagy feltételes, de a Pythonban e kettő közül csak az utóbbi létezik, cserébe van még bejárós ciklusunk.

A feltételes ciklus (while-ciklus)

Elsőként megvizsgáljuk ezt a folyamatábrát. Vajon mit csinál a program?



Mondatszerű leírással így néz ki az algoritmusunk:

```
szám := 0
ciklus amíg szám != 100:
   szám = szám + 1
   ki: szám
ciklus vége
```

A != azt jelenti, hogy: „NEM egyenlő”

Python nyelven pedig az alábbi formát ölti programunk:

```
1. szám = 0
2. while szám != 100:
3.     szám = szám + 1
4.     print(szám)
```

A „while” annyit tesz: amíg.

„Amíg” a feltétel teljesül, addig újra meg újra belépünk a ciklusba.

Behúzás, mint az if-nél.

Ez a két sor „benne van a ciklusban” – ők a CIKLUSMAG.

1. Mi történik, ha a ciklusmag első sorát elhagyjuk? (Ha sikerült végtelen ciklusba kerülnünk, a program a Ctrl + C [C jelentheti azt, hogy Cancel, jelentése töröl, érvénytelenít] billentyűkombinációval megállítható.)
2. A != helyett milyen feltétellel érhetjük el ugyanezt az eredményt?
3. Hogyan változik a program kimenete, ha a ciklusmag két sorát felcseréljük? Ha a felcserélt sorokkal is szeretnénk a számokat 1-től 100-ig kiírni, mit kell még módosítani a programon?
4. Hogyan íratható ki minden második szám 100-ig? Hogyan íratható ki minden harmadik szám 100-ig?

Következő órára leírod százszor, hogy…

Bizonyára sokaknak viccekből, régi történetekből ismerős az alcímben jelzett tanári büntetés. Jelen sorok írójának azonban még a valóság volt: le kellett írnia százszor, hogy „A tornaterembe sapkában megyek át”. A fenti program egyetlen sorának módosításával megoldható ez a feladat, mi most mégis három módosítást is teszünk.

```
1. számláló = 0
2. while számláló != 100:
3.     számláló += 1
4.     print('Tudom, sapka.')
```

A változót átnevezzük a szerepének megfelelően. Amikor a változó értékét nem igazán használjuk a ciklusban, Pythonban szokás alávonásnak (,,_'') elnevezni a változót. Próbáld ki!

Ez ugyanaz, mint a számláló = számláló + 1 Használd a neked jobban tetszőt!

Túltenni Gausson

A kis Gauss, amikor még nem volt nagy matematikus, az anekdota szerint egész osztályával együtt azt a feladatot kapta a tanárától, hogy adja össze a számokat egytől százig. A tanár közben nekiállt valami más munkának, de a kis Gauss két perc múlva szólt, hogy készen van, és az eredmény 5050. Rájött, hogy 1 + 100 = 101, 2 + 99 = 101, és így tovább. 50-szer 101 pedig 5050. Ha már van számítógépünk, illendő a két percnél jobb eredményt hoznunk, méghozzá Gauss felismerésének kihasználása nélkül.

```
1. számláló = 0
2. összeg = 0
3. while számláló < 100:
4.     számláló += 1
5.     összeg = összeg + számláló
6. print('Összesen:', összeg)
```

Ezúttal két változó is kelleni fog, mind a kettőnek kell adnunk kezdeti értéket.

Figyeljünk a ciklus feltételének változására!

Ez már nincs behúzva: nincs a ciklusmagban.

A logikai adattípus

Vegyük elő a számkitalálós programunkat, és csupaszítsuk le annyira, hogy csak a jó válaszra reagáljon! Tervezzük át úgy, hogy kérdezzen addig, amíg ki nem találjuk a számot! A mondatszerű leírásba nem írjuk bele az importálást végző utasítást:

```
gondolt_szám := 1 és 6 közötti véletlen szám
kitalálta = hamis
ciklus amíg ki nem találta:
      be: tipp
      ha tipp = gondolt_szám:
            kitalálta = igaz
ciklus vége
```

Látjuk, hogy bevezetünk egy változót, amiben igaz vagy hamis érték tárolható. A változó kezdeti értéke hamis, és akkor állítjuk át igazra, ha a tipp jó volt. Minthogy a változót hamis értékkel inicializáltuk, a ciklusba legalább egyszer belépünk, és pont ezt akarjuk.

Azok a változók, amelyek igaz (True) és hamis (False) értéket vehetnek fel, úgynevezett logikai típusúak (a Pythonban a típus neve `bool`, George Boole matematikus után, aki efféle problémákkal való foglalatosságáról vált híressé).

Mindez kódként így néz ki:

```
1. import random
2.
3. gondolt_szám = random.randint(1,6)
4. kitalálta = False
5. while not kitalálta:
6.     tipp = int(input('Szerinted? '))
7.     if tipp == gondolt_szám:
8.         kitalálta = True
```

False és True, nagy kezdőbetűvel!

Lehetne while kitalálta == False: de így jobban olvasható a kód, pythonosabb.

Mélyebb behúzás az if miatt.

Összetett ciklusfeltétel

A program türelmes, és így persze a felhasználó a végtelenségig próbálkozhat. Írjuk át a programunkat úgy, hogy csak három próbálkozást engedjen! Számolnunk kell a próbálkozásokat, de nem elég, ha a ciklus feltételeként csak azt adjuk meg, hogy még nem használtuk el mind a három próbálkozást. Arra is figyelnünk kell, ha közben a felhasználó kitalálta a gondolt számot.

Szerencsére a `while`, pont ugyanúgy, mint az `if`, képes összetett feltételek kezelésére.

5. Hogyan tartjuk nyilván az elhasznált lehetőségeket?
6. Fogalmazzuk meg a `while` feltételét!

A teljes kód a következő:

```
1. import random
2.
3. gondolt_szám = random.randint(1,6)
4. kitalálta = False
5. számláló = 0
6. while not kitalálta and számláló < 3:
7.     tipp = int(input('Szerinted? '))
8.     számláló += 1
9.     if tipp == gondolt_szám:
10.        kitalálta = True
```

7. Helyezzünk el ismét olyan programsorokat a kódban, amelyek a felhasználó dicséretéért, ugratásáért felelnek! Több helyre is elhelyezhetőek, és mindnek megvannak az előnyei és hátrányai. Hasonlítsunk össze néhány lehetséges megoldást!
8. Szeretnénk megoldani, hogy ha a felhasználó a harmadik lehetőségre már majdnem kitalálta a megoldást (csak egyet tévedett), akkor kapjon még egy esélyt. Ahogy programozáskor mindig, ismét több helyes megoldás lehetséges – valósítsuk meg a nekünk legjobban tetszőt!

Ciklusok és véletlenek

Feladatok

1. Ciklussal meg tudunk oldani egyenleteket az egész számok halmazán – próbálgatással.
   a. Oldjuk meg a $3x + 2 = 59$ egyenletet a pozitív egészek halmazán!

```
1. x=1
2. while 3*x + 2 != 59:
3.     x = x + 1
4. print('A megoldás:', x)
```

   b. Oldjuk meg a $6x^2 + 3x + 8 = 767$ egyenletet az egész számok halmazán! Honnan érdemes indítani a próbálgatást?
   c. Diophantosz ókori görög matematikus verses sírfelirata több fordításban fellelhető az interneten. A felirat alapján fogalmazz meg egyenletet, és írj programot, ami megoldja! Hány évesen halt meg Diophantosz?
   d. Ilyen módszerrel csak egész gyökű egyenlet oldható meg biztosan. Miért?
   e. Ha az egyenletnek nincs egész gyöke (például $6x^2 + 3x + 8 = 768$), akkor végtelen ciklusba kerül a programunk. Miként biztosítható, hogy ne lépjen végtelen ciklusba a program, és ha már esélytelen, hogy talál megoldást, ne próbálkozzon tovább? Gondolkozzunk összetett feltételben!
2. Írjunk pénzfeldobás-szimulátort!
   a. A gép írja ki, hogy fejet vagy írást „dobott”! Használhatjuk a `random.randint` utasítást is, de van más megoldás is.

```
1. import random
2.
3. dobás = random.randint(1,2)
4. if dobás == 1:
5.     print('fej')
6. else:
7.     print('írás')
```

```
1. import random
2.
3. dobás = random.choice(['fej', 'írás'])
4. print(dobás)
```

   b. Készítsünk statisztikát! Egymillió feldobásból mennyi lesz fej, és mennyi írás?
   c. Írjuk át a programot úgy, hogy kockadobásokat számoljon!
   d. Készítsünk cinkelt kockát! A hatos jöjjön ki kétszer akkora eséllyel, mint a többi szám!
3. Írjunk randiszimulátort!
   a. A számítógép kérdezze azt, hogy „Szeretsz?”, amíg azt nem válaszoljuk, hogy „Nagyon!”, vagy azt, hogy „Jobban, mint a kókuszgolyót!”.
   b. A számítógép legyen durcás: legfeljebb három kérdés után zavarjon el bennünket, ha nem adjuk meg a „helyes” válaszok valamelyikét!
   c. A számítógép legyen szeszélyes: zavarjon el bennünket 2-4 „rossz” válasz után véletlenszerűen!

Ciklusok oda-vissza és egymásba ágyazva

Feladatok

1. Írjuk ki a számokat csökkenő sorrendben 100 és –100 között!

A `print` utasítás mindig új sort kezd a kiírnivalót követően. Van ugyanis egy `end` nevű paramétere, ami alapértelmezetten, azaz ha mást nem adunk meg: `'\n'`. Egy visszaper és egy `n` betű. A visszaper (backslash) szól, hogy a következő betű nem is betű, hanem vezérlőkarakter, az `n` pedig azt jelenti: new line, új sor. Ezt azonban felülbírálhatjuk.

2. Írjunk ki egy számsort, aztán vizsgálódjunk még egy keveset!
   a. Írjuk ki (két) ciklussal, hogy „12345678987654321”!

```
1. szám = 1
2. while szám < 9:
3.     print(szám, end='')
4.     szám += 1
5. while szám > 0:
6.     print(szám, end='')
7.     szám -= 1
8. print('')
```

A sor végén itt nem '\n' van, és nem is szóköz, hanem egy nagy semmi. Azaz: NE KEZDJ ÚJ SORT!

Itt viszont a nagy semmit írjuk ki, aminek a végén ott van a '\n' – azaz az utoljára kiírt egyes után kérünk egy sortörést.

   b. Hogyan változik a programunk, ha elhagyjuk az utolsó sort?
   c. Hány helyen kell módosítanunk a programot, hogy ne kilencig, hanem kilencmillió-kilencszázkilencvenkilencezer-kilencszázkilencvenkilencig írja a számokat?
   d. Hogyan változik a program futásának sebessége, ha nem íratjuk ki a számokat, csak elszámoltatunk oda-vissza? (Írjunk a program harmadik és hatodik sorának az elejére kettős keresztet (#)! Így ezeket a sorokat megjegyzéssé alakítottuk, nem hajtódnak végre.)
3. Írjunk egymás mellé 10 csillagot („*”) úgy, hogy a programkódban csak egyetlen csillag karakter legyen!
4. Csillagok több sorban
   a. Írjunk egymás alá öt csillagsort! Ötlet: tegyük az előző feladat ciklusát egy másik ciklus belsejébe, figyelve a behúzásokra!

```
1. sorszámláló = 1
2. while sorszámláló <= 5:
3.     csillagszámláló = 1
4.     while csillagszámláló <= 10:
5.         print('*', end='')
6.         csillagszámláló += 1
7.     print('')
8.     sorszámláló += 1
```

A két ciklusnak két külön számlálója van.

Minden sor elején újrakezdjük a csillagok számolását.

Ez a három programsor ír ki egy sornyi csillagot.

Minden sornyi csillag után lezárjuk a sort, és újat kezdünk.

A belső ciklus magja nagyobb behúzást kap.

b. Az előző feladat megoldásának egyetlen helyen való megváltoztatásával alakítsuk háromszöggé a program kimenetét! (Ötlet: egy sorban annyi csillagot kell kiírni, ahányadik…)

```
*
**
***
****
*****
```

c. Ha elkészültünk, írjuk át úgy a programot, hogy minden sorban csak az utolsó csillag jelenjen meg!
d. Minden sorban az első és az utolsó csillag jelenjen meg!

5. Írjunk szorzótáblát a kicsiknek! A várt kimenet:

```
1 * 1 = 1
1 * 2 = 2
…
6 * 6 = 36
6 * 7 = 42
…
10 * 9 = 90
10 * 10 = 100
```

Összetartozó adatok kezelése

A lista adattípus

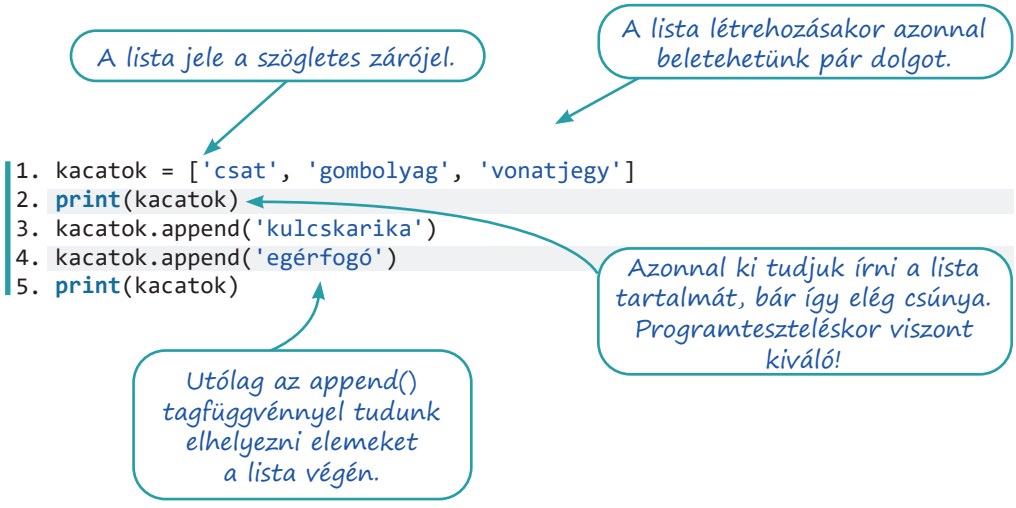
A programjainkban az adatokat változókban tároljuk, és eddig négy változótípust, adattípust ismerünk:

- a karakterláncot (szöveg, string, `str`),
- az egész számokat (integer, `int`),
- a valós, tizedestörtként megjelenő számokat (lebegőpontos szám, `float`)
- és a logikai értékeket (`True` és `False`, azaz a `bool` típus).

Ezek közül egyik sem jó akkor, amikor több, egymáshoz tartozó adatot szeretnénk tárolni, kiírni, műveletet végezni velük. Például tárolni szeretnénk az osztályunkba járók neveit. Beleírhatjuk egyetlen karakterláncba őket, de aztán hogyan írjuk ki őket egyesével? Vagy ábécérendben? Miként tároljuk az osztálylétszámokat, a kontinensek méreteit, az ország településeinek lakosságát, azt, hogy a majonézek hány grammosak és mennyibe kerülnek, a kedvenc sorozatunk szereplőit, a szomszéd kiskutyáinak neveit?

Mindezt megoldja egy **összetett adattípus** használata. Ilyen például a Python **lista** adattípusa.

Tároljuk kacatjainkat listában!



A Python listája megtartja az adatok sorrendjét, azaz mindig 'csat', 'gombolyag', 'vonatjegy' sorrendben kapjuk vissza az adatokat és nem másképp.

Nem kell egyforma típusú adatokat tenni egy listába, azaz teljesen jó a

```
csata = ['Isaszeg', 1849, 'április', 6, 'magyarok']
```

listamegadás, egy listában a csata helyszínével (szövegként), évével (egészként), hónapnevével (szövegként) és napjával (egészként), valamint a győztes oldallal (szövegként).

A listaelemek sorszámozása

Az adatok sorszámot is kapnak, mégpedig nullától kezdődően. A nullától való számozás a számítógépek világában nem szokatlan. Szokjuk meg mi is, hogy az előző listának öt eleme van, de az utolsó a negyedik sorszámú. Így az előző lista nulladik eleme `'Isaszeg'`, a harmadik pedig 6. A sorszámra a programunkban így hivatkozunk:

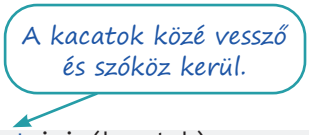
- a lista nulladik eleme: `csata[0]`
- a lista második eleme: `csata[2]`
- a lista második és az utáni elemei: `csata[2:]`
- a lista másodikat *megelőző* elemei: `csata[:2]`
- a lista másodiktól a negyediket *megelőzőig* terjedő (tehát a második és a harmadik) elemei: `csata[2:4]`
- a lista utolsó eleme: `csata[-1]`

Akiben felmerül a kérdés, hogy „És hogyan tárolnám a tavaszi hadjárat összes csatáját egyetlen listában?”, azt megnyugtatjuk, hogy lehet egy lista eleme egy másiknak. Aki ettől megijed, azt is megnyugtatjuk: ebben a könyvben nem foglalkozunk ilyen listákkal.

Listák kiírása

Láttuk már, hogy egy lista egyszerűen kiírható a `print(listanév)` utasítással. Láttuk azt is, hogy így nem szép, kell ennél valami jobbnak lennie. A megoldást egy `join()` nevű tagfüggvény jelenti, amelyiknek a használata elsőre talán meglepő. A zárójelben átadott lista elemeit fűzi össze egyetlen karakterlánccá, az elemek közé pedig az elején, aposztrófok között megadott karakterláncot teszi.

Egészítsük ki az előző programunkat az alábbi sorokkal!




```
6. kacatok_felsorolva = ', '.join(kacatok)
7. print('A kacatjaim: ', kacatok_felsorolva, '.', sep='')
```

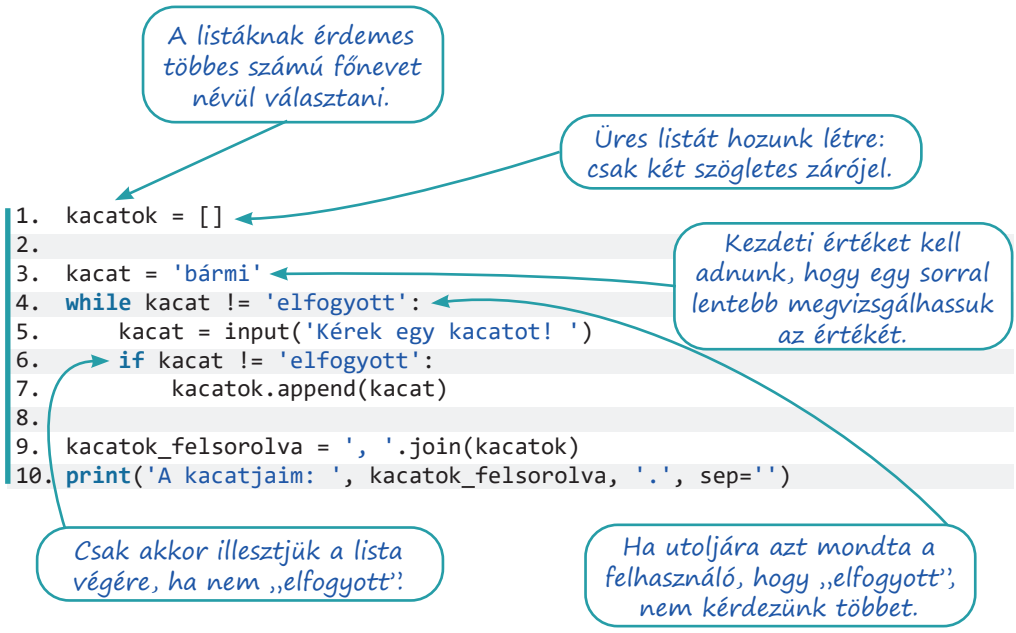
A `join()` csak karakterláncokat tud összefűzni, azaz majd ügyeskednünk kell, ha egy számokból álló listát kell kiírnunk vele. A fenti „csata” listával (amiben vegyesen vannak karakterláncok és számok) végképp nehezen boldogul.

Lista feltöltése a felhasználó által megadott adatokkal

A kacatos listát a felhasználó saját kacatjaival töltjük fel. Minthogy senkinek sincs csak egy kacatja, ciklust szervezünk a feladatra. A kacatokat egyesével kérdezzük meg, és mindegyiket hozzáfűzzük a bővülő lista végére. De honnan fogjuk tudni, hogy befejezhetjük, nincs több kacat?

Meg kell vizsgálnunk minden kacatot, még mielőtt beillesztjük a lista végére, és ha például a kacat neve az, hogy „elfogyott”, akkor befejezzük a lista feltöltését, és kiírjuk, amit kaptunk.

Írjuk meg a programunk első változatát, és próbáljuk ki!




```
1.  kacatok = []
2.
3.  kacat = 'bármi'
4.  while kacat != 'elfogyott':
5.      kacat = input('Kérek egy kacatot! ')
6.      if kacat != 'elfogyott':
7.          kacatok.append(kacat)
8.
9.  kacatok_felsorolva = ', '.join(kacatok)
10. print('A kacatjaim: ', kacatok_felsorolva, '.', sep='')
```

A harmadik sorban csak azért adjuk a „bármi”-t a kacat változó értékéül, hogy egy sorral lentebb megnézhessük, hogy az érték nem „elfogyott”-e. A „bármi” helyett akármi más is szerepelhet itt, csak az „elfogyott” nem, mert akkor egyszer sem lépnék be a ciklusba. A „bármi” értéket nem használjuk sehol, viszont zavaró lehet. A Python az ilyen esetekre tartogatja a nagybetűs semmit, azaz a `None` értéket. Ha a harmadik sort így írjuk át:

```
kacat = None
```

akkor lényegében azt mondjuk a Pythonnak: „Kérünk egy kacat nevű változót, de még ne tegyünk bele értéket.” A negyedik sorban a `while` tudni fogja, hogy a `None` nem ugyanaz, mint az „elfogyott”, és belép a ciklusba.

Informatikusszemmel nézve nem túl szerencsés, amit a hatodik (és később a negyedik) sorban művelünk. Itt ugyanis onnan tudjuk, hogy nincs több kacat, hogy egy különleges nevű kacatot adunk meg. Ha egyszer véletlenül a felhasználónak mégis lesz „elfogyott” nevű kacatja, nem tudja bevinni a programunkba.

Átírható úgy a program, hogy az `'elfogyott'` helyett az üres karakterláncot tekintse a felsorolás végének, azaz az `'elfogyott'` helyett a negyedik és a hatodik sorban is használhatunk két aposztrófot közvetlen egymás mellett: `''`.

A bejárós ciklus

A lista a Python egyik olyan adattípusa, amelyben egyesével végig tudunk lépkedni az elemeken, azaz a lista bejárható, végigjárható. A bejárást egy ciklussal végezzük el, de nem a már megismert feltételes ciklussal.

Értelmezzük és futtassuk az alábbi programot!

A bejárós ciklus egyesével végiglépked egy bejárható objektum, például egy lista értékein. Az épp aktuális értéket betölti a ciklusváltozóba (esetünkben a „folyó” nevűbe). A ciklus magjában használhatjuk ezt a változót. A ciklusmag éppúgy, mint a while-ciklusnál, bentebb kezdődik.

Módosítsuk a kacatos programunkat úgy, hogy a végén bejárós ciklussal írjuk ki a kacatlistát!

Listák és bejárásuk

Feladatok

1. Írjunk olyan programot, amely egy verseny résztvevőinek célba érkezés szerinti névsorát kéri a felhasználótól, majd kiírja a dobogósokat és a sereghajtót!
   Minden versenyző bekérése előtt írja ki az épp aktuális névsort!

```
versenyzők = []

versenyző = None
while versenyző != '':
    print('A versenyzők jelenleg:', ', '.join(versenyzők))
    versenyző = input('Kérek egy versenyzőt! ')
    if versenyző != '':
        versenyzők.append(versenyző)

print('Az első helyezett: ', versenyzők[0])
print('Az második helyezett: ', versenyzők[1])
print('A harmadik helyezett: ', versenyzők[2])
print('A sereghajtó: ', versenyzők[-1])
```

2. Írj olyan programot, amely egy-egy listába bekéri három-három leves, főétel és desszert nevét, majd kiír három menüt, mindegyikben egy levessel, főétellel és desszerttel!

A `range()` függvény arra való, hogy számsorozatokat állítson elő. Háromféle módon használható:

- csak egy számot adunk meg a zárójelben: a `range(5)` a 0,1,2,3,4 számsort állítja elő,
- két számot adunk meg: a `range(2,5)` a 2,3,4 számsort állítja elő,
- három számot is megadunk: a `range(5,15,3)` az 5,8,11,14 számsort állítja elő.

A range-objektumokat leggyakrabban arra használják, hogy az előállított számsorozatot for-ciklussal bejárják. Minden, ami for-ciklussal megoldható, megoldható while-ciklussal is, de sok esetben elegánsabb és egyszerűbb így.

3. Írjuk ki az első 10 természetes számot és a négyzetüket!

```
for szám in range(10):
    print(szám, szám**2)
```

Írjuk át a programot úgy, hogy while-ciklust használjon!

4. Három egymásba ágyazott bejárós ciklussal rajzoljuk ki az alábbi ábrát!

```
ooooo
ooooo
ooooo
ooooo

ooooo
ooooo
ooooo
ooooo

ooooo
ooooo
ooooo
ooooo
```

```
1. for téglalap in range(3):
2.     for sor in range(4):
3.         for oszlop in range(5):
4.             print('o', end='')
5.         print('')
6.     print('')
```

Mi a szerepe az ötödik sornak, és mi a hatodiknak?
Hol kell átírni a kódot, hogy három, az alábbival egyező háromszöget rajzoljon?

```
o
oo
ooo
oooo
```

Ötlet: a `range()` függvény paraméterében szerepelhet ciklusváltozó is.

A lista adattípusnak van `remove()`, azaz eltávolít, kivesz tagfüggvénye is. Az `append()`-hez hasonlóan működik: a zárójelben kell megadnunk azt az elemet, amit kiveszünk a listából. Ha több azonos törlendő van, akkor a `remove()` tagfüggvény az elsőt fogja kivenni a listából. A `len()` (azaz hossz) függvény pedig arra használható, hogy egy lista elemszámát megadja.

5. Írjunk programot, amely egy autókölcsönző munkáját szimulálja! A kölcsönző a munkanapot egy listányi autóval kezdi, és addig kölcsönöz, amíg minden autót ki nem ad. A program írja ki az autók listáját, és kérdezze meg, melyiket kölcsönzi ki a felhasználó. Írja ki, hogy mik maradtak benn, és kérdezzen újra, és így tovább.

```
1. autók = ['Trabant', 'T-Modell', 'Rolls-Royce']
2.
3. while len(autók) > 0:
4.     print('Kölcsönözhető:', ', '.join(autók))
5.     mit = input('Melyik autót kölcsönzi ki? ')
6.     if mit in autók:
7.         autók.remove(mit)
8.     else:
9.         print('Ilyen autóval nem szolgálhatunk.')
```

Addig kölcsönzünk, amíg minden autó ki nem megy.

Így biztosítható, hogy ne akarjunk nem létező elemet kivenni a listából – abba belehalna a program.

Listák mindenféle adatokkal

Feladatok

1. Szimuláljunk tízmillió kockadobást, és az eredményeket tároljuk listában! A programunk számolja meg, hogy hányszor „dobtunk” hatost!

```
1.  import random
2.
3.  dobások = []
4.  for _ in range(10000000):
5.      dobás = random.randint(1,6)
6.      dobások.append(dobás)
7.
8.  ennyi_hatos = 0
9.  for dobás in dobások:
10.     if dobás == 6:
11.         ennyi_hatos += 1
12.
13. print('Összesen', ennyi_hatos' 'hatost dobtunk.')
```

Nem használjuk fel a ciklusváltozót, ezért jó ez a semmitmondó név.

Az első ciklus előállítja a dobásokat.

A második ciklus összeszámolja a hatosokat.

Számoljuk össze mind a hat lehetőség előfordulásait egy 1–6 között futó külső ciklussal!

2. Az egy elem előfordulásának megszámlálására a Python sokkal egyszerűbb megoldást kínál – a lista adattípus `count()` tagfüggvényét. Keressük meg az interneten, hogy miként kell használni, és próbáljuk ki!
3. Kihívást jelentő feladatok, ahol nem segít rajtunk a `count()`, és mindenképp be kell járnunk a listát:
   a. Hány helyen előzi meg a hatos dobást ötös dobás?
   b. Hány helyen van egymás után két hatos?

Eddig a listáinkat érték szerint jártuk be. Amikor listák index szerinti bejárásáról beszélünk, akkor a ciklusváltozóban nem az aktuális listaelem értéke van, hanem az aktuális listaelem sorszáma, azaz indexe.

Az index szerinti bejárásnak a Pythonban két módszere is van. Az első jobban közelít a – Pythonban nem létező, de más nyelvekben elterjedten használt – számlálós ciklusok használatához.

```
1.  fővárosok = ['Párizs', 'Bécs', 'Róma', 'Prága']
2.
3.  for index in range(len(fővárosok)):
4.      print(index, fővárosok[index])
```

A len() itt 4-et ad vissza, a range(4) pedig 0,1,2,3-at. Az index változó értéke tehát 0-1-2-3 lesz.

Kiírjuk a „fővárosok” lista nulladik, első, második és harmadik elemét.

A második módszer pythonosabb, és egyszerre kapjuk meg az aktuális elem indexét és értékét. Itt lényegében két ciklusváltozónk van.

```
1. fővárosok = ['Párizs', 'Bécs', 'Róma', 'Prága']
2.
3. for index, főváros in enumerate(fővárosok):
4.     print(index, főváros)
```

enumerate: számozd be!

Futtassuk a fenti kódot, és figyeljük meg a program kimenetét! Értelmezzük a negyedik sorban lévő két változó szerepét!

4. Állítsunk elő magunknak ezúttal egy tízelemű, pénzfeldobások eredményeit tartalmazó listát! Hány olyan eset van, amikor az aktuális és az előző dobás is „fej”?

```
1.  import random
2.
3.  feldobások = []
4.  for _ in range(10):
5.      feldobás = random.choice(['f', 'í'])
6.      feldobások.append(feldobás)
7.
8.  print('A feldobások:', ', '.join(feldobások))
9.
10. fej_után_fej = 0
11. for index, feldobás in enumerate(feldobások):
12.     if index > 0 and \
        feldobás == 'f' and feldobások[index-1] == 'f':
13.         fej_után_fej += 1
14.
15. print('Ennyiszer volt fej után fej: ', fej_után_fej)
```

Listában adjuk meg, hogy mik közül lehet választani.

Az első értéknél még nem tudjuk megnézni az előzőt.

A túl hosszú sorokat visszaperrel törhetjük. Te nyugodtan írhatod egybe a gépeden.

Az előző elem az, aminek eggyel kisebb az indexe.

5. Állítsunk elő harmincelemű, nulla és kilenc közötti véletlen számokat tartalmazó listát! A számok egy útvonal magassági adatait jelentik. Meredek az útszakasz, ha legalább kettővel magasabb az aktuális hely, mint az előző. Hány meredek szakasz van az úton? És visszafelé?
6. Kihívást jelentő feladat: A programunk elején adjunk meg két listát:
   - az első tartalmazzon öt filmcímet,
   - a második a filmek egy-egy főszereplőjét!

   Az első filmhez az első szereplő tartozik, a másodikhoz a második, és így tovább.
   Írjuk ki a filmcímeket, majd az egyik, véletlenszerűen kiválasztott szereplőt! Kérdezzük meg a felhasználótól, hogy a kiírt szereplő melyik filmnek a főszereplője! Értékeljük a válaszát!

7. Kihívást jelentő feladat: Állítsunk elő nyolcvanelemű, -5 és 3 közötti egész számokból álló listát! A számok egy úszó palackorrú delfin magasságát jelentik. A delfin ki-kiugrál a vízből, ilyenkor pozitív a magassága. Nulla a magasság, amikor a felszínen úszik, negatív, amikor a víz alatt. Írjunk programot, ami választ ad a következő kérdésekre!
   a. Az út mekkora részét tette meg a delfin a vízben, illetve a víz alatt? A válaszok megadhatóak törtszámként és százalékként is.
   b. Víz alatt, vagy víz felett volt-e többet a delfin? A vízfelszínen való utazás egyik esetbe sem számít bele.
   c. Milyen hosszú volt a leghosszabb kiugrása? Az út hányadik pontjánál kezdődött?
   d. Hányszor törte át a vízfelszínt, azaz hányszor követ a listában negatív számot pozitív, vagy fordítva?
   e. Mély merülésnek számít, ha a delfin -4-es vagy -5-ös mélységben van. Az út során hányszor merült mélyre? Figyeljünk arra, hogy például a 4 -2, -4, -5, -5, 3 útvonal csak egy mélyre merülést jelent!

Mobil informatikai eszközök

A technika gyors fejlődése a mindennapi életünkben is sok változást hoz, így van ez az informatika világában is. Az informatikai eszközök köre jelentősen kibővült. Kialakult az informatika egy új ága, amelyet mobil informatikának nevezünk.

A mobil informatikai eszközök közé sorolhatunk minden olyan eszközt, amely a számítógéphez hasonlóan működik, de hordozható. Az ilyen eszközök rendelkeznek a számítógép alapvető hardverelemeivel. Tartalmaznak processzort, memóriát, kimeneti és bemeneti eszközöket. Működésüket általában operációs rendszer biztosítja. Hordozhatóságuk miatt azonban az ilyen eszközök nem vagy nemcsak hálózati áramforrásról működnek, hanem akkumulátorral rendelkeznek. A számítógépes hálózathoz való csatlakozásuk elsősorban vezeték nélkül, wifikapcsolaton vagy mobilhálózaton keresztül történik.

► Tablet, laptop, okostelefon

A mobil eszközök közé sorolhatók a tabletek, laptopok, okostelefonok, e-book-olvasók, okosórák. Körük az újabb technológiák és megoldások fejlődésével rohamosan bővül. Használatuk, új technikai megoldásaik a mindennapi életünkre is hatással vannak.

A hagyományos asztali számítógépektől működésükben a laptopok különböznek a legkevésbé. Operációs rendszereik, szoftvereik az asztali számítógépekétől nem különböznek. Hardverük felépítése is csak annyiban, ami a hordozhatóság érdekében fontos, például kisebb méretű, alacsonyabb energiaigényű alkatrészekre van szükségük.

A tablet vagy táblagép nagyobb méretű érintőképernyővel rendelkező eszköz, amelynek nincs billentyűzete. Dokumentumok szerkesztésére, hosszabb munkára kevésbé alkalmas, de nagy képernyője miatt különböző médiatartalmak kényelmesen megjeleníthetők rajta. Operációs rendszerük Android, iOS és Windows is lehet. Teljesítmény és felszereltség szempontjából az ilyen gépek széles skálán mozognak. Speciális toll segítségével kézírással is írhatunk az érintőképernyőre. Könnyű és keskeny, ezért jól hordozható eszköz.

► Nyomtatott könyv és e-book olvasó

Az e-book-olvasók a hagyományos könyvek alternatívái. A legtöbb papíralapú könyvnél lényegesen vékonyabbak és kisebbek. Elektronikus formában tárolják a könyveket, ezért az ilyen eszközökön akár több könyvespolcnyi

könyvet is magunkkal vihetünk. A tabletekkel ellentétben az e-book-olvasók többnyire e-tinta- (e-ink) technológiát használnak a kép megjelenítésére. Ennek jellemzője, hogy nincs háttérvilágítás, ezért olvasásuk az emberi szem számára sokkal kíméletesebb, mint a számítógépeké vagy tableteké, és az akkumulátor készenléti ideje is jelentősen hosszabb.

Az **okosórák** legtöbbször a használóik mobiltelefonjaihoz kapcsolódnak. Az idő jelzésén kívül számos funkcióval rendelkeznek. Általában alkalmasak a tulajdonos egészségügyi adatainak, sporttevékenységének nyomon követésére, esetleg zenelejátszásra, elektronikus fizetésre. Jelzik a telefonra beérkező hívásokat, üzeneteket, ezeket bizonyos modelleknél el is lehet indítani róluk.

Az okostelefonok

A legszélesebb körben használt mobil informatikai eszköz a **mobiltelefon**. A mobiltelefont eredeti funkciója szerint csak telefonálásra és üzenetküldésre használhattuk. Idővel a fejlesztések révén kiegészült az internetes kommunikáció lehetőségével. A ma használt mobiltelefonok többsége **okostelefon**, amelyen operációs rendszer működik, és az alapfunkciók mellett egyéb programok futtatására is alkalmas. Jellemzőjük, hogy érintőképernyővel, kamerával rendelkeznek, és a szöveges adatokat legtöbbször virtuális képernyő-billentyűzeten vihetjük be.

► Az okostelefon sok feladatra alkalmas

Az okostelefonokat a hardverük, az operációs rendszereik és a rajtuk futó alkalmazások fejlődése révén az életünk egyre több területén használjuk. A telefonunk ma már nagyon sok funkcióval rendelkezik. Alkalmas internetböngészésre, fénykép- és videókészítésre, zenehallgatásra és filmnézésre. Ezért sok régebben külön-külön megjelenő eszközt helyettesítünk vele. Szinte bármilyen online kommunikáció folytatására használhatjuk, kezelhetjük az e-maileket, az internetalapú beszélgetéseket. Meg tudunk velük nyitni dokumentumokat, sőt sok esetben szerkeszthetjük is azokat. Személyes adataink nagyon nagy részét tároljuk ezeken az eszközökön. A rájuk telepített alkalmazások segítségével alkalmasak a mindennapi ügyeink intézésére, például banki vagy közüzemi szolgáltatást vehetünk igénybe. A telefonok jelentős része alkalmas arra, hogy bank- és egyéb kártyáinkat, utazáshoz kapcsolódó jegyeinket rajtuk tárolhassuk. Sok modellel a bankkártya használatát kiváltva tudunk fizetni.

Sokunk számára az okostelefon ma lényegesen több, mint egy egyszerű kommunikációs vagy informatikai eszköz. Tekinthetjük a személyi asszisztensünknek, a munkaeszközünknek, az egyik legszemélyesebb tárgyunknak. Ezért nagyon fontos, hogy használatakor a biztonságra oda kell figyelnünk.

Az okostelefonok hardvere különbözik a számítógépekétől, és az operációs rendszereik is eltérőek. A mobiltelefonokon két elterjedt operációs rendszer működik, az Android és az iOS rendszer. Van még néhány olyan operációs rendszer, amellyel ritkábban találkozunk a telefonokon. Ilyen a Windows Phone, Symbian és BlueberryOS.

Az iOS operációs rendszer az Apple cég iPhone telefonjain és iPad tabletjein fut. Az operációs rendszer kifejezetten ezekre a készülékekre készül, ezért jól meghatározhatók azok a hardverváltozatok, amelyeken működniük kell. Az iOS zárt forráskódú operációs rendszer, csak az Apple cég fejleszti. Az operációs rendszert használó telefonok túlnyomó többsége rendszeresen megkapja a szoftver frissítésének lehetőségét.

Az Android operációs rendszert a Google cég fejleszti. Az Android rendszert futtató telefonok, tabletek nagyon sokfélék. Sok gyártó eltérő felépítésű, felszereltségű hardverén kell működnie a rendszernek. Ez az egyik oka annak, hogy az operációs rendszer nyílt forráskódú. Minden gyártó valamelyest a saját gyártmányaihoz alakíthatja, így biztosítva a jobb teljesítményt. Az Androidot futtató telefonok kisebb része kap rendszeres szoftverfrissítést. A telefon gyártója kezében van az adott modellek szoftverfrissítése, amely nem feltétlenül akkor történik, amikor a Google az új verziót közzéteszi, és nem minden modellre érhető el. Míg az iOS-nél az aránylag régebbi eszközökön is frissíthető az operációs rendszer, addig az Androidnál ez gyakran hiányzik.

A telefonok operációs rendszereire rengeteg applikációt telepíthetünk. Az applikációk egy része ingyenes, másokért fizetni kell. Az ingyenes alkalmazások nagy részében reklámok jelennek meg, illetve alkalmazáson belüli vásárlási lehetőséget ajánlanak fel, ezzel teszik gazdaságossá készítésüket. Az applikációk, függetlenül az áruktól, a megfelelő online áruházban kereshetők, vásárolhatók meg és tölthetők le. Az Android rendszer esetén ez a Play Áruház, az iOS esetén az App Store, a Windowsnál pedig a Microsoft Store.

▸ Mobil operációs rendszerek logói

A telefonos operációs rendszerekre jellemző, hogy az applikációkat ikonokkal jelenítik meg a képernyőn, és gesztusokkal irányíthatjuk ezeket. Gesztusoknak a képernyő érintésekor végzett különböző műveleteket nevezzük. Ilyen például a koppintás, a dupla koppintás, a legyintés, a húzás és a két ujjal való méretezés.

Kérdések, feladatok

1. Gyűjtsünk össze olyan informatikai vagy hétköznapi feladatokat, amelyeket gyakran végzünk mobil eszközökkel!
2. Hasonlítsuk össze a hagyományos könyvet az e-könyvvel! Milyen előnyöket találhatunk az egyik vagy a másik használatában? Miben különbözhet az e-könyv e-book-olvasóval vagy tablettel történő olvasása?
3. Milyen feladatokat láthat el az okosóra? Miért lehet hasznos fiataloknak, és miért idősebbeknek?

Az okostelefonok biztonságos használata

A telefonunkban egyre több személyes adatot tárolunk. Ezek között vannak olyan adatok, amelyeket mi magunk mentünk el, ilyenek például a névjegyeink, a jelszavaink, ezen keresztül az applikációkban tárolt adataink. Vannak olyan adataink is, amelyeket a telefonra telepített alkalmazások a beépített szenzorok segítségével gyűjtenek rólunk, anélkül hogy ennek tudatában lennénk. Ilyenek például a helyadatok, amelyek alapján követhető szinte minden lépésünk. Ezeket az adatokat használják az útvonaltervező programok a forgalmi dugók figyelésére. A személyes adatok között sok olyan akad, ami nem jó, ha illetéktelen kezekbe kerül.

Mi az, amire érdemes ügyelnünk a biztonság érekében?

Az okostelefonok lényegében számítógépek, ezért rájuk is kerülhetnek **vírusok** vagy más **kártékony szoftverek**. Az ilyen kódokat leggyakrabban új applikációk telepítésekor szerezhetjük be. Az applikációk tartalmazhatnak olyan kódokat, amelyek az eszközön folyó tevékenységet figyelik meg, és az adatokat továbbküldik idegen félnek. Ezt végezhetik úgy, hogy a felhasználó tudomására hozzák, hogy az alkalmazás javítása érdekében teszik ezt. Az ilyen tevékenységet általában meg lehet tiltani. Vannak viszont olyan alkalmazások, amelyek ezt titokban végzik, nem tudjuk, milyen adatainkat gyűjtik, és mire használják fel az információkat.

► Play Áruház és App Store

A hivatalos áruházakba az applikációk csak ellenőrzés után kerülhetnek be. Ezért ha ezekből töltünk le, az jóval biztonságosabb. Az App Store szabályzata szigorúbb, mint a Play Áruházé, ezért az Android-felhasználóknak érdemes óvatosabbnak lenniük. Nem ajánlott a hivatalos áruházi verzió helyett más forrásból, esetleg ingyenesen beszerezhető applikációkat telepíteni. Ezek sokszor pont azért ingyenesek, mert a kívánt hasznot tudtunk nélkül, az adataink megfigyelésével, továbbadásával szerzik meg. A rendszer figyelmeztetéseit érdemes komolyan venni. Az áruházakban az alkalmazás telepítése előtt tudunk tájékozódni. Ajánlott elolvasni az alkalmazás részletes leírását, figyelembe venni az értékelését, és az értékelők által írt kommenteket.

Sok alkalmazás már a telepítésekor vagy az első indításakor **engedélyek**et kér az eszközünk különböző adatainak, például tartózkodási hely, kamera, névjegyek, mobil adatforgalom használatához. Az engedélykérést érdemes elolvasni, és átgondolni, valóban szükség van-e mindenre. Arra érdemes engedélyt adnunk, ami valóban hasznos funkciót biztosít számunkra. Az ilyen engedélyek a biztonsági kockázat mellett nem kívánt adatforgalmat is generálhatnak, ezzel anyagi kárt okozhatnak. Ajánlatos a beállításokban az alkalmazások engedélyeit időről időre felülvizsgálni.

A mobil eszközök sérülékenysége a hordozhatóságuknak is köszönhető. Egy telefont, tabletet sokkal könnyebb elveszíteni vagy eltulajdonítani, mint egy asztali számítógépet. Erre a kockázatra fel kell készülni. Az adatok védelme, az illetéktelen hozzáférés megakadályozása érdekében, mindig legyen az eszközön **képernyőzár**. Ez az előre beállított idő eltelte után csak valamilyen biztonsági kód megadásával engedi az eszközhöz való hozzáférést.

A képernyőzár feloldásához a készülékek különböző lehetőségeket biztosítanak. Ilyenek lehetnek: a PIN-kód (Personal Identification Number = személyi azonosító szám) vagy a képernyőre rajzolható minta. Az újabb eszközök lehetővé teszik a **biometrikus azonosítást**. Az erre alkalmas eszközök ujjlenyomatolvasóval, arcfelismerővel rendelkeznek. Ezek az egyedi azonosítók biztonságosabbak, mint a kódok, amelyek könnyen kifigyelhetők, esetleg visszafejthetők.

► Azonosítás ujjlenyomattal

Az eszközünk elvesztése a rajta tárolt adataink elvesztésével járhat. Fontos, hogy legyen ezekről **biztonsági másolat**unk. Ezt készíthetjük saját magunk is, de a mobil operációs rendszerek mindegyike tartalmaz ehhez kapcsolódó szolgáltatást. Az adatainkat általában a rendszerhez kapcsolódó személyes felhőbe mentik (Android – Google Drive, iOS – iCloud, Windows – OneDrive). Ezekből eltulajdonítás, de egy esetleges készülékváltás esetén is könnyen helyreállíthatók a készülékhez kapcsolt adataink.

A készülék eltulajdonítása esetén jelent segítséget a **nyomkövetés** beállítása. Ennek bekapcsolásával lehetőség van az eszköz megkeresésére, távolról történő zárolására és a rajta tárolt adatok törlésére.

Az egyes alkalmazások használatakor igyekezzünk figyelni a belépés biztonságára. Gondoljuk át, melyek azok az alkalmazások, amelyekbe belépve szabad maradnunk. Indokolt esetben érdemes minden alkalommal az újra belépést választani. Ennek megkönnyítésére sok eszköz állhat rendelkezésünkre. Több alkalmazásban beállítható a biometrikus azonosítás. Vannak olyan alkalmazások, amelyekkel a jelszavainkat tudjuk tárolni. A **jelszókezelők** titkosítva tárolják a bejelentkezési adatokat, jelszavakat, esetleg más fontos adatokat (pl. bankszámlaszámokat). Ilyen jelszókezelő például a LastPass vagy az iCloud Kulcskarika.

Érdemes beszélnünk az okostelefon-használat más irányú veszélyeiről is. Napjainkban az emberek, és főképpen a fiatalok egyre több időt töltenek mobiltelefon, tablet használatával. Bizonyos határt átlépve ennek lehetnek a testi és lelki egészségre nézve káros hatásai. Figyeljünk a mobil eszközök mellett töltött időre! Ezt az időt az operációs rendszer vagy egyéb applikációk segítségével nyomon követhetjük. Érdemes az eszközt bizonyos időközönként letenni. A telefonos értesítések állandó hangjelzései szintén lehetnek zavaró hatásúak. Ajánlott az értesítéseket, hangjelzéseket legalább az éjszakai pihenés idejére elnémítani.

► A túlzott mértékű telefonhasználat káros lehet

Kérdések, feladatok

1. Milyen feltételek mellett használhatjuk a biometrikus azonosítást?
2. Milyen káros hatásokkal járhat a túlzott mobileszköz-használat? Hogyan lehet ezt megelőzni?
3. Tekintsük át a mobiltelefonunk beállításait! Mely applikációk milyen jogosultsággal rendelkeznek? Milyen biztonsági beállításokat, appokat találhatunk a mobiltelefonunkon? Hasonlítsuk össze a különböző operációs rendszerek lehetőségeit!

Mobiltanulás

A mobil eszközöket életünk egyre több területén használjuk. A kapcsolattartás, kommunikáció, szórakozás, különböző ügyintézések hatékonyságnövelése vagy az utazásszervezés mellett több olyan szolgáltatást adnak a mobil eszközök, amelyek a tanulásunkat, ismeretszerzésünket segíthetik, hatékonyabbá tehetik. A mobil eszközök funkcióit hasznosító tanulási formákat nevezzük mobiltanulásnak (M-learning). A mobil eszközeink segítségével új ismereteket szerezhetünk, gyakorolhatunk, elmélyíthetjük a tudásunkat, szemléletesebbé, érthetőbbé tehetjük a tananyagot, vagy segíthetjük egymást a tanulási folyamatban.

A mobiltanulás eszközeinek köre igen széles, és állandóan változik, bővül. Alapos áttekintésük meghaladja a könyv kereteit, itt csak néhány példát mutatunk be.

Mobiltelefont tanórán mindig csak tanári engedéllyel, az iskolai házirend betartásával szabad használni.

Általános tanulást segítő alkalmazások

Az alkalmazások egy része általánosan a tanulási folyamat segítésére használható, míg mások konkrét tantárgy tanulásához alkalmazhatók.

Jegyzetelés, megosztás

Sok olyan alkalmazás áll a rendelkezésünkre, amelyek telefonon is használhatók, és segítenek az online együttműködésben, információrögzítésben és -megosztásban. Ezek az alkalmazások telefonon, tableten is futnak, de asztali gépen is. Billentyűzettel vagy kézírással készíthetünk jegyzeteket. A jegyzet nemcsak írásos anyagot, hanem képeket, rajzokat, videókat, linkeket is tartalmazhat. Egyes tabletek képernyőjére speciális tollal írhatunk, így az jóval pontosabb, a kézíráshoz hasonló kezelést tesz lehetővé. Mobil eszközökön használható jegyzetelő alkalmazás például a OneNote, az Apple jegyzetek, az Evernote és az Inkpad. Bizonyos applikációk megosztható táblaként működnek. Ilyen például a Whiteboard alkalmazás. Szerkesztés közben többen írhatunk rá, a dokumentumot közösen, együttműködve alakíthatjuk ki. Ha megfelelő felszereléssel rendelkezünk, akkor ezekkel az alkalmazásokkalal a hagyományos papíralapú jegyzetelés teljesen kiváltható. Sok alkalmazás a kézírást is képes felismerni és átalakítani nyomtatott szöveggé. A digitális jegyzeteknek számos előnyük lehet. Kisebb helyen tárolhatók, nagy mennyiségben is magunkkal vihetjük őket. A rendszerezésük, a tartalmukban való keresés egyszerűbbé válhat.

▶ Digitális jegyzet készítése

Gondolattérkép-készítők (pl. XMind), projektszervezők (pl. Trello), alkalmazás- és linkgyűjtemények (pl. PearlTrees) szintén segíthetnek bennünket a tanulás során.

Kép-, filmkészítés, szerkesztés

A mobil eszközök sokszor kamerát tartalmaznak, így szoftvereikkel fotó, videó- és hangfelvétel készíthető. A tanulási folyamat során a felvétel sok esetben segítséget jelenthet. Felvehetünk velük egy-egy előadást, eljárást, mozdulatsort azért, hogy újra megnézhessük, lejátszhassuk, könnyebben memorizálhassuk.

Alkalmas lehet egy-egy fizikai, kémiai kísérlet felvételére. A videó segítségével pontosabban megfigyelhetjük, elemezhetjük a jelenséget, mint szabad szemmel történő megfigyeléskor. A videót egy gyorsan lejátszódó folyamat esetén lassíthatjuk, egy lassú folyamat esetén gyorsíthatjuk, vagy megállíthatjuk egy-egy fontos részletnél. A videót átalakíthatjuk, vághatjuk, feliratozhatjuk, szerkeszthetjük.

A videók készítéséről és szerkesztéséről részletesebben a *Multimédiás dokumentumok készítése* című fejezetben olvashatunk.

Oktatóprogramok

Sok kifejezetten oktatás céljára készített program áll rendelkezésünkre, melyek jelentős része mobil eszközökön is, vagy csak ott használható. Az online kvízek (Kahoot, Quizlet, Quizizz…), tesztek (Redmenta, Socrative, Forms, Google űrlap...), feladatmegoldó szoftverek (LearningApps, OkosDoboz...) szinte mindegyikéhez használhatunk mobil eszközt. Így olyan helyzetben is dolgozhatunk velük, ahol nem áll rendelkezésre asztali számítógép.

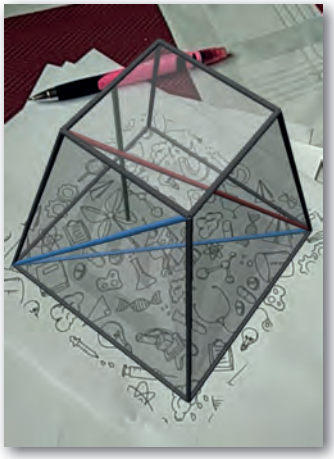
Kiterjesztett valóság (Augmented Reality, rövidítve AR)

Az oktatást segítő alkalmazások egy része **kiterjesztett valóság** segítségével teszi szemléletesebbé a tananyagot.

A kiterjesztett valóság a körülöttünk lévő valós teret virtuális elemekkel egészíti ki. Például mobil eszköz kameráján keresztül nézve olyan tárgyakat is láthatunk a környezetünkben, amelyek valójában nincsenek ott. Ellentétben a virtuális valósággal, a kiterjesztett valóság alkalmazásainak nem feltétlenül van szükségük különleges eszközre, legtöbbjük kamerával, GPS-szel, érzékelőkkel felszerelt mobiltelefonnal vagy tablettel használható.

A kiterjesztett valóság virtuális elemeit különböző módon hívhatjuk elő. Vannak olyan alkalmazások, amelyek egy előhívó (marker) segítségével jelenítik meg a virtuális elemeket. Ilyen például, amikor egy meghatározott képet kell az eszközünk kamerájával beolvasnunk a háromdimenziós ábra előhívásához. Ha a képet mozgatjuk, a virtuális alakzatot körbe tudjuk járni. Ilyen alkalmazás például a Quiver vagy az Eddie, amelyek markerként egy-egy nyomtatható ábrát használnak, vagy a Merge Cube, amelynek használatához egy speciális kocka szükséges.

Más esetben a virtuális tartalom előhívásához a helyzetünket (GPS-koordinátát, irányt, gyorsulást) használják. Egy ilyen alkalmazásban adott helyre kell eljutnunk, hogy láthassuk a virtuális tartalmat. Ha a megadott helyen vagyunk, például egy videót, egy zenét vagy egy képet érhetünk el.



▶ Kiterjesztett valóság: Eddie alkalmazás

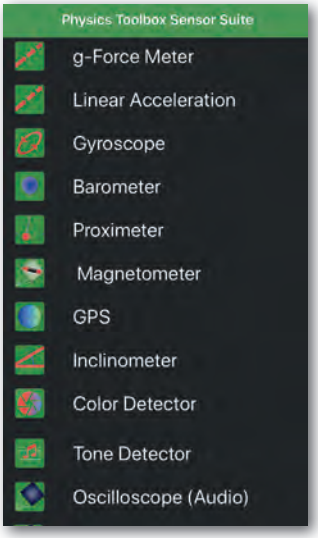
A kiterjesztett valóságot sokrétűen használják. Valós térben játszható játékban virtuális alakzatokat kereshetünk velük; vásárlás előtt kipróbálhatjuk, hogy hogyan mutatna a bútor a szobánkban; múzeumban, városban idegenvezetőként segíthet minket. Emellett természetesen az oktatásban is tudjuk alkalmazni szemléltetőeszközként, a megfelelő helyen megfelelő feladatokat, információkat átadó alkalmazásként. Ilyen például a holokauszt áldozatainak emléket állító IWalk alkalmazás. A WallaMe alkalmazás segítségével magunk is elhelyezhetünk virtuális tartalmakat bizonyos helyekre, pontokra.

Nyelvtanulás

A nyelvtanulás ma már elképzelhetetlen digitális eszközök nélkül. A nyelvtanuláskor szükséges szótárak, fordítóprogramok mobil eszközünkre telepített alkalmazásként állandóan rendelkezésünkre állhatnak. Emellett számos idegen nyelvi oktatóprogramot találhatunk a mobil rendszerek alkalmazásai között.

Mérések

A mobil eszközöket számos érzékelővel szerelik fel. Ezek az érzékelők, vagy más néven szenzorok sok telefonos funkció működéséhez szükségesek. Ezeket a tanulás során felhasználhatjuk kísérletekhez és mérésekhez. A beépített szenzorok érzékelik a mozgást (pl. gyorsulás, elfordulás), az eszköz helyzetét (GPS-koordináták, iránytű) és a környezet állapotát (nyomás, hőmérséklet, megvilágítás, távolság érzékelése). A szenzorok mérési adatait közvetlenül nem látjuk a mobil eszközünkön, de léteznek olyan alkalmazások, amelyeket feltelepítve kiolvashatjuk, sőt kezelhető formában (általában táblázatként) exportálhatjuk is. Így van mód az adatok elemzésére, kiértékelésére. Ilyen alkalmazás például a Physics Toolbox, amellyel gyorsulás, elfordulás, nyomás, mágnesesség, hangfrekvencia és sok egyéb mennyiség mérhető. A mérési adatok elemzéséhez fizikai, informatikai és matematikai ismeretekre van szükségünk.



► Physics Tollbox mérőeszközei

Tudományágak, tantárgyak alkalmazásai

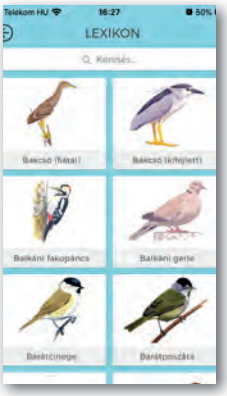
Sok alkalmazás elsősorban egy-egy téma vagy tantárgy esetében használható.

A matematika tanulását segítő szoftverek közül az egyik legalapvetőbb a GeoGebra, amelynek létezik mobilváltozata is. A függvények, geometriai ábrák, szerkesztések, koordináta-geometriai számítások során használhatjuk, de számos egyéb, nem csak szorosan matematikához köthető feladatban is segítségünkre lehet.

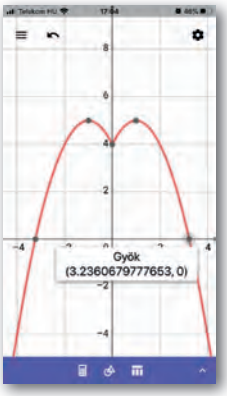
Több tantárgyban hasznosak az adattárakat tartalmazó alkalmazások. Ilyenek például a növény- és állathatározók, a periódusos rendszerek, a történelmi adattárak. Sok olyan alkalmazás van, amely a tananyagot teszi szemléletesebbé: háromdimenziós körbejárható modelleken keresztül mutat be szabad szemmel kevéssé látható vagy távoli dolgokat, történelmi helyszínek rekonstrukcióját.

► Fa Book

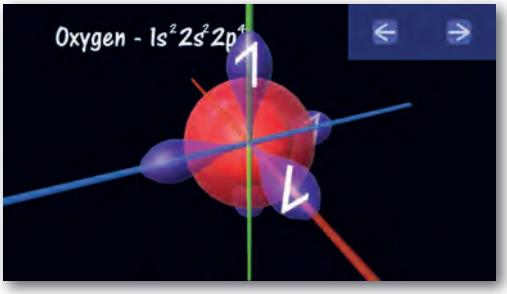
► Periódusos táblázat 2020



► Madárhatározó

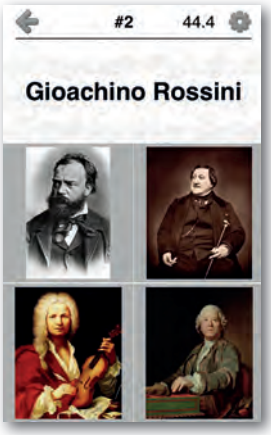


► GeoGebra

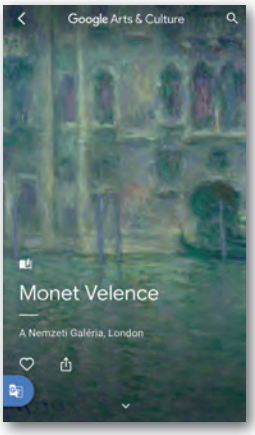


► Virtual Orbitals 3D szemléltető modell

► Ancient Egypt bejárható 3D modell



► Famous composers



► Google Arts&Culture

► Settera

Kérdések, feladatok

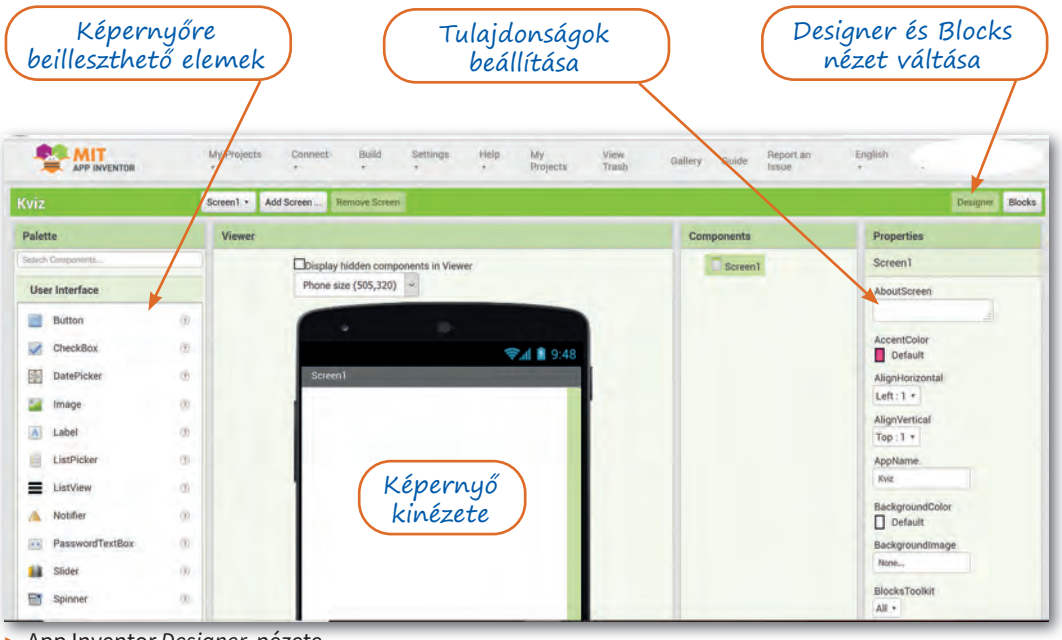
1. Gyűjtsünk alkalmazásokat, amelyekkel egyes tantárgyak tanulását segíthetjük! Próbáljuk ki őket! Indokoljuk meg, hogyan segíthetik a tanulás eredményességét!
2. Soroljuk fel a digitális jegyzetelés előnyeit és hátrányait!
3. Próbáljunk ki mobiltelefonos mérést segítő applikációkat!

Egyszerű mobilalkalmazás készítése

A mobil eszközök programozása szakértelmet igényel, de vannak olyan eszközök, amelyek ezt egy átlagos felhasználó számára is könnyen érthetővé teszik. Ilyen például az AppInventor alkalmazás, amelynek segítségével egyszerűen készíthetünk programokat Android operációs rendszerre. Az alkalmazásokat online felületen lehet létrehozni. Az ehhez szükséges weboldalt a MIT (Massachusetts Institute of Technology) tartja fenn. Az oldal használatához be kell jelentkeznünk, amelyhez a Google-fiókunk adatait kell megadnunk. Miután regisztráltunk és elfogadtuk a feltételeket, új projektet indíthatunk.

Az applikációnk elkészítése során két lényegesen különböző felületen fogunk dolgozni. Az első felület a *Designer* ablak: itt állíthatjuk be a képernyő és a képernyőn megjelenő egyes elemek kinézetét. A projekt nevének megadása után ebbe a nézetbe lépünk be. A másik a *Blocks* ablak: ott készíthetjük el az alkalmazásunk kódját. A két nézet között a jobb felső sarokban lévő gombokkal válthatunk.

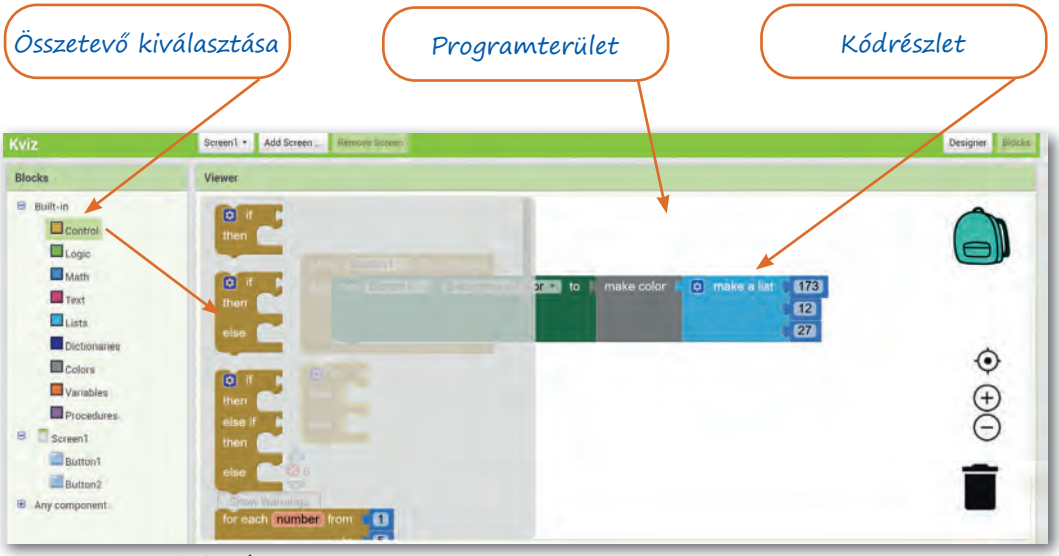
A Designer ablak felépítése:



► App Inventor *Designer* nézete

A középen látható telefon képernyőjén jelennek meg az általunk kiválasztott elemek. A beilleszthető objektumok listája *(Palette)* a telefon képétől balra helyezkedik el, a másik oldalon pedig a tulajdonságaikat állíthatjuk be. Az elemeket egyszerűen behúzhatjuk a telefon képernyőjére, majd a jobb oldalon beállítjuk a kinézetüket.

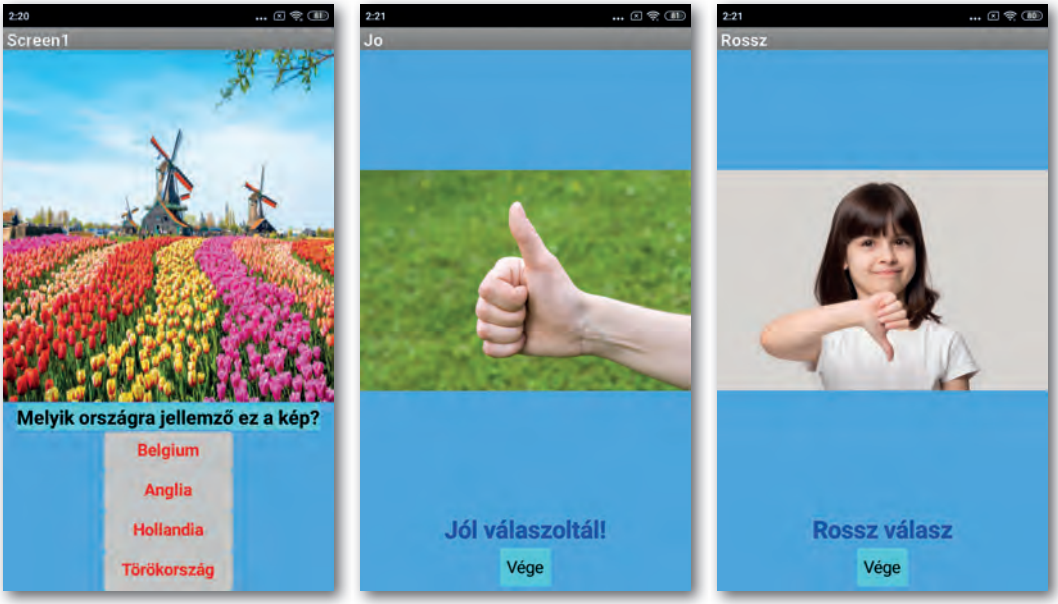
A *Blocks* nézetben a beállított elemeknek megfelelően csoportosítva jelennek meg a program egyes lehetséges utasításai. Ezeket az utasításokat kiválasztva és a kódterületre *(Viewer)* behúzva állíthatjuk össze a programunkat.



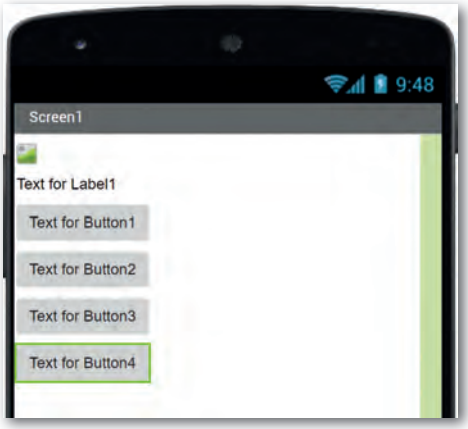
► App Inventor *Blocks* nézete

Próbáljunk meg egy egyszerű programot elkészíteni. A program egy kvízkérdést fog feltenni, amelyre a nyomógombok egyikének megnyomásával válaszol a felhasználó. A válasz helyessége szerint a program megjeleníti a megfelelő képernyőt.

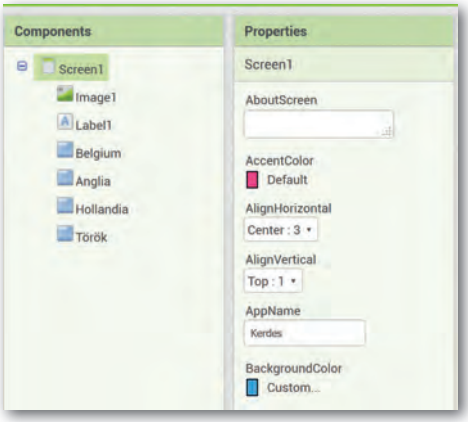
A megvalósítandó terv így néz ki:



► Képernyőképek



► A képernyőn megjelenítendő elemek



► Az összetevők tulajdonságainak beállítása

► Block nézet: utasítások típusok szerint rendezve

Először hozzuk létre az új projektet a menüsorban a *My Projects* > *New project* menüpontnál. Legyen a projekt neve *Kerdes*. Ezután a képernyő kinézetét kell felépítenünk a *Designer* nézetben. Az első képernyő kialakításához a bal oldali elemek közül szükségünk lesz egy kép elemre *(Image)*, egy szöveges mezőre *(Label)* és négy darab nyomógombra *(Button).* Behúzzuk ezeket a megfelelő sorrendben a képernyőre.

Kialakítjuk a képernyő tartalmát. A jobb oldali részben először az összetevőt kell kiválasztanunk *(Components)*, utána a komponens tulajdonságait állítjuk be *(Properties).* Először a kezdőképernyő *(Screen1)* igazítását *(Center)* adjuk meg, és beállítjuk a háttérszínt.

Ezután a kép megjelenítése következik. Ehhez először feltöltjük a megfelelő képet az oldalra. Ezt a *Components* alatt elhelyezkedő *Media* részben tudjuk megtenni. Ha kész a feltöltés, az összetevők között kattintunk az *Image* elemre, majd a jobb oldalon a tulajdonságok között a *Picture* részben kiválasztjuk e feltöltött képet. A kép most megjelenik a képernyőn. Mivel a mérete nem felel meg a képernyőnek, kilóg, ezért ezt is be kell állítanunk. A kép magasságát *(Height)* és szélességét *(Width)* úgy állítjuk be, hogy kitöltse a helyet *(Fill parent).* A szöveges mezőt és a nyomógombokat is beállítjuk. Tetszésünknek megfelelően kiválasztjuk a színeket és a nyomógombok alakját. A nyomógombok feliratát *(Text for Button)* és a képernyőn megjelenő szöveget *(Text for Label)* begépeljük. Ezzel a kezdőképernyő készen van.

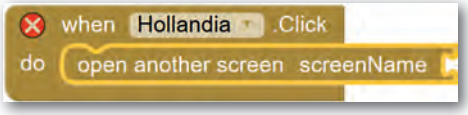
Készítünk még egy-egy képernyőt a jó és rossz válaszoknak. Érdemes a két képernyőt megfelelően elneveznünk, és a háttereik színét a kezdőképernyőével azonosra beállítanunk. Mind a kettőn elhelyezzük, beállítjuk a képeket és a szövegeket, nyomógombokat az első képernyőhöz hasonlóan. Ha ezt megtettük, a dizájn elkészítésével készen vagyunk.

Átlépünk a *Blocks* nézetbe. Itt a *Blocks* részben típusok szerint rendezve találjuk meg az algoritmusok szokásos felépítő elemeit és az egyes objektumokhoz tartozó speciális utasításokat.

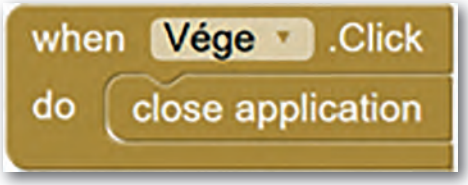
A *Control* csoportban találjuk meg a vezérlőelemeket, elágazásokat, ciklusokat. A *Logic* részben a logikai műveleteket és konstansokat, a *Math* részben a matematikai műveleteket, konstansokat és így tovább. A kategória alsó részén helyezkednek el a *Designer* ablakban beépített elemekhez tartozó műveletek. Az egyes algoritmuselemeket a középső területre húzzuk, és ott megfelelően összeillesztjük.

Programunkkal a következőt valósítjuk meg. Ha a felhasználó a helyes válaszra, vagyis a Hollandia feliratú gombra kattint, akkor a jó válasznak megfelelő képernyő jelenjen meg, ha pedig másik nyomógombra, akkor a rossz válasznak megfelelő.

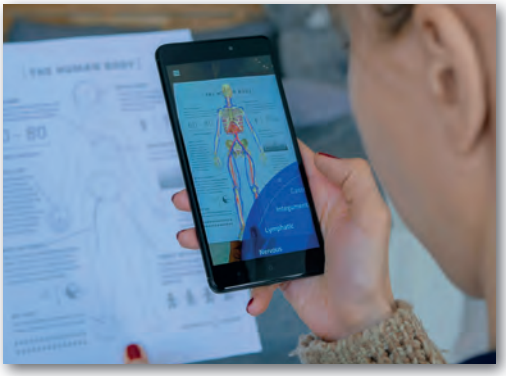
A következőképpen valósítjuk meg az algoritmust: A bal oldali *Blocks* kategóriában Hollandia nyomógombjára kattintunk. Kiválasztjuk azt az elemet, amellyel a gombra való kattintás esetére adunk további utasítást. Ez a képen látható elem lesz. Ezután kiválasztjuk, hogy mi történjen a kattintás hatására. A *Control* csoport elemei között keressük meg a képernyőváltásra vonatkozó blokkelemet (lásd a képen). Az elemek körvonalán megfigyeljük, hogy hol vannak a csatlakozási pontjaik. Ezek segítségével illesztünk hozzájuk új elemet. Összeillesztjük most az előző kettőt. Az így keletkezett elem bal felső sarkában is látszik, hogy az elemmel így még valami gond van. Természetesen az elem másik oldalán is láthatjuk az üres csatlakozási felületet. Az algoritmust végiggondolva tudjuk is a hiányosságot: nem adtuk meg, hogy melyik képernyőt kell betöltenünk. A megadáshoz kiválasztunk egy szöveges konstanst. Ezt a *Text* csoportban találjuk. Az előző elem végére illesztjük, és kitöltjük az új képernyő nevével. Ezzel az elemünk készen lesz, a figyelmeztetés is eltűnik róla. Hasonló módon elkészítjük a többi nyomógombra vonatkozó utasítást, csak ott a másik képernyő betöltését adjuk meg.



A program már működőképes, de gondoskodnunk kell arról, hogy ki lehessen lépni belőle. Ennek érdekében került az értékelés képernyőire a *Vége* feliratú nyomógomb. Átváltva ezekre a képernyőkre, a megfelelő utasítást itt is beállítjuk, ez az applikáció bezárása *(close application)*.



Ha mind a két képernyőn beállítottuk ezt a műveletet, készen vagyunk az applikáció összeállításával. A fordítás következik. A menüsorban a *Build* opció választásával a programunkat lefordítjuk. Ezen a ponton választanunk kell, hogy hogyan szeretnénk átvinni a kész programot a mobiltelefonra. A QR-kód lehetőséget választjuk, így a program fordítása



után megjelenő kódot beolvasva le tudjuk tölteni a programot a telefonra. Érdemes a kipróbáláshoz esetleg egy már használaton kívüli Android operációs rendszerrel felszerelt telefont használni. A letöltött programot telepítjük. Telepítéskor az ismeretlen forrásokat engedélyezzük, a rendszer többször is figyelmeztethet a veszélyekre, de ezt figyelmen kívül hagyhatjuk most. Nincs más hátra, kipróbáljuk a programot, és ha szeretnénk, módosítunk rajta.

Kérdések, feladatok

1. Készítsünk a fenti leíráshoz hasonló programot!
2. Fejlesszük tovább a programot, készítsünk további kérdéseket, változtassunk a kinézetén, működésén!
3. Alakítsunk három-négy fős csoportokat, és válasszunk az alábbiak közül egy témakört, amelyet közösen részletesebben kidolgozunk!
   a. Válasszunk egy tantárgyat, amelyhez tanulást segítő applikációkat gyűjtünk! Lássuk el a gyűjteményt leírásokkal, magyarázatokkal!
   b. Dolgozzunk ki egy mobiltelefon segítségével végzett mérést!
   c. Készítsünk egy új programot a mobiltelefonra az AppInventor segítségével!

Publikálás a világhálón

Ebben a fejezetben átismételjük a világhálóval kapcsolatos alapfogalmakat, majd áttekintjük, hogy mi magunk hogyan készíthetünk és publikálhatunk weboldalakat.

Az internet és a web kapcsolata

Előzetes tanulmányainkból már tudhatjuk, hogy az internet nem más, mint egy globális, az egész bolygónkat behálózó számítógépes hálózat.

Ez a hálózat valójában nem egy fizikai hálózatot jelent, hanem sokféle, önálló hálózatból, illetve hálózatfajtából áll, amelyek egymással összeköttetésben állnak. A hálózatra csatlakoztatott eszközök egy egyedi hálózati azonosítóval rendelkeznek, amelyet **IP-címnek** (**I**nternet **P**rotocol-címnek) nevezünk. Példa egy ilyen IP-címre: `84.206.104.74`.

Az internethálózat sokféle információmegosztási, kommunikációs szolgáltatást támogat. Ilyen például az **e-mail** (elektronikus levél), a **chat** (csevegés) vagy a **fájlátviteli szolgáltatás** (pl. FTP, SFTP, SCP), amely az állományok számítógépek közti átvitelét (feltöltését, letöltését) teszi lehetővé.

A **world wide web** (amelyet magyarul **világhálónak** nevezünk) szintén egy olyan szolgáltatás, amelyet az internethálózaton érhetünk el. Tehát a világháló és az internet nem ugyanazt jelentik, bár sokan (tévesen) szinonimaként használják a két fogalmat.

A world wide web megalkotása *Tim Berners-Lee*, valamint *Robert Cailliau* nevéhez fűződik, akik a CERN (Európai Nukleáris Kutatási Szervezet) munkatársai voltak. 1989-ben egy olyan rendszert kezdtek el kidolgozni, amely a világ különböző helyszínein dolgozó kutatók közti információmegosztást támogatta.

1993. április 30-án a CERN bejelentette, hogy a kidolgozott technológiát (www) mindenki szabadon, ingyenesen használhatja. Ebben az évben Magyarországon is elindult az első www-kiszolgáló, ami a `www.fsz.bme.hu` webcímen üzemelt.

1994-ben Tim Berners-Lee megalapította a *World Wide Web Konzorcium* nevű szervezetet (w3.org), amely mind a mai napig koordinálja a nyílt, webes szabványok kidolgozását.

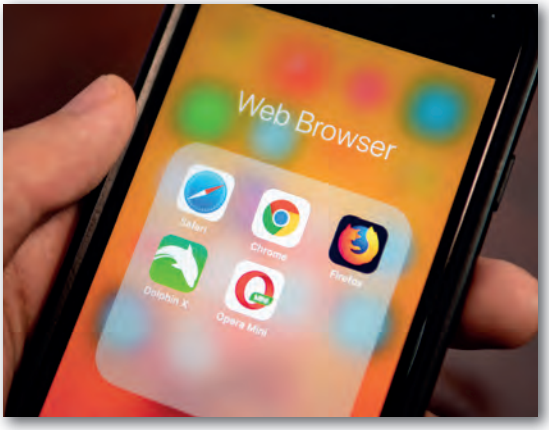
▶ Sir Timothy John Berners-Lee

A www (világháló) építőkövei

A világháló egy olyan információs rendszer, amelyre jellemző, hogy **böngészőprogramok** segítségével különböző dokumentumokat (pl. weboldalakat, PDF-dokumentumokat stb.) és egyéb állományokat (pl. képeket, videókat, zenéket stb.) érhetünk el. Ezeket összefoglalóan **erőforrásoknak** (*resource*) nevezzük. Ezen erőforrások között a **hiperhivatkozások** (*linkek*) teremtenek kapcsolatot.

Így egy weboldalról kiindulva a különböző hivatkozások követésével újabb és újabb oldalakra juthatunk el. Az viszont nem garantálható, hogy a visszafelé irányt is meg tudjuk tenni a hivatkozások segítségével, mivel a **hivatkozások egyirányúak**. A saját weboldalunkról bármilyen weboldalra hivatkozhatunk, de a hivatkozott oldalon ettől még nem biztos, hogy fognak hivatkozni a mi oldalunkra.

A www megalkotása során különböző kérdésekre kellett választ adniuk a tervezőknek. A következőkben ezekkel részletesebben foglalkozunk.



▶ Böngészőalkalmazások mobil platformon

Milyen eszközzel érjük el a világhálón publikált anyagokat?

Tudjuk, hogy a világhálón publikált anyagokat jellemzően a **webböngésző programok** segítségével érhetjük el.

Böngészőprogramokat sokféle cég, szervezet fejleszt. A statisztikák szerint a legnépszerűbb webböngészők napjainkban a következők: *Google Chrome, Safari (Apple), Mozilla Firefox, Samsung Internet, Microsoft Edge, Opera.*

Egy operációs rendszerre akár több, különböző böngészőprogramot is telepíthetünk.

Feladatok, kérdések

Három-négy fős csoportokban vitassuk meg, hogy mely böngészőprogramokat kedveljük a legjobban, és miért? Állítsunk fel ez alapján népszerűségi rangsort! Ha használunk a felsoroltakon kívül más böngészőprogramokat, azoknak milyen előnyeik, egyedi funkcióik vannak? Minden csoport szóvivője ismertesse az eredményeket!

A böngészőprogram képes megjeleníteni a világhálón közzétett (publikált) tartalmakat, de ehhez meg kell adnunk azt a webcímet, ahol az adott tartalom elérhető. Folytassuk az ismerkedést ezzel!

Hogyan hivatkozunk az elérendő dokumentumra?

A dokumentumok és más állományok elérési helyére az úgynevezett **URL** (*Uniform Resource Locator*), vagyis *Egységes Erőforrás Helymeghatározó* segítségével hivatkozhatunk. Az alábbiakban néhány példát látunk erre.

https://www.nava.hu/	A Nemzeti Audiovizuális Archívum weboldala.
https://www.mnm.hu/kiallitasok	A Magyar Nemzeti Múzeum portál egy aloldala, amely a kiállításokat mutatja be.
http://www.fortepan.hu/_photo/display/154350.jpg	A Fortepan közösségi fotóarchívum weboldalán elérhető egyik kép webcíme.
https://www.w3.org/standards/	Webes szabványok a W3 konzorcium weboldalán.
https://hirmagazin.sulinet.hu/	A Sulinet hírmagazin webportálja.
https://www.google.com/search?q=világháló	A Google kereső találati oldala a világháló szóra keresve.
https://hu.wikipedia.org/wiki/Tim_Berners-Lee#Kitüntetései	A Tim Berners-Leeről szóló Wikipédia-oldal kitüntetésekkel foglalkozó oldalrészére mutató webcím.
file:///C:/honlapra/hobbi.html	A saját számítógépünk c:\honlapra mappájában található `hobbi.html` nevű dokumentum.

Ahogy a példákban is látszik, az URL több részre tagolódik. Ezek a következők:

1. Séma

A séma utal arra, hogy milyen szabályrendszer (protokoll) szerint történik a kommunikáció a hálózat tagjai között. A protokoll leírja, hogy hogyan kell történnie a kapcsolatfelvételnek, hogyan történik az üzenetváltás, mi az üzenetek felépítése és az egyes üzenetekben milyen adatok lehetnek.

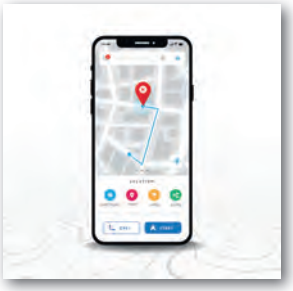
Érdekesség: A protokoll kifejezést hétköznapi értelemben is használjuk, a különböző viselkedési szabályokra, illemtani ismeretekre utalva.

A webes dokumentumok, állományok vonatkozásában leggyakrabban a `http` vagy `https` sémát szoktuk megadni. A séma neve után egy kettőspontot kell írni. A kettőspontot webes tartalmak esetén két perjel (//) követi.

Fontos! Amikor a böngészőprogramban megadunk egy webcímet, sokszor nem szoktuk begépelni az előtagot (pl. http://, https://), mégis megjelenik a weboldal. Ennek oka, hogy a böngészőprogramok ilyen esetekben alapértelmezetten a http-sémát használják.

Amennyiben a saját számítógépünk egy mappájából nyitunk meg egy dokumentumot (pl. weblapot) a webböngészőben, akkor láthatjuk, hogy nem a *http* sémát, hanem a *file* sémát használja a böngészőprogram.

2. Útvonal



A séma után kell megadnunk az erőforrás elérési útvonalát. De ezt hogyan tudjuk megtenni?

Ha egy épülethez szeretnénk eljutni, akkor használhatjuk a helyszín GPS-koordinátáit. Például a Parlament épületének a GPS-koordinátái: N47.50708 E19.04591. Ez a számsor jól használható egy navigációs szoftverben, de nehezen lehetne megjegyezni. Helyette így is megadhatnánk ugyanezt a címet: Budapest V. kerület, Kossuth tér 1–3. Ez utóbbi cím szintén azonosítja a helyet, és jól megjegyezhető.

Láttuk korábban, hogy az internethálózatra csatlakoztatott eszközök egy egyedi IP-címmel rendelkeznek (pl. `84.206.104.74`). A webböngésző programban akár ezen IP-cím segítségével is elérhetnénk a gépek szolgáltatásait. Ez azonban túlságosan bonyolult lenne, hiszen éppúgy egy hosszú, számokból álló azonosítót kellene használnunk, mint az előbbi példában a GPS-koordinátákat. Másrészt ezek az IP-címek változhatnak. Elképzelhető, hogy egy eszköz minden csatlakozáskor más-más IP-címet kap.

Ezért fontos, hogy a webes kiszolgálókra ne csak IP-cím alapján, hanem egy könnyebben megjegyezhető, állandó névvel is hivatkozhassunk. Ilyen a korábbi példánkban a város, utca, házszám. Ez az azonosító az **állomásnév**, vagy más néven **hosztnév** (hostname). A teljes elérési útvonalnak azonban nemcsak az állomásnév a része, hanem az úgynevezett doménnév is.

A **doménnév** (domain name) egy olyan egyedi azonosító, amely kettő vagy több részből áll, és ezeket pontok választják el. A doménnév végződése (a korábbi példákban a *hu, com* és *org*) a **legfelső szintű tartományt** jelenti. Ezek lehetnek országokra utaló, illetve kategóriákra utaló rövidítések. Például a *hu* kód Magyarországot jelöli, a *jp* Japánt, az *org* a szervezeteket, a *com* az üzleti weboldalakat, és így tovább.

A legfelső szintű tartomány előtt a **második szintű tartományt** találjuk. Például a *sulinet.hu* tartománynévben a *sulinet* a második szintű tartománynév.

Második szintű tartománynevet meghatározott díjért bárki lefoglalhat saját célra (pl. vállalkozása, alapítványa, családja számára), amennyiben az adott név még szabad.

Tipp! Azt, hogy egy doménnév lefoglalható-e még, az Internet Szolgáltatók Tanácsa weboldalán (http://www.domain.hu/domain/szabad-e/) lehet ellenőrizni.

Feladatok, kérdések

Nézzük meg, hogy a saját családnevünkkel megegyező doménnév lefoglalható-e (pl. Balaton Éva esetén azt kell ellenőrizni, hogy a balaton.hu doménnév szabad-e).

Mit tapasztalunk? Az osztály mekkora hányada tudná regisztrálni a saját nevét?

A második szintű tartomány tulajdonosa saját hatáskörben akár további aldoménneveket (harmadik, negyedik szintű) is kioszthat, és **helyi állomásneveket (hosztnév)** is meghatározhat. Az állomásnév egyben doménnév is lehet. Például a `hirmagazin.sulinet.hu` cím esetén a doménnév a helyi állomásnévből (`hirmagazin`) és a `sulinet.hu` szülődoménnévből áll össze.

Hivatkozás mappákra és a bennük található erőforrásokra

A webes kiszolgálókon (szervereken) az információk mappaszerűen vannak szervezve, kicsit hasonlóan ahhoz, mint a saját számítógépünkön is. Az elérési út azt írja le, hogy milyen útvonalon érhető el az adott erőforrás.

Például a http://www.fortepan.hu/_photo/display/154350.jpg webcím esetén a www.fortepan.hu tartományhoz tartozó szerver _photo mappájában van egy display mappa, amelyben a 154350.jpg nevű képre hivatkozunk.

Ha nem adunk meg útvonalat, vagy csak egy mappa nevét adjuk meg, akkor egy alapértelmezett tartalom fog megjelenni a kiszolgáló szerver beállításai alapján (pl. `index.html, index.php, default.aspx`)

Feladatok, kérdések

1. Szerveződjünk három-négy fős csoportokba! Egy böngészőprogramban nyissuk meg a saját iskolánk weboldalát! A böngészés során próbáljuk kideríteni, hogy vajon melyik aloldalnak van a leghosszabb webcíme! Osszuk fel egymás között, hogy ki melyik részét térképezi fel a weboldalnak, például a menüpontok alapján. Jegyezzük fel a leghosszabb webcímet, és értelmezzük, hogy az elérési útvonalban mi mit jelenthet!
2. Vannak-e az iskolának harmadik szintű tartománynevei? Ha igen, akkor melyek ezek? Gyűjtsünk össze párat!

3. Lekérdezési paraméterek

A webcímben akár paraméterek is lehetnek, amelyek alapján a weboldal akár más-más tartalmat tud szolgáltatni. A paramétereket *kulcs = érték* formában lehet megadni, az első kulcs elé pedig kérdőjelet kell tenni.

Például a https://www.google.com/search?q=világháló webcímben paraméterként egy q nevű kulcs van megadva, amelynek értéke a világháló szöveg. A q az angol query (lekérdezés) szó rövidítése. Ezt a kulcsot és értéket egy program feldolgozza, és eredményként azon weboldalak listáját mutatja meg, amelyekben szerepel a „világháló” szöveg.

Ha több ilyen kulcs lenne, akkor azokat & jellel kellene elválasztani.

Például a https://www.google.com/search?q=világháló&num=3 webcím már két kulcsot is tartalmaz. A második a num, ami az angol number (szám) szó rövidítése. Ez a megjelenítendő találatok számát jelenti.

Feladatok

Próbáljuk ki a Google keresőjében azt, hogy közvetlenül a webcímben módosítjuk a paraméterek értékét! Keressünk rá ezzel a módszerrel a „Duna” kifejezésre!

4. Oldalrész

A webcímek végén egy olyan azonosító is állhat a # jel után, amely egy oldalrészre hivatkozik. Ilyenkor a böngészőprogram az oldal adott elnevezésű részéhez fog ugrani.

Fontos! Csak akkor tudunk egy weboldal adott részére ugrani, ha az oldal készítője az adott oldalrészt ellátta egy egyedi névvel. Ezalól kivételt jelent az oldal teteje, amelyre név nélkül, csak a # jel megadásával hivatkozhatunk.

Például a https://hu.wikipedia.org/wiki/Tim_Berners-Lee#Kitüntetései webcím a Tim Berners-Leeről szóló Wikipédia-oldal azon részére ugrik, ahol a kapott kitüntetéseiről van szó.

Feladatok

A Wikipédia oldalán keressünk olyan szócikket, ami egy olyan témával kapcsolatos, amelyet mostanában tanultunk, és rendelkezik oldalrész-hivatkozással. Másoljuk ki az oldalrész-hivatkozást tartalmazó linket, és mentsük el egy állományba!

Az URN és URI fogalma

Érdekesség. Nemcsak az elérési útvonal, hanem speciális esetben egy **név (azonosító) alapján is hivatkozhatunk** egyes objektumokra. Ezt az URN *(Uniform Resource Name)*, vagyis *Egységes Erőforrás Név* segítségével tudjuk megtenni. Ebben az esetben viszont alapvető követelmény, hogy a név világviszonylatban (térben és időben) egyedi legyen, vagyis valóban egyértelműen azonosítson egy elemet.

A saját nevünk például nem lehet URN, mert nem garantálható, hogy egy másik embert (akár a világ másik részén) ne hívjanak ugyanígy, most, vagy a jövőben. De az is lehet, hogy a múltban is élt már ilyen névvel valaki.

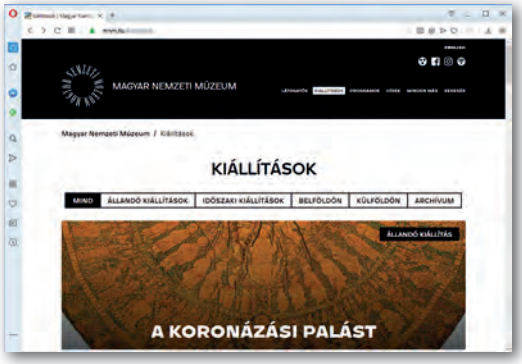
De akkor mi lehet URN? Bizonyára találkoztál már az ISBN számmal a könyvek borítóin. Ez az iktatószám világszinten egyedi. Magyarországon ezen azonosítókat a kiadók kérésére az Országos Széchényi Könyvtáron belül működő Magyar ISBN és ISMN Iroda osztja ki. Például Magyarország Alaptörvényére ezzel az URN azonosítóval hivatkozhatunk: **urn:isbn:9786155269813**

Mind a korábban látott **URL-ek**, mind az **URN-ek** beletartoznak egy **nagyobb kategóriába**, amely az **URI** (*Uniform Resource Identifier*) névre hallgat. Ezt magyarul *Egységes Erőforrás Azonosítónak* hívjuk. Vagyis minden URL egyben URI is, de nem minden URI-ra igaz, hogy URL.

Hogyan történik a kommunikáció?

A világhálón jellemzően a http (*hypertext transfer protocol*, magyarul *hiperszöveg-átviteli protokoll*) szerint történik a kommunikáció, illetve biztonsági okokból gyakran a https-sémát használjuk a webböngészőkben, ami azt jelenti, hogy titkosított, biztonságos formában történik a kommunikáció.

A kommunikáció kérés-válasz formájában zajlik a kliens és a szerver között. A kliens (ami tipikusan a webböngésző program, de más alkalmazás is lehet) kérést küld a kiszolgálónak, vagyis a szervernek. Erre a kérésre válaszol a webszerver.



▶ Weboldal megjelenése egy böngészőben

A kommunikáció során a webböngésző megkapja az adott weboldal forráskódját is. A weboldal tartalmazhat képeket, videókat és egyéb elemeket is, amelyek szükségesek a megjelenítéshez. Ezeket a böngészőprogram szintén le kell, hogy kérje a szervertől újabb kérés-válasz ciklusokban, majd a kapott adatok alapján megjeleníti az oldalt.

Hogyan, milyen nyelven írjuk le a dokumentumok tartalmát, illetve kinézetét?

A tartalom leírása (HTML)

A világhálón jellemzően a HTML (*HyperText Markup Language*) jelölőnyelvet használjuk arra, hogy a weboldalak tartalmát leírjuk. Ennek a nyelvnek több verziója is van. A legutolsó és egyben legkorszerűbb változat a HTML5. Ezzel fogunk mi is foglalkozni.

Fontos! *A HTML nem programozási nyelv, hanem a webes dokumentumok tartalmának leírására szolgáló leírónyelv.*

A HTML nyelvben úgynevezett tagek (kiejtve *tegek*), vagy magyarul címkék segítségével tudjuk leírni a dokumentum tartalmát. A címkék „<" jellel kezdődnek és „>" jellel végződnek. Például egy bekezdés kezdetét a `<p>` taggel jelöljük. A p az angol *paragraph*, vagyis *paragrafus* szó rövidítése.

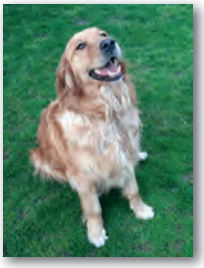
Egy címke hatása addig tart, amíg meg nem adjuk a záró párját. A záró címke hasonló, mint a kezdő címke, csak egy „/" karakter van a címke neve előtt. Vagyis egy bekezdést így jelölhetünk a HTML nyelv segítségével:

```
<p>Ez egy bekezdés</p>
```

Vannak olyan címkék is, amelyeknek nincsen záró párjuk. Ilyen például a `<br>` tag is, amellyel az utána következő szöveget új sorba törhetjük.

```
<p>Ez egy bekezdés, <br> amiben sortörést alkalmaztunk.</p>
```

A címkék használata során akár paramétereket is megadhatunk, sőt bizonyos esetekben ez kötelező is. A paramétereket a címke nyitó részében kell megadnunk `paraméter="érték"` alakban. Egy címkéhez több paramétert is írhatunk, ekkor szóközzel kell elválasztanunk őket. Vannak úgynevezett globális paraméterek. Ezeket bármilyen tag esetén megadhatjuk. Ilyen például a `title`, `class`, `id`. Más paramétereket csak az adott tag esetén használhatunk.



Például egy kép beillesztéséhez az `<img>` címkét kell használnunk. Az img az angol image (kép) szó rövidítése. Viszont ebben az esetben kötelezően meg kell adnunk azt is, hogy hol érhető el a kép (mi az URL-je). Ez kerül az `src` paraméterbe. Az src a source (forrás) szó rövidítése. Sőt, azt is le kell írnunk szövegesen, hogy mi látható a képen. Ez az `alt` paraméterbe kerül Az alt az alternate (helyettesítő leírás) rövidítése..

```
<img src="goldi.jpg" alt="A kutyám, aki egy Golden
Retriever.">
```

Az előbbi példában a `goldi.jpg` képet illesztettük be az oldalra. Mivel nincs megadva semmilyen hosszabb útvonal, csak a fájl neve, ez azt jelenti, hogy a kép ugyanabban a mappában van, mint maga a weblap, amelybe be van illesztve. Ha esetleg a képet nem tudná letölteni a böngészőprogram, akkor helyette az `alt` paraméterben megadott leíró szöveget jeleníti meg. A vak emberek szintén ezen szöveg alapján tudják eldönteni, hogy mi látható a képen.

A kinézet leírása (Stíluslap)

A HTML5 nyelv segítségével a weboldalak tartalmát írhatjuk le, a kinézetét viszont nem. Erre a CSS (*Cascading Style Sheets*) szabvány szolgál, amelyet magyarul legtöbbször *lépcsőzetes stíluslapoknak, csatolt stíluslapoknak vagy egymásba ágyazott stíluslapoknak* fordítanak.

A szabványnak több verziója van, a jelenlegi legkorszerűbb változat a CSS3.

A stíluslapok szakszerű alkalmazásának számos előnye van:

- Számos formázást (pl. színbeállítások, szövegbeállítások) és oldalfelépítést (pl. kétoszlopos elrendezés) támogat.
- A weboldalak karbantartása egyszerűsödik, mivel az oldal kinézetének leírása nem keveredik a tartalom leírásával.
- Segíti az akadálymentes oldalak előállítását (később erre még kitérünk).
- Különböző eszközökre (pl. okostelefonokra, nyomtatókra, okosórákra) külön-külön stíluslap készíthető, amely biztosítja az optimális elrendezést és használhatóságot.

Nézzünk egy egyszerű példát! Korábban láttuk, hogy a bekezdést a `<p>` címkével tudjuk jelölni. Hogyan tudnánk minden bekezdés kinézetét megváltoztatni a stíluslap segítségével úgy, hogy a szöveg kék színű legyen, és nagyobb betűmérettel jelenjen meg?

Ahhoz, hogy **kijelöljük** a formázáshoz az oldal összes bekezdését, először meg kell adnunk a tag (címke) nevét. Ezért ezt a részt kijelölőnek vagy **szelektornak** hívjuk.

```
p {
    color: blue;
    font-size:120%;
  }
```

A szelektor után a { és } jelek közötti **deklarációs blokkban** megadhatjuk, hogy hogyan szeretnénk megjeleníteni az adott elemet. Itt meg kell adnunk a tulajdonságokat, illetve az ahhoz tartozó értékeket.

A tulajdonság-érték megadást **deklarációnak** nevezzük. A különböző deklarációkat pontosvesszővel kell elválasztani egymástól.

Például a `color` tulajdonság a szöveg színét állítja be. Értékként megadhatunk különböző színneveket angolul.

A `font-size` tulajdonság a betű méretét jelöli. A 120% érték azt jelenti, hogy az alapértelmezett méret 120%-át szeretnénk beállítani, vagyis kis mértékben meg szeretnénk növelni a betűméretet.

CSS osztályok használata

Persze egyáltalán nem biztos, hogy minden egyes bekezdést ugyanúgy szeretnénk formázni. Ebben az esetben használhatunk **osztályokat**. A HTML-kódban az egyes elemeket osztályokba sorolhatjuk. Ehhez a `class` paramétert kell használnunk. Az osztály nevét mi találhatjuk ki. A névben számok, betűk, kötőjelek és aláhúzások lehetnek, de például szóköz és más speciális karakter nem!

Ha például ki akarunk emelni egy bekezdést a többi közül, akkor használhatjuk osztálynévként a `kiemelt` szót!

Tipp! Mindig próbáljunk meg olyan osztálynevet megadni, amely általánosan írja le a formázás jellegét. Például ne adjunk olyan osztálynevet, hogy *`piros`*, mert lehet, hogy később az elem kinézetet megváltoztatjuk lilára, és akkor ez az osztálynév félrevezető lesz.

A CSS-kód:

```
p.kiemelt {
    color: blue;
    font-size:120%;
  }
```

A HTML-kód

```
<p class="kiemelt">Ez egy kiemelt bekezdés</p>
<p>Ez pedig egy normál bekezdés</p>
```

Látható, hogy a CSS-kódban pontosan ugyanazt a nevet kell használnunk, mint a HTML-kódban, úgy, hogy a bekezdést jelölő `p` után egy pontot teszünk, majd leírjuk az osztály nevét. Ebben az esetben a `kiemelt` nevű osztály csak a bekezdés esetén érvényesül.

Létrehozhatunk olyan általános osztályokat is, amelyeket akár minden elemnél felhasználhatunk. Ekkor a pont elé ne írjunk semmit!

A CSS-kód:

```
.szegely {
    border:1px solid blue;
  }
```

A HTML-kód

```
<p class="szegely">Szegélyezett
bekezdés</p>
<h1 class="szegely">Szegélyezett
címsor</h1>
```

A fenti példában a `szegely` osztályban azt adtuk meg, hogy az elem körül legyen egy szegély (*border*), ami 1 képpont vastag (*1px*), folytonos vonallal van megrajzolva (*solid*), kék (*blue*) színnel. Mivel ez egy általános osztály, több tag esetén is érvényre jut a formázás. Használhatjuk bekezdésnél (`<p>`), címsornál (`<h1>`), és még sok más elemnél is.

Reszponzív weboldalak

A weboldalakat napjainkban igen sokféle eszközön tekintjük meg. Használhatunk asztali számítógépeket extranagy felbontású monitorokkal, notebookokat nagy felbontású kijelzőkkel, közepes kijelzőjű tableteket és kis kijelzős okostelefonokat, az egészen apró kijelzőjű okosórákról nem is beszélve!

Bármelyik eszközzel is nézzük meg a weboldalt, nagyon fontos, hogy az oldal az adott kijelzőn optimális módon jelenjen meg, vagyis egyszerűen lehessen navigálni az oldalak között, jól olvasható legyen a tartalom, ne kelljen átméretezni, illetve indokolatlanul sokszor görgetni a tartalmat. Ha egy weboldal teljesíti ezeket a követelményeket, akkor **reszponzív kialakítású weboldalnak** nevezzük.

Amikor magunk készítünk weboldalakat, akkor is törekednünk kell arra, hogy az oldal reszponzív legyen. Később látjuk majd, hogy akár különböző sablonokból is kiindulhatunk a weblapok készítése során. Itt mindig próbáljunk olyan sablont, illetve stílust kiválasztani, amely reszponzív megjelenést tesz lehetővé!

Azt, hogy egy weboldal reszponzív-e, úgy tesztelhetjük le, hogy többféle eszközön is megtekintjük az oldalt, illetve például asztali számítógépen a böngészőablakot átméretezzük (lecsökkentjük, felnagyítjuk).

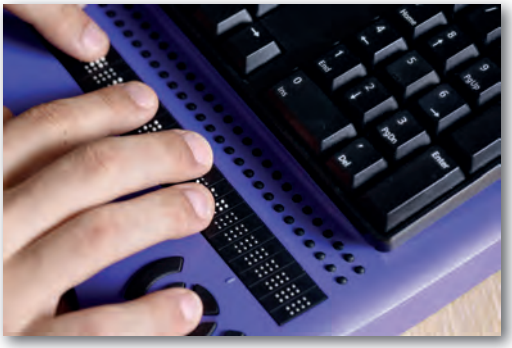
Feladatok, kérdések

Keressünk a saját hobbinkhoz kapcsolódó weboldalt, és vizsgáljuk meg, hogy az reszponzív módon lett megvalósítva, vagy sem. Használjunk a teszteléshez okostelefont, notebookot és/vagy asztali számítógépet!
Amikor mindenki elkészült, összesítsük az eredményeket! A vizsgált honlapok közül mekkora hányad volt reszponzív megvalósítású?

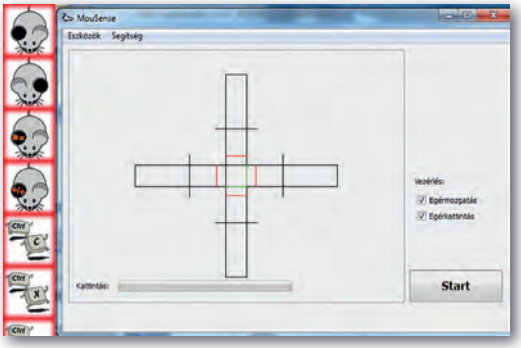
Weboldalak akadálymentessége

Nagyon sok, fogyatékkal élő embertársunk is böngészik a világhálón, ezért rendkívül fontos, hogy a weboldalakat olyan formában készítsük el, hogy mindenki számára, **akadálymentes** módon használhatók legyenek. Ha figyelünk a honlapok akadálymentes megvalósításra, az minden felhasználó számára előnyökkel járhat.

A fogyatékkal élő felhasználók jelentős hányadára jellemző, hogy a számítógép kezeléséhez, illetve a böngészőprogramok használatához valamilyen segédeszközt használnak. Ez lehet például egy speciális billentyűzet, olyan szájba vehető vagy fejre erősíthető pálca, amellyel le lehet nyomni a billentyűket, lehet Braille-kijelző a vak emberek által használt pontírás megjelenítésére, vagy akár egy speciális segédprogram (pl. képernyőolvasó program, fejegér, képernyőnagyító program) is.



▸ Braille-írás (pontírás) megjelenítésére képes kijelző a vak felhasználók számára



▸ Fejegér-alkalmazás (Mousense), amely lehetővé teszi, hogy az egeret ne kézzel, hanem fejmozgással irányítsák a felhasználók

A **vak emberek** nagyon gyakran úgynevezett képernyőolvasó programot használnak, amely felolvassa számukra a böngészőprogramban megnyitott weboldalak tartalmát, mégpedig a weblap forráskódja alapján. Ebben az esetben például nagyon fontos, hogy a **képek és más médiaelemek (pl. animációk, videók)** tartalmát **szövegesen** is leírjuk. Használjunk egyértelmű, jól érthető, a tartalomhoz megfelelően kapcsolódó **címsorokat**. Szintén alapvető, hogy a tartalom leírásakor mindig a megfelelő elemeket használjuk. Egy címsor nem attól lesz címsor, hogy nagy, félkövér betűkkel jelenítjük meg, hanem hogy a megfelelő HTML-címkét használjuk a kódban. Fontos, hogy lehetőleg **ne állítsunk be háttérzenét** az oldalon, mivel ez zavaró a vak emberek számára (illetve nem csak számukra). A linkek szövegét úgy kell megfogalmaznunk, hogy abból kiderüljön, hogy milyen információt találunk rajta. Például, ha a *„Kattints ide”* szöveget állítjuk be hivatkozásként, akkor ebből nem derül ki, hogy mi fog történni a linkre kattintáskor.

A **gyengénlátó** emberek számára biztosítani kell, hogy jól olvasható, kontrasztos legyen a weboldal, és akkor is jól jelenjen meg, ha a betűméretet megnövelik a böngészőprogramban. Az animációk elkészítésénél is ügyeljünk a kontrasztarányra, illetve adjunk lehetőséget az animáció megállítására, kimerevítésére.

A **siket emberek** számára nagyon fontos, hogy a témát szemléletesen mutassuk be. Amennyiben meg tudjuk oldani, a videókat feliratozzuk, vagy a videók felhasználásakor részesítsük előnyben azokat, amelyek felirattal vannak ellátva. Az akadálymentesség egy mélyebb szintjét valósítják meg azok a videók, ahol jelnyelvi tolmácsolás is látható. A vak, illetve siket felhasználók számára a videó teljes szövegű átiratának megléte is nagyon fontos lehet. Ez az állomány tartalmazza a videó minden fontos történésének (párbeszédek, cselekmények) szöveges leírását, hasonlóan, mintha egy forgatókönyvet írnánk.

Nagyon sok **színtévesztő**, illetve **színvak** felhasználó nehézségekbe ütközik akkor, ha bizonyos információkat csak színekkel különböztetünk meg egymástól. Az alábbiakban egy olyan példát láthatunk, amikor a színnel való megkülönböztetés nem elegendő. Ilyen esetben például használhatunk ikonokat is a helyes, illetve helytelen válaszok jelölésére.

Minek a rövidítése a HTML?

- HyperText Markup Language
- HyperText Meta-Language
- HiperText Meta-Language
- HiperText Markup Line

▶ Teszt, melyben zöld és piros színekkel van megkülönböztetve a helyes és helytelen válasz

Minek a rövidítése a HTML?

- HyperText Markup Language
- HyperText Meta-Language
- HiperText Meta-Language
- HiperText Markup Line

▶ Ugyanez a teszt egy olyan ember szemével, aki zöld-sárga-piros tartományba eső színeket nem tudja megkülönböztetni

A **mozgáskorlátozott** emberek egy része nehézségekbe ütközik, ha billentyűzetet kell használnia, vagy az egérrel finom mozdulatokat kell elvégeznie. Ezért elegendő időt kell hagyni az adatbevitelre, illetve ügyelni kell arra, hogy ne kelljen nagyon pici területre kattintaniuk a felhasználóknak. Figyelni kell arra, hogy a weboldal csak billentyűzettel (egér nélkül) is használható legyen, de ez például a vak felhasználók számára is rendkívül fontos.

Feladatok, kérdések

1. A *funkify.org* webcímen találunk egy ingyenesen kipróbálható szimulátort, amelyet a Google Chrome böngészőbe lehet telepíteni kiterjesztésként. Ez a szimulátor különböző állapotokat (pl. gyengénlátás, színtévesztés, remegő kezek) tesz kipróbálhatóvá. Próbáljuk ki ezeket a funkciókat úgy, hogy közben a saját iskolánk weboldala van megnyitva a böngészőprogramban!
2. Akár átmenetileg is kerülhetnek a felhasználók olyan helyzetbe, mint ha fogyatékossággal élnének. Például ha rossz a hangkártya a számítógépben, akkor nem fogjuk hallani a számítógép hangját, akárcsak a siket emberek. Szerveződjünk három-négy fős csoportokba, és vitassuk meg, hogy milyen hétköznapi helyzetekben fordulhat elő az, hogy átmenetileg hasonló problémába ütközünk, mint a fogyatékosságokkal élő embertársaink!

Készítsünk weblapot!

A weboldalak készítésének több módja is van. Kisebb, pár oldalból álló honlapokat elkészíthetünk úgy, hogy a HTML-állományok forrását egy kódszerkesztőben készítjük el, vagy használhatunk segédprogramokat, amelyek hasonlítanak ahhoz, mintha egy szövegszerkesztő programban dolgoznánk, de az eredményt képesek elmenteni HTML-formátumban. Amennyiben a kódot közvetlenül szerkesztjük, majd azt publikáljuk, akkor valójában **statikus oldalakat** készítettünk.

Dinamikus honlapok készítése

Összetettebb weboldalakat, amelyek számos oldalból állnak, és esetenként többen is szerkesztik a tartalmukat, már nem hatékony statikusan előállítani. Érdemesebb egy **CMS** rendszert (*Content Management System*), vagyis *Tartalomkezelő rendszert* használni, amely az oldalakat valós időben, **dinamikusan** állítja elő. A dinamikusság azt jelenti, hogy a webszerveren futó alkalmazás állítja elő azt a HTML-kódot, amelyet majd a böngésző megjelenít. Így például megtehetjük azt is, hogy egy közösségi oldalon megjelenő bejegyzés automatikusan megjelenik a weboldalon is anélkül, hogy nekünk kellene a tartalmat módosítanunk.

Statikus honlapok készítése

Későbbi tanulmányainkban több tartalomkezelő rendszer használatával is megismerkedünk. Azonban ezekben a rendszerekben is előnyös, ha a HTML-kódot közvetlenül tudjuk módosítani, illetve a kinézetet leíró stíluslapállományt testre tudjuk szabni. Ezért először ismerkedjünk meg a statikus oldalak elkészítésének módjával!

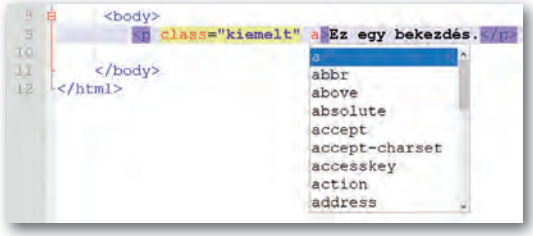
Ahhoz, hogy egy statikus honlapot elkészítsünk, egy nagyon egyszerű kódszerkesztő programot is használhatunk.

Előnyös, ha ez a program rendelkezik a következő tulajdonságokkal:

- Kijelzi a sorok sorszámát, mert így egyszerűbben meg tudjuk találni az adott sorszámú sort, ha például egy tanári bemutatót követünk.
- Képes színnel megkülönböztetni a tageket, a paramétereket és a paraméterek értékeit. Ezt szintaxiskiemelő funkciónak nevezzük.
- Amikor elkezdjük gépelni a tageket vagy paramétereket, akkor felkínálja, hogy milyen tagek/paraméterek kezdődnek az adott karakterekkel, így nem kell feltétlenül pontosan emlékeznünk az adott kifejezésre, hanem akár ki is választhatjuk a megjelenő listából.

Kódszerkesztő alkalmazások

▶ Kódkiemelés és paraméterek felkínálása a Brackets szerkesztőprogramban



▶ Kódkiemelés és paraméterek felkínálása a Notepad++ szerkesztőprogramban

Ha kiválasztottuk a megfelelő programot, amelyben kódolni fogunk, akkor induljunk ki az alábbi HTML5-sablonból! Ebben fogjuk elhelyezni az oldal tartalmát, a különböző HTML-címkéket használva.

```
<!DOCTYPE html>
 <html lang="hu">
    <head>
        <meta charset="utf-8">
        <title></title>
     </head>
     <body>
     </body>
</html>
```

▶ HTML5-alapstruktúra

Az első sor tartalmazza a **dokumentumtípust**. Ez azt írja le, hogy melyik szabvány szerint írjuk le a tartalmat, Ez a konkrét sor most arra utal, hogy a HTML5-szabvány szerint készítettük el az oldalt. Ez a sor nagyon fontos. Ha hiányzik, akkor a böngészőprogram nem biztos, hogy helyesen fogja megjeleníteni az oldalt.

A következő sorban a `<html>` taget helyezzük el, amely jelzi a böngészőnek, hogy egy HTML-dokumentumról van szó. A `lang` paraméterben az oldal nyelvét kell megadnunk. Magyar nyelvű tartalom esetén a `hu` értéket kell beírni, angol nyelvű tartalom esetén az `en` értéket.

A HTML-tagen belül két fő egységre tagolódik a tartalom, a **fejre** (`<head>`) és a **törzsre** (`<body>`).

A fej részben (`<head>`) kell beállítani például a karakterkódolást. Itt érdemes az UTF-8 kódolást használni, hogy a magyar ékezetes karakterek megfelelően jelenjenek meg az oldalon. Ehhez viszont a szerkesztőprogramban is ugyanezt a karakterkódolást kell beállítanunk a mentés előtt.

A `<title>` tagben az oldal címét kell megadni, amely megfelelően utal a weblap tartalmára.

A `<body>` és `</body>` tagek közti rész a dokumentumtörzs. Ebben helyezhetjük el a lap tartalmát, a címsorokat, bekezdéseket, képeket, videókat stb.

Fontos! A `<body>` tagben elhelyezett szöveget normál esetben nem tudjuk formázni sortörésekkel, tabulátorokkal, szóközökkel, vagyis ezek hatása nem fog megjelenni a böngészőprogramban. Térközöket a megfelelő tagek használatával, illetve majd a stíluslapok használatával tudunk beállítani.

Készítsünk közösen egy weblapot!

A folytatásban készítsünk egy statikus honlapot, amelynek kapcsán megismerjük a legfontosabb címkéket!

A honlap a golden retriever kutyafajtáról szól. A hozzávalóit a letölthető állományok között találod a *goldi* mappában. Ebben megtalálhatók a felhasználható szövegek, a képek (`kepek` mappa) és a videók (`videok` mappa) is.

Az `index.html` állományban megtaláljuk a korábban bemutatott alapstruktúrát. Nyissuk meg ezt az állományt a kódszerkesztő alkalmazásban!

Módosítsuk az oldal címét (a `<title>` tag tartalmát) úgy, hogy a szöveg utaljon a honlap tartalmára. Pl. így:

```
<title>Golden Retrieverek - Kezdőlap</title>
```

Először a honlap tartalmi részét fogjuk elkészíteni, és csak utána térünk rá arra, hogyan nézzen ki az oldal. Ezért kezdetben még nem a képen látható módon jelenik meg az oldal. Látni fogjuk ugyan a címsorokat, bekezdéseket, a fotókat és a videót is, csak az alapértelmezett megjelenéssel, fehér háttéren, fekete szöveggel.

► Az elkészítendő honlap kezdőlapjának képe

Címsorok használata (<h1>, <h2>…)

A szövegszerkesztés témakörben megtanultuk, hogy a dokumentumot a címsorok segítségével kell tagolnunk annak érdekében, hogy az olvasó könnyen eligazodjon a tartalomban. Ez a weboldalak esetén sincs másként.

Kezdjük azzal, hogy elhelyezünk egy egyes szintű címsort, mindjárt a `<body>` nyitótag után. Erre a `<h1>` tag szolgál.

Szintén adjuk meg most az oldal alcímsorait is! Ezek kerüljenek kettes címsorba a `<h2>` taggel! Összesen hat címsorszintet lehetne használni a HTML nyelvben, vagyis akár a `<h6>` címkét is használhatnánk.

```
<h1>A golden retriever bemutatása</h1>
<h2>Előnyös tulajdonságai</h2>
<h2>Goldi, a kutyám</h2>
```

► A begépelendő kód

Mentsük el az állományt, és nyissuk meg a böngészőprogramban! Ezt megtehetjük úgy, hogy a fájlt ráhúzzuk a böngészőprogram ablakára, de azt is megfelelő, ha duplán rákattintunk az állományra. Utóbbi esetben az alapértelmezett böngészőprogramban jelenik meg az oldal. Ha mindent jól csináltunk, akkor megjelennek a címsorok a böngészőprogramban. A címsorokat a böngésző nagyobb, félkövér betűkkel jeleníti meg.

Megjegyzés beírása a forráskódba (<!-- -->)

Ha szeretnénk megjegyzést tenni a forráskódba, akkor ezt az `<!--` és `-->` jelek közé kell tennünk. A megjegyzésbe írt szöveg nem jelenik meg a böngészőben. Ha átmenetileg ki szeretnénk venni a megjelenítésből egy kódrészletet, akkor a megjegyzés jelekkel körbekeríthetjük. Azt azonban fontos tudnunk, hogy ha valaki megnézi a böngészőprogramban a forráskódot, akkor megtalálja, hogy milyen szöveget, illetve kódot tettünk megjegyzésbe. A megjegyzésbe tett szöveget általában eltérő színnel (pl. zöld) jelzik a kódszerkesztő programok.

A `<body>` tag után tegyünk be egy megjegyzést a kódba, amelyben a saját nevünk szerepeljen!

```
<!-- Nagy Tamara -->
```

Bekezdés és sortörés használata (<p>,
)

Helyezzük el az első bekezdést a címsor alá! Használjuk ehhez a `<p>` taget! Próbáljuk ki a `<br>` tag hatását a bekezdésen belül, amely egy sortörést hoz létre. Ezt a „származik.” szó után helyezzük el. Vigyázzunk, ennek a címkének nincsen záró párja!

Ellenőrizzük a címke hatását a böngészőprogramban úgy, hogy frissítjük az oldalt!

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<p>A golden retriever egy kedves, barátságos kutyafajta, mely Skóciából származik. <br>A fajta tudományos megnevezése: golden retriever – Canis lupus familiaris.</p>
```

Egyszerű szövegformázások (<b>, <i>)

Ismerkedjünk meg két új címkével, amelyekkel alapvető szövegformázásokat végezhetünk el. A `<b>` tag a szöveg félkövér formázására alkalmas. Főként kulcsszavak kiemelésére használatos. Például egy bekezdésen belül azon szavakat érdemes így kiemelni, amelyek fontosak a megértés szempontjából. A honlap látogatója ezen szavakat végigpásztázva gyorsabban áttekintheti a szöveget.

Az `<i>` tag segítségével dőlt betűs formázást állíthatunk be. Akkor használjuk, ha egy szövegrész hangulatban, hangnemben eltér a szöveg többi részétől. Az élőlények latin nevének leírásakor is ezt a címkét kell használni.

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<p>A <b>golden retriever</b> egy kedves, barátságos kutyafajta, mely Skóciából származik. <br>A fajta tudományos megnevezése: golden retriever - <i>Canis lupus familiaris.</i></p>
```

Félkövér és dőlt formázás azonban nem csak a fenti címkék használatával valósítható meg. De vigyázzunk, a hasonló megjelenés mögött, más-más jelentés lehet, amely a böngészőprogramokban nem vehető észre. Ha viszont egy vak felhasználó képernyőolvasó programot használ, akkor máshogy olvashatja fel a program ezeket a szövegeket.

Fontos, hangsúlyos szövegek jelölése (,)

A címkével olyan szövegeket emelhetünk ki, amelyen nagyon fontosak, ezért erősen ki szeretnénk emelni. Ezek megjelenítése éppúgy félkövér formázással történik a böngészőben, mint a <b> tag esetén.

Az címke a hangsúlyos kiemelésre szolgál. Az élő beszédben is sokszor nyomatékosítunk egy-egy szót, amely a mondat jelentését is befolyásolhatja.

Például az alábbi mondatoknak kissé módosul a jelentése, ha más-más szavakat hangsúlyozunk. Tegnap Ábel *hiányzott* az iskolából. Tegnap *Ábel* hiányzott az iskolából.

Az ilyenfajta hangsúlyok jelzésére alkalmas tehát az tag.

A példánkban szereplő bekezdésben jelöljük meg, hogy melyek a fontos, hangsúlyos elemek!

```
<p><strong>Fontos!</strong> Ha gazdi szeretnél lenni, nézz utána,
hogy a <em>felelős</em> kutyatartással kapcsolatban milyen ajánlások
és kötelezettségek vannak!</p>
```

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

Felsoroláslista használata (<ul>, <li>)

A következő téma a kutyusok előnyös tulajdonságairól szól. Ezeket egy felsoroláslistában fogjuk elhelyezni.

Felsoroláslistát úgy készíthetünk, hogy egy <ul> címkét nyitunk meg, és abban helyezzük el a listaelemeket, amelyek <li> tagekbe kerülnek. Az ul az unordered list (felsorolás lista), a li a list item (listaelem) szavak rövidítése.

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<ul>
      <li>Okos, könnyen képezhető</li>
      <li>Szelíd, családbarát</li>
      <li>Nagyon jó vadászkutya</li>
</ul>
```

Sorszámozott lista használata (<ol>, <li>)

A sorszámozott listák nagyon hasonlóan hozhatóak létre, mint a felsoroláslisták, csak ebben az esetben az <ol> taget kell használni. Az ol az ordered list (sorszámozott lista) szavak rövidítése.

A sorszámozott lista általános megadási módja:

```
<ol>
      <li>listaelem</li>
      <li>listaelem</li>
      <li>listaelem</li>
</ol>
```

A példánkban most nincs ilyen lista, ezért folytassuk a képekkel!

Képek beillesztése (<img>)

A honlapokon képeket is elhelyezhetünk az `<img>` tag használatával. A legfontosabb paraméterei:

`src` A kép elérési útvonalát tartalmazza.
`alt` Alternatív szöveg, vagyis a kép rövid tartalmi leírását tartalmazza.
`title` A képhez rendelt cím, amely a böngészőkben akkor jelenik meg, ha a kép fölé visszük az egeret.
`width` A kép szélessége képpontokban.
`height` A kép magassága képpontokban.

A kép forrása (src)

A kép forrását az `src` paraméterben kell megadnunk. Ennek egyik módja az **abszolút útvonalmegadás** (a kép teljes webcímének beírásával), de ennek előfeltétele, hogy a kép már publikálva legyen a világhálón. Pl. http://tiny.cc/8mqblz

Ha a honlapunk alkönyvtárszerkezete alapján adjuk meg az elérési utat, akkor **relatív útvonalmegadást** alkalmazunk. Fontos, hogy annak a HTML-állománynak az elhelyezkedéséhez képest adjuk meg a kép elérési útját, amelyben hivatkozni szeretnénk rá. Lássunk néhány példát!

`src="csaladikep.jpg"`	Ebben az esetben a kép ugyanabban a mappában van, mint a HTML-állomány.
`src="kepek/csaladikep.jpg"`	A kép most a kepek nevű mappában van. Ez a mappa pedig abban a könyvtárban van, amelyben a HTML-állomány.
`src="../kepek/csaladikep.jpg"`	A kép most a kepek nevű mappában van. Ez a mappa pedig egy szinttel feljebbi könyvtárból nyílik, ahhoz képest, ahol a HTML-állomány található.

Fontos! Az a könyvtár, amelyben az `index.html` állományt elhelyezzük, lesz a gyökérkönyvtára a honlapunknak. Ezen belül tetszőleges könyvtárstruktúrát kialakíthatunk, de ne hivatkozzunk az `index.html` állományban (és a vele egy szinten lévő HTML-állományokban) olyan könyvtárra, amely feljebbi szinteken lévő alkönyvtárakra mutat (pl. `../kepek`), mert ez a weblap publikálásánál problémákat okozhat!

Illesszünk be most két képet egymás mellé, azonos magasságban. A magasság legyen 200 képpont! Adjuk meg az alt és title paramétereket is!

```
<img src="kepek/goldi.jpg" height="200" alt="A kutyám, aki egy golden retriever." title="Goldi, a kutyám (felnőttkorában)">
<img src="kepek/kolyok.jpg" height="200" alt="A kutyám kölyökkorában." title="Goldi, a kutyám (kölyökkorában)">
```

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!
Fontos! Amennyiben a szélesség (`width`) és a magasság (`height`) közül csak az egyiket adjuk meg, akkor a böngésző aránytartóan átméretezi a képet. Ha mindkét paramétert megadjuk, akkor vigyázzunk arra, hogy a kép arányai megváltozhatnak, ami a kép torzulásához vezethet.

▶ Kép átméretezése aránytartó és torzított formában

Ábrák, illusztrációk jelölése feliratokkal (<figure>, <figcaption>)

A HTML5-szabványban a `<figure>` címkét használhatjuk arra a célra, hogy illusztrációt adjunk meg. Ezen illusztrációhoz egy feliratot is társíthatunk a `<figcaption>` címkével, amelyet a `<figure>` tagen belül kell elhelyeznünk.

Ha az illusztrációnk egy (vagy több) kép, akkor természetesen az `<img>` taget (vagy tageket) kell elhelyeznünk a `<figure>` tagen belül.

A korábban már beillesztett képeket helyezzük el a `<figure>` tagen belül, és adjuk meg a kép feliratát is. Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<figure>
      <img …>
      <img …>
  <figcaption>Fotók Goldiról</figcaption>
</figure>
```

Fontos! Az illusztrációk nemcsak fotók lehetnek. Grafikonokat, versrészleteket, programkód részleteket és más elemeket is elhelyezhetünk ebben a címkében.

Hivatkozások (linkek) megadása (<a>)

A hivatkozások (linkek) segítségével el tudjuk érni, hogy az oldalunkról egy másik erőforrásra (weboldalra, médiaelemhez, egyéb fájlhoz) lehessen eljutni. Oldalon belüli ugrásra is használhatunk linkeket.
Érdekesség: Az angol link szó láncot jelent magyarul. A lánc pedig sokszor egy horgonyhoz kapcsolódik (pl. a hajók esetén). A horgonyt angolul anchornak nevezik. Ennek a rövidítéséből jött létre az `<a>` tag.

A link létrehozása a következő módon történik: `<a href="url">Link szövege</a>`

Vagyis a `href` paraméterben kell megadnunk a hivatkozott erőforrás URL-jét. A címke nyitó és záró párja közti szöveg lesz a link szövege, amelyre majd rá tudunk kattintani.

Készítsünk egy olyan linket, amelyet követve a honlapunk látogatói még több kutyás képet nézhetnek meg. A weboldal, amelyre hivatkozunk, a http://tiny.cc/goldipix *rövidített címen elérhető Pixabay portál. A* https://pixabay.com/ portálon *jogtiszta, ingyenesen felhasználható fotók találhatók. Csak rá kell keresned egy témára, és máris rengeteg jó minőségű fotót kapsz eredményül.*

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<p>Ha szeretnél még több kutyás fotót nézegetni, látogasd meg a
<a href="http://tiny.cc/goldipix">Pixabay portál - Goldenek</a>
fotó oldalát!</p>
```

Hivatkozás oldalon belül

Azt is megtehetjük, hogy a hivatkozás segítségével **a lap egy adott pontjára ugrunk**. Ehhez azt a helyet el kell neveznünk az `id` paraméterrel. Ennek a névnek egyedinek kell lennie a teljes oldalon belül. Ezt a paramétert bármelyik címkénél felhasználhatjuk. Vigyázzunk, a név kis- és nagybetű-érzékeny.

Az egyedi nevet a link `href` paraméterében úgy kell megadnunk, hogy egy kettős keresztet teszünk elé (`href="#egyedinév"`).

Lássunk egy példát!

Van egy egyes címsorunk, amelynek a `bevezeto` nevet adtuk.

```
<h1 id="bevezeto">Bevezető</h1>
```

Ha erre a pontra szeretnénk ugrani, akkor az `<a>` tag `href` paraméterében ugyanezt a nevet kell megadnunk, de elé egy kettős keresztet (hashtaget) kell írnunk.

```
<a href="#bevezeto">Ugrás a bevezetőre</a>
```

Ha egy másik oldal (pl. magamrol.html) egy pontjára szeretnénk ugrani, akkor az oldal elérhetősége után kell megadnunk a # jelet és a nevet.

```
<a href="magamrol.html#bevezeto">Ugrás a bevezetőre</a>
```

Helyezzünk el a weblapunk alján egy olyan linket, amellyel vissza tudunk ugrani az oldal elejére! Ehhez az egyes címsornak adjunk egy egyedi azonosítót, és a hivatkozásban ugyanezt a nevet adjuk meg!

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<h1 id="eleje">A golden retriever bemutatása</h1>
<p><a href="#eleje">Vissza az oldal tetejére</a></p>
```

Hivatkozás saját oldalra, több oldalból álló weboldalak

Fejlesszük tovább úgy a weboldalt, hogy ne csak egy oldalból álljon. Mentsük el a korábban látott alapstruktúrát tipusok.html néven ugyanabba a mappába, mint ahol az index.html állomány van. Majd módosítsuk az index.html állományt, hogy a másik lapra is el lehessen jutni. Soroljuk be ezt a bekezdést a „menu" osztályba, hogy majd a stíluslap segítségével megváltoztathassuk a kinézetét!

```
<p class="menu"><a href="index.html">Kezdőlap</a> <a
href="tipusok.html"> Típusok</a></p>
```

Videó beillesztése (<video>, <source>)

A HTML5-szabvány jelentős újdonsága, hogy a multimédiás állományok beillesztését (pl. hangok, videók) jelentősen leegyszerűsíti.

Videókat a `<video>` taggel tudunk az oldalba illeszteni. Hogy egy vezérlő eszköztár is megjelenjen (lejátszás, megállítás, hangerő stb. gombokkal), a `controls` paramétert is használnunk kell. Ez egy logikai paraméter, így elég leírnunk a nevét, nem kell értéket adnunk.

A `<video>` tagen belül a `<source>` tag használatával adhatjuk meg a videó forrását. Ezen tag `src` paraméterében kell megadnunk a videó elérési útját. A legnagyobb böngészőtámogatottsága az mp4-formátumú videóknak van, így célszerű ilyen formátumban létrehozunk a videót. Az mp4 formátumú videók esetén a `video/mp4` értéket kell megadnunk a `type` (típus) paraméterben.

A `<video>` esetén is megadható a szélesség (`width`) és magasság (`height`) paraméter csakúgy, mint a képek esetén.

A videó beillesztésének általános módja tehát:

```
<video controls width="" height="">
      <source src="" type="video/mp4">
</video>
```

Próbáljuk ki a videóbeillesztést! A médiaelemek között találunk egy videót Goldiról. Hozzunk létre egy bekezdést, amelyben bevezetjük szövegesen a videó tartalmát, majd ágyazzuk be az oldalra a videót!

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban!

```
<p>A videón is láthatod, hogy Goldi nagyon szeret a földön hempe-
regni, amely miatt általában koszosan jövünk haza a sétából.</p>
<video controls width="400">
<source src="videok/goldiseta.mp4"
        type="video/mp4">
</video>
```

Hangok beillesztése (<audio>, <source>)

Hangokat nagyon hasonló módon lehet beilleszteni, mint a videókat. Ebben az esetben az `<audio>` taget kell használni, és érdemes mp3-formátumú állományokat használni a széles körű böngészőtámogatottság miatt.

A hang beillesztésének általános módja:

```
<audio controls>
      <source src="" type="audio/mpeg">
</audio>
```

Figyelem! Ha videókat, illetve hangokat illesztesz be az oldalra, akkor abban az esetben lesz akadálymentes a weboldal, ha a videó vagy hangállomány tartalmát szövegesen is leírod. Ha a videó rövid, akkor magában a weboldal szövegében is lehet utalni arra, hogy mi látható a videón. Hosszú leírást érdemes külön weblapon elkészíteni, és az adott videó vagy hang alá elhelyezni egy hivatkozást, amely erre a leírásra mutat. Videó esetén egy forgatókönyvszerű leírást érdemes mellékelni, hang esetén pedig az elhangzó szöveget érdemes leírni.

Táblázatok használata (<table>,<tr>,<th>,<td>)

Ha táblázatos formában szeretnénk elhelyezni adatokat a weboldalunkon, akkor több címkével is meg kell ismerkednünk.

A `<table>` tagben kell elhelyeznünk a teljes táblázatot. Ebben megadhatjuk a táblázat feliratát (`<caption>`) is.

A táblázat sorait a `<tr>` címke jelöli. A táblázat soraiban elhelyezhetünk fejléccellákat (`<th>`), amelyek leírják, hogy az adott sorban, illetve oszlopban milyen jellegű adatok vannak. Az adatokat tartalmazó cellákat pedig `<td>` tagek közé kell zárnunk. Az egy sorban lévő elemeket balról jobbra haladva kell leírnunk.

Nézzünk egy példát! Ha az alábbi táblázatot szeretnénk egy honlapon elhelyezni, amely az iskolai futóverseny eredményeit tartalmazza, akkor az itt látható kódot kell használnunk.

Az 5 km-es futóverseny eredményei

Helyezés	Név	Osztály
1.	Kiss Réka	9.C
2.	Hegedűs Ábel	9.A
3.	Horváth Kristóf	9.B

```
<table>
<caption>Az 5 km-es futóverseny eredményei</caption>
<tr>
      <th>Helyezés</th>
      <th>Név</th>
      <th>Osztály</th>
</tr>
<tr>
      <td>1.</td>
      <td>Kiss Réka</td>
      <td>9.C</td>
</tr>
<tr>
      <td>2.</td>
      <td>Hegedűs Ábel</td>
      <td>9.A</td>
</tr>
<tr>
      <td>3.</td>
      <td>Horváth Kristóf</td>
      <td>9.B</td>
</tr>
</table>
```

Feladatok

Készítsük el önállóan az itt látható táblázatot, és helyezzük el a `tipusok.html` oldalon! Ne csak a táblázatot valósítsuk meg, hanem a címsort, a főmenüt és a forrásra mutató hivatkozást is!

A Golden Retriever bemutatása

Kezdőlap Típusok

Típusok

A Golden Retriever típusai

Típus	Típus leírása
Munkatípus	Könnyedebb felépítésű. Vékonyabb , hosszabb fej, sötétebb szőrszín, élénk temperamentum, karcsúbb, rugányosabb, gyorsabb.
Kiállítási típus	Dúsabb, hosszabb szőrzet, robusztusabb felépítés, világosabb szőrszín, határozottabb.
Európai típus	Szőrszíne az arany világosabb árnyalata. Rövidebb szőrű, mint az amerikai. Lába rövidebb az amerikainál, így megjelenése zömökebb. Jól pigmentált. Jó szögelésű elöl, hátul.
Amerikai típus	Orr és fej része finomabb. Színe középarany, rézvörös, de sosem krém. Mellső szögellései szerényebbek, hátul olykor túlszögelt.

Forrás: hu.wikipedia.org/wiki/Golden_retriever

Vissza az oldal tetejére

A legfontosabb HTML5-címkék összefoglaló táblázata

Tag (címke)	Leírás
`<p>Bekezdés</p>`	Bekezdés (*paragraph*)
`<h1>Címsor 1</h1>`	Egyes címsorszint (*heading*)
`<h6>Címsor 6</h6>`	Hatos címsorszint (*heading*) (a közbenső szintek értelemszerűen `h2, h3, h4, h5`)
`<b>szöveg</b>`	Félkövér (*bold*) szöveg (kulcsszavak jelölése)
`<i>szöveg</i>`	Dőlt (*italic*) kiemelés (eltérő hangulatú szöveg, latin nevek)
`<strong>fontos szöveg</strong>`	Fontos, erősen kiemelt (*strong*) szöveg
`<em>hangsúlyos szöveg</em>`	Hangsúlyosan kiemelt (*emphasis*) szöveg
` `	Sortörés (*break*)
`<img src="uri" alt="leírás" width="" height="">`	A megadott forráson (*source*) elérhető kép (*image*) beszúrása leírással (*alternate*), megadott szélességben (*width*), magasságban (*height*).

`<figure>`
`(ide illesztjük be a képet, stb.)`
`<figcaption>Felirat </figcaption>`
`</figure>`
	Illusztráció (*figure*), felirattal (*figure caption*)

`<ul>`
`<li>listaelem</li>`
`</ul>`
	Felsoroláslista (*unordered list*), listaelemmel (*list item*)

`<ol>`
`<li>listaelem</li>`
`</ol>`
	Sorszámozott lista (*ordered list*), listaelemmel (*list item*)

`<a href="uri">Link szövege</a>`	Hiperhivatkozás (link) (*a=anchor, href=hyperlink reference*)

`<video controls width="" height="">`
`<source src="mp4_videó_url" type="video/mp4">`
`</video>`
	Videó (mp4) (*video*) beillesztése, vezérlő eszköztárral (*controls*), megadott szélességben (*width*), magasságban (*height*). A forrás elérhetőségét (*source*) és típusát (*type*) is meg kell adjuk.

`<audio controls>`
`<source src="mp3_hang_url" type="audio/mpeg">`
`</audio>`
	Hangállomány (mp3) (*audio*) beillesztése, vezérlő eszköztárral (*controls*). A forrás elérhetőségét (*source*) és típusát (*type*) is meg kell adjuk.

`<!-- megjegyzés -->`	Megjegyzés (*comment*) elhelyezése a HTML-kódban

`<table>`
`<caption></caption>`
`<tr>`
`<th></th>`
`<td></td>`
`</tr>`
`</table>`
	Táblázat (*table*) elhelyezése:
	• `table`: táblázat
	• `caption`: felirat
	• `tr`: táblázat sora (*table row*)
	• `th`: fejléccella (*table heading*)
	• `td`: adatcella (*table data*)

A stíluslapok (CSS) használata

Most már elkészült a honlapunk tartalma. De hogy kapja meg azt a kinézetet, amelyet a korábbi képen láthattunk? Csatoljunk hozzá egy megfelelő stíluslapot! A css mappában megtalálható az a stíluslap (goldi.css), amely a fotón látható kinézetet eredményezi. Csatoljuk hozzá a HTML-állományhoz.

A stíluslapok csatolása sokféle módon történhet a dokumentumokhoz. Most a külső stíluslap csatolását mutatjuk meg.

A <link> tag használata

A külső stíluslapokat kétféle módon is hozzácsatolhatjuk a HTML-állományokhoz. Az első módszer, hogy használjuk a `<link>` taget a HTML-állomány `<head>` részében, az alábbi módon:

```
<link rel="stylesheet" type="text/css" href="url">
```

A `href` paraméterben kell megadnunk a külső stíluslapállomány elérhetőségét.

A mi esetünkben a CSS-állomány a `css` mappán belül helyezkedik el, `goldi.css` néven. Ezért a következő kódot kell használnunk az oldal `<head>` részében:

```
<link rel="stylesheet" type="text/css" href="css/goldi.css">
```

Próbáljuk ki a kódot, és ellenőrizzük az oldal megjelenését a böngészőprogramban! Ugye megváltozott az oldal kinézete a korábban látott képnek megfelelően?

A @import szabály használata

Külső stíluslapot a CSS-szabvány segítségével is lehet csatolni a dokumentumhoz. Ahhoz, hogy ezt kipróbáljuk, tegyük megjegyzésjelek közé a `<link>` taget vagy töröljük ki!

Ezt a módszert követve, a `<head>` részben el kell helyezni egy lapon belüli stílusleírást a `<style>` taggel. Ennek az elején lehet használni a `@import` szabályt.

```
<style>
   @import url("");
</style>
```

A mi esetünkben a következő kódot kell használnunk:

```
<style>
@import url("css/goldi.css");
</style>
```

Próbáljuk ki a kódot, és mentsük el az eredményt! Frissítsük az oldalt a böngészőben! Ugye most is érvényre jut a csatolt stíluslap?

A stíluslap módosítása

A korábbi fejezetben bemutattuk, hogy hogyan tudunk a CSS-szabvány segítségével kijelölni elemeket (szelektorok), és hogyan adhatjuk meg a tulajdonságokat a deklarációs blokkban. Most ismerkedjünk meg a témával a gyakorlatban is!

Feladatok

Nyissuk meg a `goldi.css` állományt abban a szerkesztőprogramban, amelyet a HTML-állományok szerkesztésére is használtunk. Megnyitás után láthatjuk, hogy milyen CSS-szabályokat alkalmaztunk a stíluslapállományban. A jobb megértés érdekében minden sorhoz egy magyarázatot is elhelyeztünk.

Annak érdekében, hogy stíluslapot tudjunk készíteni, vagy meglévőt módosítani, az alábbi táblázatban elhelyeztük a leggyakrabban használt tulajdonságokat és magyarázatukat.

`text-align`	A szöveg vízszintes igazításának módja. Értékei: `left` (balra), `center` (középre), `right` (jobbra), `justify` (sorkizárt módon). Példák: `text-align:center;` `text-align:right;`
`font-size`	A betű mérete. Megadható például képpontokban (px), vagy akár százalékosan (%) is. Használható az em mértékegység is, amellyel az aktuális betűméret valahányszorosát állíthatjuk be. Példák: `font-size:16px;` `font-size:120%;` `font-size:1.5em;`
`font-weight`	A szöveg félkövér megjelenítése. Tipikus értéke a `'bold'`. Ha vissza akarjuk állítani a normál megjelenést, használjuk a `'normal'` értéket! Példák: `font-weight:bold;` `font-weight:normal;`
`font-style`	A szöveg dőlt megjelenítésére szolgál. Tipikus értéke az `'italic'`. Ha vissza akarjuk állítani a normál megjelenést, használjuk a `'normal'` értéket! Példák: `font-style:italic;` `font-style:normal;`
`font-family`	A betűtípus beállítására szolgál. Konkrét betűtípusokat fel lehet sorolni, a végén egy általános betűcsaládot kell megadni (pl. `serif` = talpas betűk, `sans-serif` = talp nélküli betűk). Ha a betűtípus neve szóközt tartalmaz, akkor aposztrófjelek közé kell tenni. Példák: `font-family:Arial,Verdana,sans-serif;` `font-family:'Times New Roman',Times,serif;`
`border` `border-top` `border-right` `border-bottom` `border-left`	Az elemet körbevevő szegély beállítására szolgál. Megadhatjuk a szegély vastagságát képpontokban (px), a szegély stílusát (`solid` = folytonos, `dotted` = pontozott, `dashed` = szaggatott, `double` = dupla stb.), valamint a színét. Ha a `border` tulajdonságot használjuk, akkor mind a négy oldali szegély ugyanolyan lesz. Ha nem ezt akarjuk, akkor használhatjuk csak az adott oldalra vonatkozót (`top`=felső, `right`=jobb, `bottom`=alsó, `left`=bal) Példák: `border-top:1px solid blue;` `border:4px double #8b0000;`
`border-radius`	A szegély lekerekítettségét állítja be. Például: `border-radius:20px;`
`float`	Az adott elemet lehet balra vagy jobbra lebegtetni. A lebegtetés azt jelenti, hogy a környező elemek körbefolyják az adott elemet. Például: `float:left;` `float:right;`

`padding`
`padding-top`
`padding-right`
`padding-bottom`
`padding-left`
	Kitöltés megadása, ami a tartalom és az ezt körbevevő szegély közti térközt jelenti. Belső margónak is hívják. Ha a `padding` tulajdonságot használjuk, és egy értéket adunk meg, akkor mind a négy oldali beállítás ugyanakkora lesz. Ha nem ezt akarjuk, akkor használhatjuk csak az adott oldalra vonatkozót.

`margin`
`margin-top`
`margin-right`
`margin-bottom`
`margin-left`
	A margó megadására szolgál. A margó az adott elem szegélye és az őt körülvevő többi elem közti távolságot jelenti. Ha a `margin` tulajdonságot használjuk és egy értéket adunk meg, akkor mind a négy oldali beállítás ugyanakkora lesz. Ha nem ezt akarjuk, akkor használhatjuk csak az adott oldalra vonatkozót. Az elemek középre igazítására is a margóbeállítást kell használni, ilyenkor `'auto'` értéket kell adni a bal és jobb margónak. Ekkor a `margin-left:auto;` és `margin-right:auto;` tulajdonságokat kell beállítani.

`color`	A szöveg színének beállítására szolgál. Megadhatunk színnevet (pl. blue), de akár RGB színkódot is. A képszerkesztő programokban gyakran a színkódot hexadecimális kóddal is megtaláljuk, amelynek az elején egy # karakter áll. Az alábbi példában ugyanazon szín megadását láthatjuk különböző módokon: `color:darkred; color: #8b0000; color: rgb(139,0,0);`
`background-color`	Háttérszín beállítása. A színek hasonlóan adhatóak meg, mint a szövegszín esetén.
`width`	Az elem szélességét állítja be. A méret megadható képpontokban és akár százalékban is. Példák: `width:600px; width:90%;`
`height`	Az elem magasságát állítja be. Hasonlóan állítható be, mint a szélesség.
`min-width` `min-height`	Az elem minimális szélességét, illetve magasságát jelenti, amely alá nem lehet átméretezni a böngészőprogramban.
`max-width` `max-height`	Az elem maximális szélességét, illetve magasságát jelenti, amely fölé nem méretezi át a böngésző az elemet.
`background-image`	A háttérkép beállítására szolgál. A háttérkép elérhetőségét az `url('')` szövegen belül kell megadni, az aposztrófok között. Például: `background-image:url('kepek/mancs.png');`
`background-repeat`	A háttérkép ismétlődésének beállítására alkalmas. Az értékei a következők lehetnek: `repeat` (mozaikszerűen ismétlődő), `no-repeat` (nem ismétlődő), `repeat-x` (csak vízszintes irányba ismétlődő), `repeat-y` (csak függőleges irányba ismétlődő). Például: `background-repeat:no-repeat;`
`background-position`	háttérkép bal felső sarkának koordinátái. Először az x tengelyre vonatkozó, utána az y tengelyre vonatkozó értéket kell beállítani. Például: `background-position:0 20px;`

Feladatok

Készítsünk másolatot a `goldi.css` állományból, és csatoljuk ezt az új állományt a már elkészült HTML-oldalakhoz. Módosítsuk az oldal kinézetét saját elképzeléseinknek megfelelően. Próbáljunk ki új színsémákat, igazítási módokat!

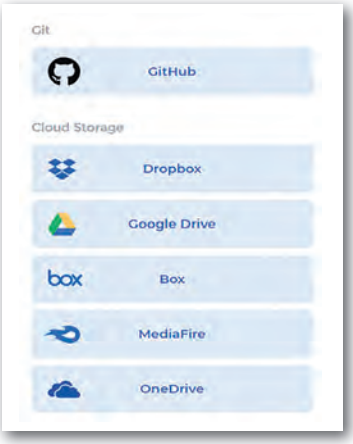
A statikus honlap publikálása

Ahhoz, hogy az általunk készített statikus honlap mindenki számára elérhető legyen a világhálón, publikálni kell azt egy webes kiszolgálón (webszerveren). Ez azt jelenti, hogy egy segédprogram segítségével csatlakozni kell a webszerverhez, és a megfelelő mappájába fel kell tölteni az összes állományt.

Vannak olyan ingyenes szolgáltatások (pl. fast.io), amelyek lehetővé teszik a statikus honlapok publikálását egy webcímen.

Ilyenkor a regisztráció után meg kell adnunk, hogy milyen webcímen szeretnénk elérhetővé tenni a honlapunkat, majd az ahhoz tartozó összes állományt a portálon ismertetett módon fel kell töltenünk.

Ne feledjük, a kezdőlapot `index.html` néven kell elneveznünk!



▶ A fast.io szolgáltató sokféle tárhelyhez tud doménnevet rendelni

Feladatok, kérdések

1. Derítsük ki, hogy a saját iskolánkban van-e lehetőség a honlapok publikálására. Ha nincs, akkor publikáljuk az elkészített honlapot egy ingyenesen elérhető szolgáltató segítségével (pl. fast.io, gitHub.com)!
2. Három-négy fős csoportokban készítsünk el tetszőleges témában egy honlapot az alábbi elvárások szerint:
   - A weblap legalább két oldalból álljon! Mindegyik oldalba legyen beillesztve egy menü, amellyel el lehet jutni a többi oldalra.
   - Az oldalcímeket precízen töltsük ki mindegyik oldal esetén!
   - A tartalom legalább öt bekezdésből álljon! A bekezdésben lévő szöveget formázzuk meg a tanult címkék segítségével!
   - Legyen az oldalon felsorolás lista és sorszámozott lista is!
   - A tartalom címsorokkal legyen tagolva! Használjunk több címsorszintet!
   - Legyen egy oldalmenü az oldalon, amellyel az oldalon belüli címsorokra rá lehet ugrani!
   - Legyen az oldalon beillesztve kép és videó! A beillesztésnél figyeljünk az akadálymentességi irányelvekre! (Szorgalmi feladatként hangot is elhelyezhetünk az oldalon, ha van a témához kapcsolódó hangállomány.)
   - Legyen egy olyan táblázat beillesztve, amely legalább három sorból és legalább két oszlopból áll! A táblázatnak legyen felirata! Használjunk fejléc- és adatcellákat is!
   - Csatoljunk a honlaphoz egy stíluslapot!
   - Dolgozzunk jogtiszta forrásból! Felhasználhatunk saját készítésű képeket, videókat, hangállományokat is.
   - Minden felhasznált forrást tüntessünk fel és linkeljünk be!
   - Publikáljuk a weblapot, és a webcímet osszuk meg a többi csoporttal is!

A munka felosztásánál ügyeljünk arra, hogy minden tagnak jusson olyan feladat, amelyben tartalmat kell összegyűjteni, és azt a HTML nyelven le kell írni.

D Táblázatkezelés

A táblázatkezelés alapjai

A „számolótábla” alapötlete

A szövegszerkesztő program mellett a leggyakrabban használt irodai alkalmazás a táblázatkezelő. Míg azonban a szövegszerkesztés egyidős az írásbeliséggel, addig a táblázatkezelő programok megjelenése a számítógépekhez kötődik.

A „számolótábla” alapötlete az, hogy a táblázat a kiszámolható adatok helyett a számítás módját megadó képletet tartalmazza. Például ha egy táblázatban a megvásárolandó tej mennyisége és egységára szerepel, akkor a fizetendő összeget nem kézzel számítjuk ki és írjuk be a táblázatba, hanem ezt maga a számolótábla végzi el a megadott képlet *(mennyiség és egységár szorzata)* alapján.

Az első széles körben elterjedt táblázatkezelő program, a *VisiCalc,* amelyet két egyetemista, Dan Bricklin és Bob Frankston készített, 1979-ben jelent meg, utolsó változatát 1985-ben adták ki. A táblázatkezelő elvi felépítése azóta sem változott.

	A	B	C	D
1	Termek	Mennyi	Egysar	Fizeten
2	Tej	2	3.6	7.2
3	Kenyer	1	3.6	3.6
4	Kifli	3	.4	1.2
5	Sportsz	2	1.2	2.4
6	Fozokol	.2	28	5.6
7	Osszese			20

	A	B	C	D
1	Termék	Mennyiség	Egységár	Fizetendő
2	Tej	2	319 Ft	638 Ft
3	Kenyér	1	249 Ft	249 Ft
4	Kifli	3	29 Ft	87 Ft
5	Sportszelet	2	89 Ft	178 Ft
6	Főzőkolbász	0,2	1 290 Ft	258 Ft
7	**Összesen**			**1 410 Ft**

▸ Napi bevásárlás VisiCalckal (1979-ből) és Excellel (2020-ban). A képletek majdnem ugyanazok, például a D2-es cellában VisiCalc esetén még +B2*C2 volt (ma =B2*C2-t írunk).

A táblázatkezelő programok az adatok tárolása és látványos megformázása mellett **lehetővé teszik számítások automatikus elvégzését, diagramok készítését.** A mai táblázatkezelő programok támogatják ezenkívül üzleti előrejelzések készítését, egyenletek közelítő megoldását, illetve tartalmaznak adatbázis-kezelő funkciókat is.

Az asztali programok mellett egyre elterjedtebbek az online elérhető számolótáblák (pl. *Google Táblázatok*) és a mobilappok. Tanulás közben ezért a hangsúlyt az általános alapelvekre és funkciókra kell helyeznünk, hogy könnyen tudjunk közöttük váltani.

Ebben a fejezetben főleg két asztali táblázatkezelő programot, a *Microsoft Excelt* és a *LibreOffice Calcot* mutatjuk be példaként. Ha egy funkciót a két programban eltérően kell használni, akkor a szövegben először a *Microsoft Excelre* vonatkozó megoldás szerepel, ezt követi zárójelben a *LibreOffice Calc-beli* megvalósítás.

1. példa: A színjátszófesztivál bevétele

Az iskolai színjátszófesztivál gálaelőadását az Irka Iskola a Városi Színház nagytermében tartja. Ebben a példában az előadás bevételeit összesítjük.

Írjuk be az adatokat az ábrának megfelelően! Az adatok bevitele közben ügyeljünk arra, hogy a szomszédos cellákba a tabulátorgomb vagy a kurzormozgató billentyűk használatával lépjünk! Egy cella tartalmát utólag is javíthatjuk, ha például előbb kettőt kattintunk rá az egér bal gombjával.

Egy **cellára** vagy **cellatartományra** az oszlop- és sorazonosítók „sakkszerű” megadásával hivatkozhatunk. Az ábrán például éppen a *C5*-ös cellába írjuk be a karzatra szóló jegyek árát, míg a jegyek eladott darabszámát a hely függvényében az *A2:B5* tartomány tartalmazza.

	A	B	C	D
1	Hely	Darab	Ár (Ft)	Összeg
2	Földszint	124	3200	
3	Erkély	68	4200	
4	Páholy	16	7200	
5	Karzat	42	22	
6	Összesen			

► Adat bevitele a C5-ös cellába

	A	B	C	D
1	Hely	Darab	Ár (Ft)	Összeg
2	Földszint	124	3200	=B2*C2
3	Erkély	68	4200	
4	Páholy	16	7200	
5	Karzat	42	2200	
6	Összesen			

► A D2-es cella másolása

Az adatok beírása után meghatározzuk, hogy mennyi volt a bevétel helytípus szerint. Például a földszintre eladott jegyek után a bevételt a *B2*-es és a *C2*-es cellák tartalmának összeszorzásával kapjuk, így a *D2*-es cellába a következő **képletet** írjuk:

```
=B2*C2
```

A *D2-es* cella alatti *D3:D5* tartományba hasonló képletet kell bevinnünk. Ezt megtehetjük egyenkénti beírással is, de célszerűbb a képletet lefelé másolni: egérrel „megfogjuk” a *D2*-es cella jobb alsó sarkán lévő kis négyzetet, majd lefelé húzzuk a *D5*-ös celláig. Ekkor **a cellákba az eredeti képlet a másolás irányának megfelelően módosítva kerül**, vagyis ezúttal a sorok azonosítója mindig eggyel nő.

Végül határozzuk meg a teljes bevételt a *D6*-os cellában! Erre megoldás lehet az

```
=D2+D3+D4+D5
```

képlet beszúrása, de látható, hogy ez sok cella esetén nem használható. A táblázatkezelők ezért az ilyen esetekre **függvények** használatát teszik lehetővé. Például a *D2:D5* tartomány celláinak összegét a *SZUM* függvény segítségével határozhatjuk meg a következő képlettel:

```
=SZUM(D2:D5)
```

A SZUM függvény a zárójelben szereplő tartomány celláinak értékét adja össze. Ha a tartományban szöveget tartalmazó cellák is vannak, azokat nem veszi figyelembe.

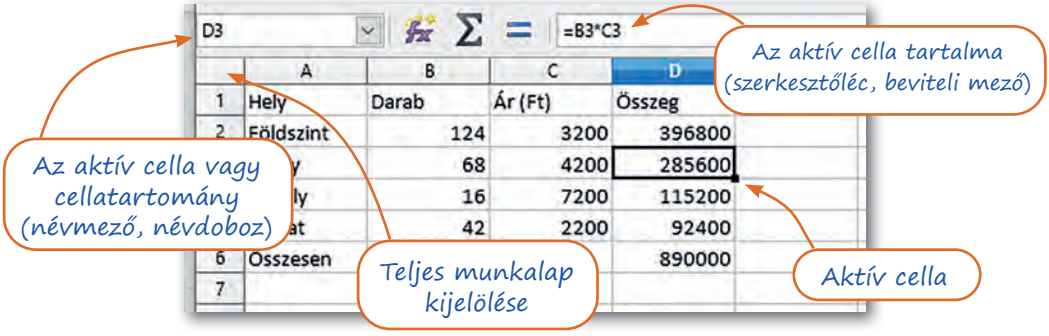
A táblázatkezelés alapfogalmainak áttekintése

A táblázatkezelő programok a táblázat **oszlopait** *A*-val kezdve betűkkel, **sorait** egytől kezdve számokkal azonosítják. Egy adott **cellára** az oszlop- és sorazonosítójával hivatkozhatunk, például *C5*.

Egy téglalap alakú **cellatartományt** a bal felső, valamint a jobb alsó sarkában lévő cellák azonosítójával adunk meg, pl. *A1:B5*.

Adatainkat a táblázatkezelő program egy **munkafüzetben** tárolja, amely egy vagy több **munkalapból** áll. A munkalapok között az alkalmazásablak alsó részén válthatunk. Az adatok mentésekor a teljes munkafüzet kerül a fájlba.

A munkafüzetben mindig egy olyan cella van, amelyikbe adatokat tudunk bevinni a billentyűzetről, ezt **aktív cellának** nevezik.



▸ Néhány fontos elem a táblázatkezelők képernyőjén

Az oszlopok azonosítása *Z* után kétbetűsre vált, rendre *AA*, *AB* stb.

Egy vagy több teljes sor vagy teljes oszlop is alkothat tartományt, ennek megadása például *B:B* vagy *B:G*.

Egy tartomány kijelölése többnyire az egérrel történik, de megtehetjük azt a SHIFT gomb nyomva tartása mellett a kurzormozgató billentyűkkel is. Egy sort vagy egy oszlopot kijelölhetünk, ha a megfelelő azonosítóra kattintunk. A teljes munkalap kijelöléséhez pedig az oszlopazonosítók előtt és a sorazonosítók fölött lévő téglalapra kell kattintanunk.

A munkalap nem korlátlan méretű. Az üres munkalapot (vagy egy kitöltött tartományt) a CTRL + kurzormozgató nyilak lenyomásával járhatjuk körbe.

Új munkalapot a képernyő alsó részén lévő ⊕ ikonra (illetve a + ikonra) kattintva adhatunk a munkafüzethez. A munkalap nevét a jobb egérgombbal rákattintva, helyi menü *Átnevezés* (illetve a *Munkalap átnevezése*) menüponttal módosíthatjuk.

Ha a cella **szöveget** tartalmaz, akkor annak tartalmát a táblázatkezelő program **balra** zárja automatikusan, ha pedig **számot**, akkor **jobbra**. Ha tehát egy szám típusú adat balra kerül, akkor azt valószínűleg hibásan írtuk be.

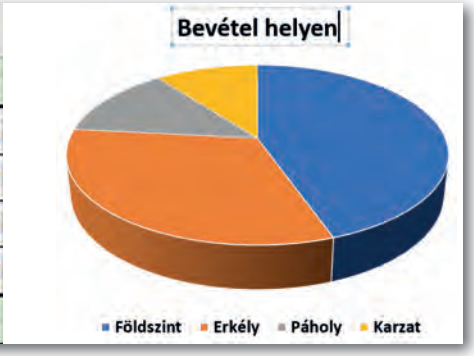
A képletek egyenlőségjellel kezdődnek. **A képlet eredményét a táblázatkezelő programok az adatok változása esetén automatikusan újraszámolják.** Az újraszámítást az F9 funkciógomb lenyomásával is kiválthatjuk.

2. példa: A táblázat formázása és diagram beillesztése

A táblázatot a szövegszerkesztésben már megismert módon formázhatjuk: a cellákat **szegélyezhetjük**, megváltoztathatjuk a **háttérszínét**, módosíthatjuk a **betűk formátumát** és a cellák **igazítását**.

	A	B	C	D
1	Hely	Darab	Ár (Ft)	Összeg
2	Földszint	124	3200	396800
3	Erkély	68	4200	285600
4	Páholy	16	7200	115200
5	Karzat	42	2200	92400
6	Összesen			890000

► A táblázat megformázása betűformázás, igazítás, háttérszín, szegély alkalmazásával



► A diagram összetevőit módosíthatjuk: a cím átírása

A legtöbb ember számára áttekinthetőbbek az adatok, ha azokat diagramon ábrázoljuk. Szemléltessük ezért a bevételt a hely függvényében egy kördiagramon!

Ehhez az *A1:A5* és *C1:C5* tartományokat kell megadnunk. Jelöljük ki ehhez az *A1:A5* tartományt, majd nyomjuk le a CTRL gombot, és annak nyomva tartása közben jelöljük ki a *C1:C5* tartományt is!

A kijelölt tartományhoz a táblázatkezelő program automatikusan elkészíti a kördiagramot a *Beszúrás* > *Diagramok* > *Kör vagy perecdiagram* (illetve *Beszúrás* > *Diagram* > *Diagramtípusok* > *Torta*) pont választásával.

A kész diagramot az egérrel mozgathatjuk, a sarkok húzásával átméretezhetjük, illetve az egyes összetevőkre kattintva, azok megjelenését módosíthatjuk.

Cellák, sorok, oszlopok elrendezése

Szöveges adatoknál gyakori, hogy a szöveg nem fér el a cellában. Amíg a szomszédos cella üres, addig látszólag ott folytatódik, de ha oda is adat kerül, akkor a „kilógó” rész nem látszik. Ilyenkor *csökkenthetjük a betű méretét, vagy növelhetjük az oszlop szélességét* az oszlop betűjelét tartalmazó téglalap jobb szélének húzásával. Mivel ez a probléma igen gyakori, a táblázatkezelő programok lehetővé teszik azt is, hogy a szöveget automatikusan több sorra tördelve helyezzük el a cellában, például a *Sortöréssel több sorba* (illetve a *Szöveg tördelése*) ikonnal.

A címeket előnyös a táblázat fölött középre zárni. Legegyszerűbb, ha ilyenkor a megfelelő cellákat egyesítjük: ekkor a szöveg automatikusan középre igazodik. Ezt megtehetjük a megfelelő cellák kiválasztása után például a *Cellaegyesítés* (illetve a *Cellák egyesítése és középre zárása*) ikonnal.

Számok esetén nyilván nem járható út, hogy sokjegyű szám esetén annak egy része lemaradjon. Ezért ilyen esetben a táblázatkezelő program figyelmeztet: a cellába egy ###### karaktersorozat kerül.

Sorok és oszlopok beszúrását, törlését vagy méretének számszerű megadását a megfelelő menüpontokkal végezhetjük. Ilyenkor értelemszerűen ki kell jelölnünk azt a sort vagy oszlopot, amelyet törölni akarunk, vagy amely elé szeretnénk új sort vagy oszlopot beszúrni.

Irka Iskola - papírgyűjtés			
Osztály	Osztályfőnök neve	Létszám (fő)	Papír (kg)
8.a	Nagy-Bende Bc	32	########

► Milyen megoldásokat látunk a cella tartalmának elrendezésére az ábrán?

Ezeket a műveleteket például a *Kezdőlap* > *Cellák* csoport ikonjaival (illetve a méretezést a *Formátum* menü, a beszúrást és törlést pedig a *Munkalap* menü pontjaival) végezhetjük el.

3. példa: Fagylaltgombócok ára

Készítsünk táblázatot, amely megkönnyíti a fagylaltárus feladatát: leolvasva meg tudja róla mondani, mennyit kell fizetnie a vevőnek a gombócok száma alapján!

A táblázat feliratait készítsük el a bal oldali ábrának megfelelően! A táblázatnak tartalmaznia kell a gombócok számát, ezt gyorsan megcsinálhatjuk kitöltéssel. Írjuk be a *B5:B6* tartomány celláinak tartalmát, majd jelöljük ki, és húzzuk a *B6*-os cella jobb alsó sarkában lévő négyzetet lefelé!

	A	B	C
1			
2		Egységár:	250 Ft
3			
4		Darab	Összeg
5		1	
6		2	
7			
8			
9			

► Kitöltés az egér húzásával

	A	B	C	D
1				
2		Egységár:	250 Ft	
3				
4		Darab	Összeg	
5		1	250 Ft	=B5*C2
6		2	0 Ft	=B6*C3
7		3	#ÉRTÉK!	=B7*C4
8		4	1 000 Ft	=B8*C5
9		5	0 Ft	=B9*C6

► Relatív hivatkozás az egységárra

	A	B	C	D
1				
2		Egységár:	250 Ft	
3				
4		Darab	Összeg	
5		1	250 Ft	=B5*$C$2
6		2	500 Ft	=B6*$C$2
7		3	750 Ft	=B7*$C$2
8		4	1 000 Ft	=B8*$C$2
9		5	1 250 Ft	=B9*$C$2

► Abszolút hivatkozás az egységárra

Mivel a fizetendő összeg az egységár és a gombócok számának szorzata, írjuk be a *C5*-ös cellába a megfelelő képletet: *=B5*C2*, majd másoljuk lefelé! Jóllehet a képlet egy gombócra helyes értéket ad, a többi esetben helytelen, vagy a program hibát jelez.

Nézzük meg, milyen képletek születtek másoláskor! A cellacímek a képlet másolásakor a másolás irányának megfelelően módosultak. A gombócok száma esetén ez így helyes is, de az egységár esetén nem, annak nem lett volna szabad változnia.

Az egységár cellacímét tehát másolás előtt rögzíteni kell, ezt a következő formában tehetjük meg: *=B5*$C$2*. Másolás után ezúttal helyes eredményt kapunk.

A képletben lévő relatív cellahivatkozás a képlet másolásakor a másolás irányának megfelelően módosul, míg az abszolút cellahivatkozás nem változik.

Relatív cellahivatkozás esetén a táblázatkezelő nem a hivatkozott cellacímet tárolja, hanem annak az aktuális cellához viszonyított helyzetét. (Például: a gombócok száma egy cellával balra van). Abszolút cellahivatkozás esetén viszont a program a cella tényleges helyét tárolja.

Abszolút cellahivatkozás megadásakor a sor-, illetve oszlopkoordináta elé $ jelet kell tennünk. Például az *=B5*$C$2* képletben a *B5* relatív, míg a *$B$2* abszolút hivatkozás.

Szükség lehet táblázatok előállításához a vegyes cellahivatkozásra, amikor az egyik koordináta abszolút és a másik relatív. Például: *$B5* vagy *C$2.*

Az ábrán a 10-es szorzótábla egy részletét látjuk. A táblázat vegyes cellahivatkozással készült.

A *C3*-as cellába a *B3*-as és *C2*-es cellák szorzata van. De vajon a cellahivatkozás melyik koordinátája abszolút és melyik relatív?

Mivel az egyik tényező a *B* oszlopban, a másik pedig a *2*. sorban van, ezért ezt a két koordinátát kell rögzítenünk. Így tehát a *C3*-as cellába az *=$B3*C$2* képlet került.

	A	B	C	D	E	F	G	H
1								
2			1	2	3	4	5	6
3		1	1	2	3	4	5	6
4		2	2	4	6	8	10	12
5		3	3	6	9	12	15	18
6		4	4	8	12	16	20	24

► Példa vegyes cellahivatkozásra

Feladatok

1. A bal oldali ábra az Irka Iskola 9. b osztályának éves pénzforgalmát tartalmazza. Az osztálypénz szeptember 1-jén 0 Ft-tal indult, utána havi bontásban látjuk a bevételek és kiadások összegét. Készítsük el a táblázatot! A hónapok oszlopát kitöltéssel vigyük be. Milyen képlet került a *E4*-es cellába?

	A	B	C	D	E
1		*Osztálypénz - havi bontásban*			
2			Bevétel	Kiadás	Egyenleg
3		Aug			0
4		Szept	80000	7800	72200
5		Okt	120000	13700	178500
6		Nov	40000	23780	194720
7		Dec	20000	13300	201420
8		Jan	10000	1800	209620

	A	B	C	D	E
1	Benzinár:	380	Fogyasztás:	7,4	l/100 km
2					
3	*mikor*	*honnan*	*hová*	*út (km)*	*költség (Ft)*
4	hétfő	Budapest	Kaposvár	190	5342,8
5	kedd	Kaposvár	Répcelak	179	5033,48
6	szerda	Répcelak	Tófej	102	2868,24
7	csütörtök	Lenti	Siófok	164	4611,68
8	péntek	Siófok	Tyukod	427	12007,24
9	Összesen:				29863,44

2. A jobb oldali ábrán egy utazó ügynök havi útnyilvántartását látjuk. Az ügynök a saját gépkocsiját használja, amely 100 km-en a *D1*-es cellában szereplő mennyiségű benzint fogyasztja. A benzin ára a *B1*-es cellában van. Milyen képlet kerül az *E4*-es és *E9*-es cellába? Az *A4:A8* tartományban kitöltést alkalmazzunk!
3. A Fibonacci-sorozat első két tagja 1, majd minden következő tag az előző két tag összege. Képlet segítségével jelenítsük meg a sorozat első 20 tagját!

Számok, szövegek, logikai kifejezések kezelése

4. példa: A teremszépségverseny értékelése *(tizedesjegyek, ezres tagolás)*

Az Irka Iskola diákönkormányzata minden évben teremszépségversenyt hirdet, a legjobb osztály egy egynapos kirándulást kap jutalmul. A tantermeket három szempont szerint értékelik, minden szempontra legfeljebb 10 pont kapható.

Töltsük le a *terem* nevű fájlt a tankönyv weboldaláról, vagy gépeljük be az adatokat a szürke hátterű cellák kivételével!

	A	B	C	D	E	F	G	H	I	J	K
1	Osztály	**9.nyek**	**9.a**	**9.b**	**10.a**	**10.b**	**11.a**	**11.b**	**12.a**	**12.b**	**Átlag**
2	Tisztaság	5	6	10	9	9	9	4	6	7	7,22222
3	Dekoráció	9	8	10	3	10	7	7	9	9	8
4	Növények	8	2	9	7	7	5	8	8	8	6,88889
5	**Összesen**	22	16	29	19	26	21	19	23	24	22,1111
6											
7		Legjobb eredmény:		29		Második legjobb:		26		Osztályok száma:	
8		Legrosszabb eredmény:		16		Utolsó előtti:		19			10

▶ Az iskolai teremszépségverseny statisztikai adatai

A leggyakoribb statisztikai függvények, a ***SZUM(), ÁTLAG(), MAX(), MIN()*** megadják a zárójelben lévő tartományban szereplő számok összegét, átlagát, legnagyobb értékét, illetve legkisebb értékét.

A ***NAGY***(*tartomány; n*) és a ***KICSI***(*tartomány; n*) függvények kétparaméteresek, megadják a tartomány n-edik legnagyobb, illetve legkisebb értékét.

Tehát a *K2*-es cellába az *=ÁTLAG(B2:J2)*, a *B5*-ös cellába az *=SZUM(B2:B4)*, a *D7*-es cellába az *=MAX(B5:J5)*, a *D8*-as cellába pedig az *=MIN(B5:J5)* képlet kerül.

Hasonlóképp a *H7*-es cella tartalma az *=NAGY(B5:J5; 2)* képlet, a *H8*-as celláé pedig az *=KICSI(B5:J5; 2)* képlet. A *MAX* és *MIN* helyett is alkalmazhattuk volna a *NAGY* és *KICSI* függvényeket, például az *=NAGY(B5:J5; 1)* képlettel.

Az osztályok számát többféle módon is megadhatjuk. Például a *B5:J5* tartomány számokat tartalmazó celláinak megszámolásával: *=**DARAB**(B5:J5)*, vagy a *B2:J2* tartományban lévő nem üres cellák számának megadásával: *=**DARAB2**(B2:J2)*.

A *K2:K5* tartományban az átlagok különböző számú tizedesjeggyel szerepelnek. Jobban összehasonlíthatók az adatok, ha ezeket azonos számú, tipikusan két tizedesjegy pontossággal jelenítjük meg.

A tizedesjegyek számát például a *Tizedeshelyek növelése, csökkentése*  (illetve a *Tizedesjegy hozzáadása, törlése* ) ikonokkal változtathatjuk meg.

Fontos tudnunk, hogy a táblázatkezelő program továbbra is a pontos értékeket tárolja, és csupán megjelenítéskor formáz adott tizedesjegyre.

Legyen például két cella tartalma 3,33 és 2,22, ezek összege 5,55. Ha azonban mindhárom adatot tizedesjegyek nélkül jelenítjük meg, akkor látszólag a 2 és 3 összege 6 lesz. Ha az értékeket ténylegesen kerekíteni szeretnénk, akkor a *KEREKÍTÉS* függvényt kell használnunk: például az *A2*-es cellában: *=KEREKÍTÉS(A1; 0)*.

A ***KEREKÍTÉS*** függvénynek két paramétere van: az első a kerekítendő szám, a második pedig a tizedesjegyek előírt száma. Például 12,357 kerekítése 1 tizedesjegyre: *KEREKÍTÉS*(12,357; 1) = 30,4. A tizedesjegyek száma lehet negatív is, ilyenkor a függvény a tizedesjeltől balra kerekít, például *KEREKÍTÉS*(12,345; -1) = 10.

	A	B	C	D
1	2,22	3,33	5,55	C1: =A1+B1
2	2	3	6	C2: =A2+B2
3	A2-ben is 2,22 van			

	A	B	C	D
1	2,22	3,33	5,55	C1: =A1+B1
2	2	3	5	C2: =A2+B2
3	A2: =KEREKÍTÉS(A1;0)			

► Az adatok látszólagos kerekítése formázással és pontos kerekítése függvény alkalmazásával

Nagyobb számok beírásánál gyakran használt formátum, hogy a számokat ezres tagolással és 2 tizedesjegy pontossággal jelenítjük meg, például: 12 345,67. Ezt általában egy ikonra való kattintással is elérhetjük, például a *Kezdőlap > Szám > Ezres csoport* (illetve *Eszköztár > Számformátum: Szám*).

Fontos tudnunk, hogy amíg Magyarországon az ezres elválasztójel a szóköz, a tizedesjel pedig a vessző, addig angol nyelvterületen az ezres elválasztójel a vessző, a tizedesjel pedig a pont.

5. példa: Az átlagkereset változása *(százalék és pénznem)*

Az alábbi ábrán a magyarországi átlagkereset növekedését látjuk 2000 és 2005 között. Töltsük le a *kereset* nevű fájlt a tankönyv weboldaláról, vagy gépeljük be az 1. és 2. sor, valamint az *A* oszlop adatait!

Határozzuk meg a 3. sorban, hogy az előző évihez képest mennyivel nőtt az átlagkereset; a 4. sorban azt, hogy az előző évihez képest hány százalékkal nőtt az átlagkereset, végül az 5. sorban azt, hogy 2000-hez képest hány százalékkal nőtt az átlagkereset! Végül formázzuk meg a táblázatot a mintának megfelelően!

	A	B	C	D	E	F	G
1	**év**	**2000**	**2001**	**2002**	**2003**	**2004**	**2005**
2	**átlagbér**	87 645 Ft	103 553 Ft	122 482 Ft	137 193 Ft	145 520 Ft	158 343 Ft
3	**növekedés**		15 908 Ft	18 929 Ft	14 711 Ft	8 327 Ft	12 823 Ft
4	**növekedés aránya**		18%	18%	12%	6%	9%
5	**... 2000-hez képest**		18%	40%	57%	66%	81%

► A havi bruttó átlagkereset változása Magyarországon 2000 és 2005 között a Központi Statisztikai Hivatal adatai alapján

A 2. sorban az adatok pénznem formátumban szerepelnek. Ezt elérhetjük úgy, hogy az adatok bevitelekor beírjuk a forint jelét is, de választhatjuk a *Kezdőlap* > *Szám* > *Könyvelési számformátum* (illetve *eszköztár* > *Számformátum: pénznem*) ikont is. A megjelenő tizedesjegyek számát csökkentsük nullára!

Az átlagkereset változásának megadása az előző évihez képest a *C3*-as cellában: *=C2-B2*; a változás aránya az előző évihez képest a *C4*-es cellában: *=(C2-B2)/B2.* Ha az 5. sorban másolható képletet szeretnénk alkalmazni, akkor a 2000-es adatokat rögzíteni kell, így a *C5*-ös cellába a következő képlet kerül: *=(C2-$B$2)/$B$2*.

A 4. és 5. sorban az adatok százalék formátumban vannak megadva. Az adatok kijelölése után ezt például a *Kezdőlap* > *Szám* > *Százalék* % (illetve *eszköztár* > *Számformátum: Százalék* %) ikonnal, majd a tizedesjegyek számának növelésével állíthatjuk be.

Számformátumok

A táblázatkezelő programok **a tárolt számokat többféle formátumban meg tudják jeleníteni** (például pénznem, dátum formában), de ettől a tárolásuk változatlan.

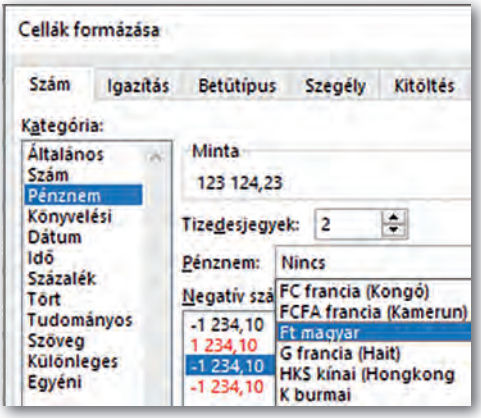
A megjelenő formátumok gyakran országonként is eltérő szabványt követnek (például pénznem jele, tizedesjel, dátumforma), ezért a táblázatkezelők **az adott nyelvterületen használt formátumokat az operációs rendszer beállításaiból veszik.**

Megjelenítéskor megadhatjuk a tizedesjegyek számát. Ilyenkor az adat adott tizedesjegyre formázva jelenik meg, de a program továbbra is a pontos értéket tárolja.

Az előző példában már megvizsgáltunk két számformátumot: a százalékot és a pénznemet. **A százalék formátum a szám százszorosát jeleníti meg**, így például 16-ból 1600% lesz.

A pénznemet gyakran megjeleníthetjük *könyvelői formátumban* is, ebben az esetben *a tizedesjelek egymás alá kerülnek*, függetlenül a tizedesjegyek számától.

Ugyan ikonok segítségével a leggyakoribb számformátumokat közvetlenül beállíthatjuk, de az összes lehetőség eléréséhez általában be kell lépnünk a megfelelő párbeszédablakba, például a *Kezdőlap* > *Szám* > *Számformátum* (illetve a *Formátum* > *Cellák*) ponttal.



► Számformátumok beállítása (Microsoft Excel)

6. példa: Vendégszoba kiadása (*dátum és idő*)

Címer Gábor nyaranta rendszeresen kiadja siófoki nyaralóját, a bérlők a kereslettől függően eltérő összeget fizetnek az ott töltött éjszakák után. A takarítást egy helyi vállalkozó órabérben végzi, a fizetendő összeg szigorúan arányos a munkaidővel nem egész számú órák esetén is.

Töltsük le a tankönyv weblapjáról a *nyaralo* nevű fájlt, vagy vigyük be az adatokat az ábra alapján, a mintának megfelelően, de a szürke hátterű cellák kivételével!

Határozzuk meg először a vendégek után fizetendő összeget! Írjuk a *C3*-as cellába az *=B3-A3* képletet! A cellában láthatóan az eltelt éjszakák száma jelenik meg, így adódik, hogy az *E3*-as cellába az *=C3*D3* képlet kerüljön.

Ha ugyanezt elvégezzük a takarítónak fizetendő adatokkal, meglepő eredményre jutunk. Míg az *I3*-as cellában az *=H3-G3* képlet helyes eredményt mutat, addig a *J3*-as cellába írt *=I3*$J$1* képlet eredménye csupán 833 Ft lesz a várt érték helyett.

A megoldáshoz meg kell ismerkednünk a táblázatkezelők dátum- és időkezelésével. A hétköznapi életben használt rendszer ugyanis nemcsak bonyolult, de kultúrkörönként eltérő is; így egy új, logikus, de szokatlan megoldást alakítottak ki.

	A	B	C	D	E	F	G	H	I	J
1	**Vendégek**						**Takarítás (óránként)**			5 000 Ft
2	**Érkezés**	**Távozás**	**Nap**	**Egy éj**	**Fizetendő**		**Kezd**	**Végez**	**Óra**	**Összeg**
3	2020.06.19	2020.06.25	6	8 000 Ft	48 000 Ft		14:20	18:20	4:00	833 Ft
4	2020.06.26	2020.07.04	8	9 500 Ft	76 000 Ft		12:00	15:30	3:30	
5	2020.07.11	2020.07.14	3	12 000 Ft	36 000 Ft		14:00	18:15	4:15	
6	2020.07.15	2020.07.16	1	14 000 Ft	14 000 Ft		10:10	13:25	3:15	

	A	B	C	D	E	F	G	H	I	J
2	**Érkezés**	**Távozás**	**Nap**	**Egy éj**	**Fizetendő**		**Kezd**	**Végez**	**Óra**	**Összeg**
3	44001	44007	6	8000	48000		0,597	0,7639	0,167	833,33333

► Az időszámítás egysége a nap, így két időpont különbsége az eltelt napok száma.
Az alsó ábrán a számformátumok eltávolítása után a ténylegesen tárolt értékeket látjuk.

A táblázatkezelő programok esetén az idő egysége a nap, az időszámítás kezdete pedig 1900. január 1. 0 óra 0 perc. Ez azt jelenti, hogy 1900. január 2-a 2-nek, 1900. február 1-je 32-nek felel meg.

Mivel a dátum egysége a nap, *két dátum különbsége* az eltelt napok számát adja.

Az adott napon belüli időt a szám tizedesjegyei tárolják. Ha a tárolt szám 32,5, akkor az ténylegesen 1900. február 1. 12:00-át jelenti.

Ha meg szeretnénk nézni, hogy a táblázatkezelő program ténylegesen milyen értékeket tárol, állítsuk át a táblázat számformátumát *Általánosra*!

Leolvasható, hogy a példában szereplő 2020. 06. 19. 44 001-nek, míg 2020. 06. 25. 44 007-nek felel meg, a két szám különbsége valóban 6.

A jobb oldali táblázatban láthatjuk, hogy az idő egysége is a nap. Mivel egy nap 24 órából áll, a *J3*-as cellába az *=I3*$J$1*24* képletet kell írnunk.

Végül érdemes megismerkednünk a ***SZÖVEG*** függvénnyel, amely a dátum egyes részeit szövegesen jeleníti meg. Két paramétere van: az első a dátum, a második a formátumkód. Például *=SZÖVEG(2020.02.04;"nnnn")* eredménye „kedd”. Az „n” kód a nap számát adja egy vagy két jegyre (4 vagy 14), az „nn” pontosan két jegyre (04), az „nnn” a rövidített nevet (K), végül az „nnnn” a teljes nevet (kedd). Hasonló módon kaphatjuk a hónap adatait is („h”, „hh” stb.).

További számformátumok, egyéni formátum kialakítása

A *Tudományos számformátum* a matematikában megismert normálalak megfelelője, csupán a kitevőt (exponens) egy E betűvel elválasztva a szám (mantissza) mellé írja. Például a $6{,}022 \cdot 10^{23}$ helyett 6,022E+23 jelenik meg.

A *Tört formátum* az adott tizedes törtet az általunk megadott pontossággal közönséges tört alakban adja meg. Mivel a pontosságot mi állítjuk be, ugyanaz a szám (0,1357) a beállításoktól függően lehet például 1/7, 8/59 vagy akár 65/479 is.

A felhasználó létrehozhat *Egyéni számformátumot* is. Ezt főleg mértékegységeket tartalmazó adatok megadásánál használjuk. Például ha egy adatot km/h egységben szeretnénk megjeleníteni, az alkalmazandó formátumkód: 0" km/h"

A formátumkódot a formázandó cellák kijelölése után például a *Kezdőlap* > *Szám* > *Számformátum* (illetve a *Formátum* > *Cellák*) ponttal elérhető párbeszédablakban az *Egyéni* számformátumokhoz írhatjuk be.

	A	B	C
1	**Idő**	**Sebesség**	**Út**
2	3 h	70 km/h	210 km
3	C2: =A2*B2		

▸ Egyéni formátum beállítása és megjelenése a B2-es cellában (LibreOffice Calc)

7. példa: Továbbjutás a színjátszó fesztiválon *(logikai típus)*

Az Irka Iskola színjátszó fesztiválja több fordulóban zajlik, alkalmanként három előadást mutatnak be. A zsűri az alábbi táblázatnak megfelelően négy szempont szerint értékel, és dönt a továbbjutásról, a különdíjról, illetve hozzájárul a költségekhez is.

A továbbjutás feltétele, hogy az összpontszám 80-nál több legyen. Írjuk a *G2*-es cellába ennek feltételét: *=F2>80*. (Az egyenlőségjel a képletre utal.) A cellában megjelenik az így megadott logikai kifejezés értéke: IGAZ vagy HAMIS. A logikai értékeket a táblázatkezelők középre igazítják.

Az eredmény áttekinthetőbb, ha az IGAZ és HAMIS értékek helyett a *HA* függvény segítségével a „Továbbjut” szöveget, illetve az „” üres szöveget írjuk a cellába:

```
=HA(F2>80; "Továbbjut"; "")
```

	A	B	C	D	E	F	G	H	I	J
1	Előadás	Darabválasztás, szövegkönyv	Jelmezek, díszlet	Jelenetek	Színészi teljesítmény	Összesen	Továbbjut (logikai)	Továbbjut (Ha...)	Különdíj (Logikai)	Költség-térítés
2	Macbeth	23	25	22	24	94	IGAZ	Továbbjut	IGAZ	10 000 Ft
3	Éretlenek	24	12	25	16	77	HAMIS		IGAZ	7 700 Ft
4	A kígyó	24	23	20	19	86	IGAZ	Továbbjut	HAMIS	10 000 Ft
5	G2: =F2>80	H2: =HA(F2>80;"Továbbjut";"")	I2: =VAGY(B1=25; C2=25; D2=25; E2=25)	J2: =HA(F2>80;10000;F2*100)						

▸ A továbbjutás, a különdíj és a költségtérítés meghatározása logikai függvényekkel

A ***HA* függvénynek** három paramétere van: az első egy logikai kifejezés, a második a cella tartalma IGAZ, a harmadik a cella tartalma HAMIS érték esetén.

A táblázatkezelők használják az ***ÉS***, illetve a ***VAGY*** logikai műveleteket. Ezeket függvényként kell beírnunk. Például ha különdíj jár arra az előadásra, amely valamely értékelési szempontra a maximális 25 pontot kapta, akkor egy lehetséges megoldás:

```
=VAGY(B2=25; C2=25; D2=25; E2=25)
```

A *J* oszlopban a költségtérítésre vonatkozó képlet alkalmazását látjuk, ezt a fentiek alapján elemezzük önállóan!

Szöveges adatok kezelése

A táblázatkezelők a szöveges adatokat alapértelmezetten a cellában balra zárják. Ha egy számot tehát a program balra zár, azt valószínűleg hibásan vittük be.

Gyakran van szükségünk arra, hogy a program egy számként is értelmezhető értéket mindenképpen szövegként kezeljen. Például az 1.23 lehet egy termék azonosítója is, de ezt a legtöbb program automatikusan január 23-nak értelmezi. Ilyenkor egy aposztrófot kell az adat elé írni.

Ha egy szöveges adatot cellák értékéből és szövegekből kell összefűznünk, akkor az & műveleti jelet használjuk. Például: *=C1&". helyezett: "&C2*.

Feladatok

1. Hány napot éltünk eddig? Életünk hány százalékában voltunk eddig jelenlegi iskolánk tanulói? Válaszoljunk ezekre a kérdésekre táblázatkezelő program segítségével! (Az aktuális dátumot az *=**MA**()* paraméter nélküli függvény adja meg.)
2. Alkalmazzunk olyan képletet a 4. példában, amely a különdíjban részesülőket *VAGY* függvény használata nélkül választja ki!
3. Csaba bácsi 55. születésnapjára biciklit kapott, azóta naponta gyakorol. Heti útjait mobilapplikációval rögzítette. Készítsük el a táblázatot teljesítményének vizsgálatához! A távolságot km-ben, az időt óra:perc formában adjuk meg.
   a) Határozzuk meg a *D* oszlopban a napi utak átlagsebességét!
   b) Határozzuk meg a *C9*-es cellában a héten megtett teljes utat, a *C10*-esben pedig az egy napra jutó átlagos utat!
   c) Írassuk képlettel a *D9*-es cellába a legnagyobb sebességet!
   d) Az *E* oszlop celláiba kerüljön „+” jel, ha aznap Csaba bácsi 20 km-nél többet tett meg, egyébként kerüljön „-” jel!
   e) A *C* oszlopban alkalmazzunk *km*, a *D* oszlopban pedig *km/h* mértékegységet!
   f) Az *F* oszlopban százalék formában jelenítsük meg, hogy az aznap megtett út hány százaléka volt a teljes hetinek!

	A	B	C	D	E
1	**Indulás**	**Érkezés**	**Út**	**Sebesség**	**Több**
2	14:21	15:35	14,25		
3	10:13	11:55	20,2		
4	9:49	11:22	19,6		
5	15:05	17:19	27,4		
6	14:15	16:13	25,55		
7	14:02	14:53	13,37		
8	13:59	15:57	21,91		

Diagramkészítés

Az alábbi táblázat a világ népességének alakulását szemlélteti az ENSZ adatainak alapján az elmúlt 50 évben, 5 éves periódusonként. Példánkban a táblázat adatait diagramok segítségével szemléltetjük. Az adatokat a *kontinensek* nevű fájlban találjuk.

	A	B	C	D	E	F	G	H	I	J	K	L
1		1970	1975	1980	1985	1990	1995	2000	2005	2010	2015	2020
2	Afrika	363	415	476	549	630	717	811	916	1 039	1 182	1 341
3	Ázsia	2 142	2 401	2 650	2 921	3 226	3 493	3 741	3 978	4 210	4 433	4 641
4	Európa	657	677	694	708	721	727	726	729	736	743	748
5	Észak-Amerika	231	242	254	266	280	294	312	327	343	357	369
6	Latin Amerika	287	323	361	402	443	483	522	558	591	624	654
7	Óceánia	20	22	23	25	27	29	31	34	37	40	43
8	Összesen	3 700	4 079	4 458	4 871	5 327	5 744	6 143	6 542	6 957	7 380	7 795

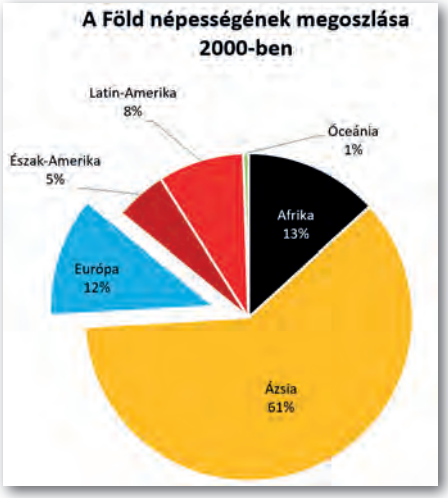
▶ A világ népességének változása az ENSZ adatai alapján
(Forrás: https://population.un.org/wpp/Download/Standard/Population/ Utolsó letöltés: 2020.03.14.)

8. példa: Hogyan oszlott meg az egyes kontinensek között a Föld lakossága 2000-ben?
Az adatokat ezúttal a Föld teljes népességéhez viszonyítva szeretnénk ábrázolni, ezért készítsünk kördiagramot! A diagram elkészítéséhez nemcsak az adatokat, hanem a kontinensek nevét is ki kell jelölnünk, vagyis az *A2:A7* és a *H2:H7* tartomány celláit.

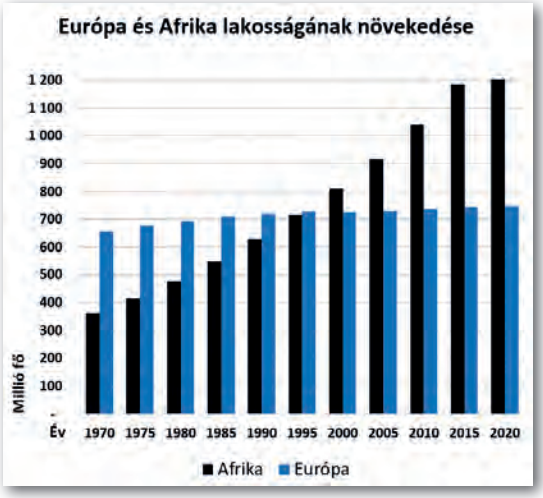
A diagram típusát a *Beszúrás* > *Diagramok* > *Kör- vagy perecdiagram beszúrása* menüponttal (illetve a *Beszúrás* > *Diagram* > *Diagramtündér* eszközzel) adhatjuk meg. Itt többféle lehetőségünk van: szokásos kördiagram (illetve tortadiagram), 3D kördiagram (illetve térhatású tortadiagram), körgyűrűcikkekből álló perecdiagram (illetve fánkdiagram).

Ahhoz, hogy a diagram megfeleljen elképzeléseinknek, további beállításokat is végezhetünk, ezeket programtól függően tehetjük meg. A *Microsoft Excel* menüje például kiegészül két új lehetőséggel: a *Diagramtervezéssel* és *Formátummal*, míg a *LibreOffice Calcban* új menürendszert kapunk, ahol elsősorban a *Beszúrás* és a *Formátum* menüpontok lehetőségeire van szükségünk.

- Megadhatjuk, milyen elrendezésben jelenjenek meg a diagramon az egyes összetevők (cím, jelmagyarázat, adatok), például a *Diagramtervezés* > *Kész elrendezések* lehetőséggel (illetve a *Formátum* menü pontjaival).
- Megváltoztathatjuk a diagram egyes összetevőinek (címek, jelmagyarázatok) betűtípusát, szövegét, színét, illetve az egyes körcikkek színét, szegélyvonalát, ha a megfelelő elemre kettőt kattintunk.
- További összetevőket adhatunk hozzá, például a *Diagramtervezés* > *Diagram összetevő hozzáadása* (a diagramra kattintva > *Beszúrás* menü).
- Az összetevőket (pl.: diagramcím, adatfeliratok, egyes adatokat reprezentáló körcikkek) az egér húzásával mozgathatjuk.
- A diagramot más munkalapra helyezhetjük, például a *Diagramtervezés* > *Diagram áthelyezése* ikonnal (illetve a vágólapon át).
- 3D-s diagramok esetén pedig a diagramot térben elforgathatjuk, módosíthatjuk a perspektívát.



► Adatok viszonyítása egymáshoz és az összegükhöz képest kördiagram segítségével



► Két adatsor adatainak egymáshoz való viszonyítása oszlopdiagrammal

9. példa: Hogyan növekedett Európa és Afrika lakossága ebben az időszakban?

Ezúttal – feliratokkal együtt – az *A1:L2* és *A4:L4* tartományt kell kijelölnünk. Az adatokat oszlopdiagramon ábrázoljuk, mivel az lehetővé teszi két adatsor adatainak egymáshoz való viszonyítását.

Alapvetően két lehetőségünk van. *Csoportosított oszlopdiagram* (illetve *normál oszlopdiagram*) esetén az azonos évhez tartozó oszlopok egymás mellé kerülnek, így jól látható, hogy bár mindkét kontinens népessége nőtt, Afrikáé nagyobb ütemben. *Halmozott oszlopdiagram* esetén a két oszlop egymásra kerül.

Ezúttal is létrehozhatunk térhatású diagramot, illetve a két tengely felcserélésével úgynevezett sávdiagramot.

Oszlopdiagramok esetén további beállítási lehetőségeink vannak.

- A tengelyekhez tengelycímeket adhatunk hozzá (Év, Millió fő), például a *Diagramtervezés* > *Diagram-összetevő hozzáadása* > *Tengelycímek* pont (illetve a diagram kiválasztása esetén a *Címek* gomb) segítségével.
- Megadhatjuk a skálabeosztást, vagyis a tengelyen mettől meddig, milyen osztásközzel jelenjenek meg az adatok. Ezúttal az adattartomány az *y* tengelyen 0-tól 1200-ig terjed, 100-as lépésközzel. A beállítást például a tengelyfeliratokra jobb gombbal kattintva a *helyi menü* > *Tengely formázása* ablakban érjük el. (Néha a diagramon az adatokat az adott tengelyen fordított sorrendben szeretnénk látni, ezt is itt állíthatjuk be.)
- Az oszlopdiagramok megjelenését esztétikusabbá teheti, ha az oszlopok alapértelmezett távolságát (átfedés) és távolságát a szélesség módosításával (térköz) beállítjuk. Az ábrán az átfedés 0, mivel az összetartozó oszlopértékek összeérnek. A beállítást elérhetjük például az oszlopokra kattintva a *helyi menü* > *Adatsorok formázása* menüpontjával.

10. példa: Európa lakosságának változása

Az ábrán Európa lakosságának időbeli változását grafikonon ábrázoltuk, ezt pontdiagram beszúrásával értük el. Ezúttal az *A1:L1* és *A4:L4* tartomány celláit jelöltük ki, és a *Pont görbített vonalakkal és jelölőkkel* (illetve a *Pontok és vonalak*, vonaltípus*: Sima*) lehetőséget választottuk.

A diagramon az oszlopdiagramnál megismert változásokat állíthatjuk be a tengelyeken, de lehetőségünk van a vonal és a jelölők megválasztására is.

Érdekes lehetőség az adatokból matematikai úton számítható trendvonal felvétele, amely lehetővé teszi az értékek jövőbeni becslését. Ezt például az egér jobb gombjával a görbére kattintva a *Trendvonal felvétele* (illetve *Trendvonal beszúrása*) ponttal tehetjük meg. Beszúráskor kiválaszthatjuk a görbére illesztendő függvényt, ezúttal például a pontokhoz legközelebb álló egyenest kértük, és előírtuk a trendvonal egyenletének megjelenítését is.

Megjegyzés: Az adatokat ábrázolhatjuk vonaldiagramon is, de tudnunk kell, hogy a vonaldiagram az oszlopdiagram logikáját követi, például az *x* tengelyen az értékek a sávok közepén jelennek meg.

▶ Grafikon trendvonallal és a trendvonal egyenletével

A diagramok áttekintése

Kördiagram: Csak egy adatsort mutat be, de lehetővé teszi az egyes adatok egymáshoz és az adatok összegéhez való viszonyítását is. Látványos változata a 3D kördiagram (illetve térhatású tortadiagram).

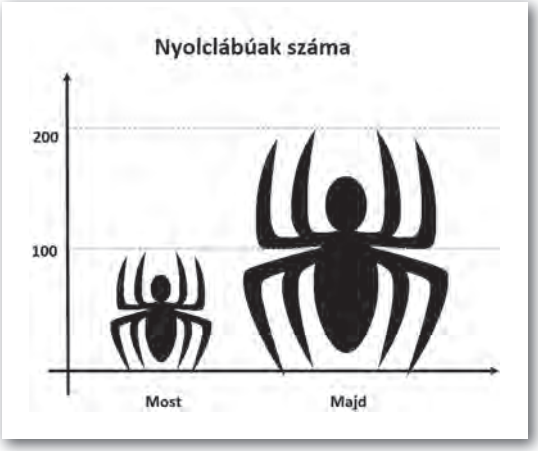
Oszlopdiagram: Lehetővé teszi egy vagy több adatsor időbeli változásának követését és az adatok összehasonlítását is. Változatai a 3D (illetve térhatású) oszlopdiagram, valamint a sávdiagram.

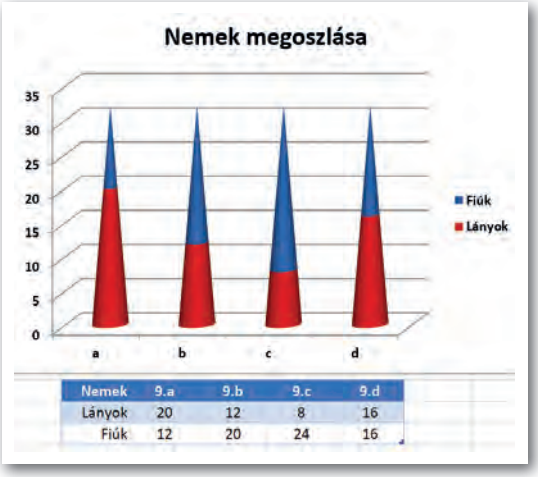
Pontdiagram: Adatsorok változását a természettudományokban megszokott módon, grafikonon szemlélteti. Lehetőségünk van például trendvonalak felvételére is.

Feladatok

1. Ábrázoljuk grafikonon a világ népességének változását ebben az időszakban! Milyen trendvonal illeszkedik hozzá a legjobban? Állítsunk be a diagramterület hátterének egy képet a földgömbről!
2. Ábrázoljuk térhatású halmozott oszlopdiagramon Észak-Amerika és Latin-Amerika lakosságának változását ebben az időszakban! Latin-Amerika adatai világos-, Észak-Amerika adatai pedig sötétvörös színben jelenjenek meg!

3. Ábrázoljuk 3D területdiagramon valamennyi kontinens lakosságának növekedését úgy, hogy az egyes kontinensekhez tartozó adatsorok ne takarják egymást!
4. Ábrázoljuk a *Számok, szövegek, logikai kifejezések kezelése* című fejezet első példájában szereplő adatokat (Teremszépségverseny) sugárdiagram (illetve hálódiagram) segítségével!
5. Sajnos az adatok diagramokon történő szemléltetése sok esetben lehetővé teszi az adatok megtévesztő ábrázolását is. Az alábbi ábrákon erre látunk példákat.
   a) Milyen eszközöket használtak az egyes diagramok készítői, és mit akartak „bizonyítani”?
   b) Készítsük el *A világ népessége 2000-ben* című diagramot jól!





Problémamegoldás táblázatkezelővel

11. példa: Belépési adatok generálása

Egy cégnél a helyi hálózatba való belépéshez minden új dolgozó felhasználói nevet és egy egyszer használatos jelszót kap. A felhasználói név a vezeték- és utónév első két betűje, kiegészítve a születési év utolsó két jegyével, a jelszó pedig az utónév utolsó három karaktere, kiegészítve egy háromjegyű véletlen számmal.

Töltsük le a tankönyv weboldaláról az *account* nevű fájlt, vagy gépeljük be a táblázat első sorát és első két oszlopát!

	A	B	C	D	E	F	G	H	I
1	**Név**	**Szülnap**	**Vez**	**Szk**	**Utó**	**Év**	**Felh**	**Vél**	**Jelszó**
2	Füle Imre	1988.10.28	Fü	5	Im	1988	Fülm88	832	mre832
3	Megkő Tóni	1998.04.28	Me	6	Tó	1998	MeTó98	752	óni752
4	Gaz Ella	1992.05.07	Ga	4	El	1992	GaEl92	331	lla331
5	Ügyet Lenke	2001.01.01	Üg	6	Le	2001	ÜgLe1	370	nke370

▸ Belépési adatok generálása

Mintapéldánk megoldásához szükségünk lesz az alábbi szöveg-, illetve dátumkezelő függvényekre:

***BAL**(szöveg; darabszám)*, ***JOBB**(szöveg; darabszám)*: A *szöveg* bal, illetve jobb oldaláról adott *darabszám*ú karaktert ad vissza.

***KÖZÉP**(szöveg; sorszám; darabszám)*: Visszaad adott *darabszámú* karaktert a *szöveg* adott *sorszámú* karakterétől kezdve.

***SZÖVEG.KERES**(mit; hol; honnan)*: Megkeresi a *mit* szöveg első előfordulását a *hol* szövegben a *honnan* sorszámú karaktertől indulva. Az utolsó paraméter elhagyható.

***ÉV**(dátum)*, ***HÓNAP**(dátum)*, ***NAP**(dátum)*: Megadják, hogy az adott *dátum* melyik év hányadik hónapjának hányadik napjára esik. Visszafelé: az *év, hónap, nap* számokból a dátumot a ***DÁTUM**(év; hónap; nap)* függvény határozza meg.

***VÉL**()*: Egyenletes eloszlású véletlenszerű számot ad vissza, amely 1-nél kisebb, de 0-nál nem kisebb. Nincs paramétere.

***VÉLETLEN.KÖZÖTT**(alsó; felső)*: Az *alsó* és a *felső* érték közé eső véletlenszerű egész számot ad vissza.

A fenti függvények felhasználásával a vezetéknév első két betűjét a *C2*-es cellában az *=BAL(A2; 2)* képlet adja.

Az utónévhez meg kell keresnünk a szóköz helyét a *D2*-es cellában az *=SZÖVEG.KERES(" "; A2)* képlettel, ezt követi az utónév két karaktere az *E2*-es cellában: *=KÖZÉP(A2; D2+1; 2)*.

A születési évet az *F2*-es cellában az *=ÉV(B2)* képlet adja, ennek utolsó két jegyét kell hozzáfűznünk az előző karakterekhez a felhasználói név előállításához: *=C2&E2&MARADÉK(F2; 100)*.

A jelszóhoz a *H2*-es cellában egy háromjegyű véletlenszámot generálunk a *=VÉLETLEN.KÖZÖTT(100; 999)* képlettel, amihez hozzá kell fűznünk a név utolsó 3 karakterét, így a jelszó az *I2*-es cellában az *=JOBB(A2;3)&H2* képlettel áll elő.

Érdemes megemlíteni, hogy a véletlenszám-generátorok egy matematikai képlettel állítják elő az egyes tagokat, vagyis ezek valójában nem véletlenszerűek; ugyanakkor eloszlásuk hasonló a ténylegesen elvégzett kísérletekhez.

12. példa: Dolgozat értékelése

A 9. c osztály matematikadolgozatot írt, az eredményeket az *A:B* oszlopban látjuk. A dolgozatokat az elért pontszámok alapján a matematikatanár a *G4:H8* tartományban szereplő segédtábla felhasználásával számítja át osztályzatokra, például, aki 50 pontot elért, de 67 pontot már nem, az közepes (3) osztályzatot kap.

Töltsük le a *dolgozat* nevű fájlt a tankönyv weblapjáról, vagy gépeljük be az első sort, az első 2 oszlopot és a segédtáblát!

	A	B	C	D	E	F	G	H
1	**Név**	**Pontszám**	**HOL.VAN**	**INDEX**	**FKERES**		**Dió Dina**	67
2	Alma Ajna	84	4	jó	jó			
3	Barack Bardó	85	5	jeles	jeles		**Segédtábla**	
4	Citrom Ciceró	66	3	közepes	közepes		0	elégtelen
5	Cseresz Nyeste	67	4	jó	jó		33	elégséges
6	Dió Dina	67	4	jó	jó		50	közepes
7	Eper Erik	49	2	elégséges	elégséges		67	jó
8	Füge Fürtike	50	3	közepes	közepes		85	jeles

▶ Példa keresőfüggvények használatára

A *C* oszlopban az osztályzatok számszerű értékét látjuk, vagyis azt, hogy a *B* oszlopban szereplő pontszám hányadik sávba esik a *G4:G8* tartománnyal megadott sávok közül. Ezt a *HOL.VAN* függvénnyel érhetjük el. Példánkban a *C2*-es cellában a *=HOL.VAN(B2; $G$4:$G$8)* másolható képlet szerepel. Ez a függvény végighaladva a *G4:G8* tartományon megáll a 4. elemnél (67), mert ez a *B2*-es cella értékénél (84) még kisebb, de a következő érték már nagyobb; és visszaadja a sáv sorszámát (4).

A *D* oszlopban az osztályzatok neveit látjuk. Ezeket a *H4:H8* tartomány elemei adják meg: az 1. elem az elégtelen, a 2. az elégséges stb. Ezúttal az *INDEX* függvényt használjuk, amely visszaadja egy tartományból az adott sorszámú elemet. Példánkban a *D2*-es cella az *=INDEX($H$4:$H$8; C2)* másolható képletet tartalmazza. Ez a „jó” szöveget adja vissza, mert *C2* értéke 4, és ez a *H4:H8* tartomány negyedik eleme.

A két képletet akár egymásba is ágyazhatjuk, így nem szükséges egy külön cellát igénybe vennünk. Ebben az esetben a *D2*-es cella tartalma:

```
=INDEX($H$4:$H$8; HOL.VAN(B2; $G$4:$G$8))
```

A keresőfüggvények egy gyakori alkalmazását látjuk a *G1:H1* tartományban. Ha a *G1*-es cellába beírjuk egy tanuló nevét, akkor a *H1*-es cellában az adott tanuló pontszáma jelenik meg.

A feladat ezúttal is megoldható az *INDEX…HOL.VAN* függvénypárossal. Az *A2:A8* tartomány celláiban a *HOL.VAN* függvénnyel megkeressük a kiválasztott tanuló helyét: *HOL.VAN(G1; A2:A8; 0)*, majd az „ennyiedik” adatot kiválasztjuk a *B2:B8* tartományból az *INDEX* függvény segítségével:

=INDEX(B2:B8; HOL.VAN(G1; A2:A8; 0)

A *HOL.VAN* függvénynek most van egy harmadik paramétere is, a 0. Ezzel érjük el azt, hogy pontosan a *G1*-es cellában lévő adatot keresse a függvény a megadott tartományban. Most – a sávokban való kereséssel ellentétben – nem kell rendezettnek lennie a megfelelő tartománynak.

Az eddigi feladatok megoldhatók más függvényekkel is, ezek közül a leggyakrabban használt az *FKERES*. Például az *E2*-es cellába a következő képletet is írhatjuk:

```
=FKERES(B2; $G$4:$H$8; 2)
```

Ez a következőt jelenti: keresse meg a program az *B2*-es cella értékét a *G4:H8* tartomány első oszlopában, és adja vissza a segédtábla *2*. oszlopában lévő értéket. Az *FKERES* függvényt használhatjuk a *H1*-es cellában is, ebben az esetben is egy 0 paraméterrel kell jelezni a pontos keresést: *=FKERES(G1; A2:A8; 2; 0)*

Az ***INDEX****(tartomány; sor; oszlop)* függvény a *tartomány* megadott sorszámú *sorából* visszaadja az adott *oszlopban* lévő értéket. Ha az *oszlop* értéke 1, akkor a második paraméter elhagyható.

A ***HOL.VAN****(érték; tartomány; egyezés)* függvény megkeresi az *értéket* a *tartományban*, és visszaadja, hogy az hányadik. Ha pontos egyezést írunk elő, akkor az *egyezés* értéke 0. Ha azt a sávot keressük, amelyikbe az adott érték tartozik, akkor ez a paraméter elhagyható, de ekkor a tartománynak növekvően rendezettnek kell lennie.

Az ***FKERES****(cella; tartomány; k; egyezés)* függvény megkeresi a *cella* értékét a *tartomány* első oszlopában, és kapott sorból visszaadja a tőle jobbra, a *k.* oszlopban lévő értéket. Pontos keresésénél negyedik paraméterként 0-t kell megadnunk.

13. példa: Feltételhez kötött statisztikai számítások

Egy osztály életében rendszeresen visszatérő kérdés, hogy az angolosok vagy a németesek teljesítenek jobban, gyakran készülnek olyan statisztikák, hogy mennyit hiányoznak az angolos fiúk, németes lányok stb. Az ilyen feladatokhoz a statisztikai függvények (*SZUM*, *ÁTLAG* stb.) egy olyan változatát kell használnunk, amelyeknél megadhatunk feltételeket is.

Töltsük le a tankönyv weboldaláról az *osztstat* nevű fájlt! A táblázatban a tanulók neve, neme, az általuk tanult első idegen nyelv, a félévi évfolyamdolgozat eredménye és az első félévben mulasztott órák száma szerepel az *A1:E26* tartományban.

Először határozzuk meg, hogy hány fiú és hány lány jár az osztályba! A feladatban a *B2:B26* tartományban kell megszámolnunk a „fiú”, illetve a „lány” szó előfordulásait. Ezt megtehetjük például a *DARABTELI* függvénnyel, amelynek két paramétere van: az első a vizsgált tartományt, a második a feltételt adja meg. Például a fiúk számát az

=DARABTELI(B2:B26;"fiú") képlettel határozhatjuk meg a *H3*-as cellában. A feladat megoldható másolható képlet alkalmazásával is: *=DARABTELI($B$2:$B$26;G3)*

A *H7*-es cellában arra vagyunk kíváncsiak, hogy hány tanuló nem mulasztott egyetlen órát sem. Ehhez meg kell számolnunk az *E2:E26* tartományban az üres cellák számát, amit a *DARABÜRES* függvénnyel tehetünk meg: *=DARABÜRES(E2:E26)*.

A *DARABTELI* függvényben egyenlőtlenséget is vizsgálhatunk, ilyenkor a relációs jelet idézőjelek közé kell tenni, és az & jellel kell hozzáfűzni a hivatkozott értéket, cellát vagy függvényt. Például azok száma, akik 10-nél több órát hiányoztak a *H8*-as cellában: *=DARABTELI(E2:E26;">"&10)*, illetve az átlagosnál többet hiányzók száma a *H9*-es cellában: *=DARABTELI(E2:E26;">"&ÁTLAG(E2:E26))*

	A	B	C	D	E	F	G	H
1	**Név**	**Nem**	**Nyelv**	**Jegy**	**Mulasztás**			
2	Alma Ajna	lány	angol	4	2		A tanulók száma:	
3	Barack Bardó	fiú	angol	3	4		fiú	12
4	Citrom Ciceró	fiú	német	5			lány	13
5	Cseresz Nyeste	lány	német	4	14			
6	Dió Dina	lány	angol	4	5		Mulasztások száma:	
7	Eper Erik	fiú	angol	4	48		nem hiányzott	6
8	Füge Fürtike	lány	német	5	2		10-nél több óra	6
9	Galagonya Gala	lány	angol	4	3		átlagnál több óra	5
10	Grép Gerda	lány	német	3	66			
11	Josta Jolán	lány	angol	2	3		Angolos fiúk adatai:	
12	Kivi Kitti	lány	angol	4			tanulók száma	5
13	Körte Kötöny	fiú	német	5			együttes mulasztás	58
14	Licsi Liem	fiú	német	4	1		osztályzatok átlaga	3,80

► Feltételes számítások (a teljes osztály adatai az A1:E26 tartományban vannak)

Ha több feltétel együttes teljesülését vizsgáljuk, akkor a *DARABTELI* helyett a *DARABHATÖBB* függvényt kell használnunk. Ennek páros számú paramétere van: az első tartományt követi az első feltétel, majd a második tartományt a hozzá tartozó feltétel, és így tovább. A *H12*-es cellában az angolos fiúk számát keressük, vagyis azt, hogy a *C2:C26* tartományban hányszor szerepel az „angol” szó, miközben a *B2:B26* tartományban a „fiú” szó van: *=DARABHATÖBB(C2:C26;"angol";B2:B26;"fiú")*.

Hasonlóan használjuk a *SZUMHATÖBB és* az *ÁTLAGHATÖBB* függvényeket, amelyek az első paraméterben megadott tartomány adataival végzik a megfelelő statisztikai számításokat. A feltételeket az ezt követő paraméterek tartalmazzák a *DARABHATÖBB* függvénynél megadott módon. Például a *H13*-as cellában az *=SZUMHATÖBB(E2:E26; C2:C26;"angol"; B2:B26;"fiú")* képlet adja meg az angolos fiúk hiányzásainak összegét, míg ugyanezen tanulók évfolyamdolgozatának átlagát az *=ÁTLAGHATÖBB(D2:D26; C2:C26; "angol"; B2:B26; "fiú")* képlettel kapjuk. E két függvényhez hasonlóan használhatjuk a néhány programban már elérhető *MAXHA* és a *MINHA* függvényeket is.

Ha csupán egyetlen feltételt írunk elő, akkor használhatjuk a *SZUMHA* és az *ÁTLAGHA* függvényeket is, de esetükben ügyelni kell arra, hogy az összegzendő, illetve átlagolandó tartományt a harmadik paraméter, míg a kritériumot az első két paraméterük adja meg.

A ***DARABHATÖBB****(tart1; felt1; tart2; felt2; …)* megszámolja, hány olyan sor van egy adattáblában, amelyekre a *tart1* tartományában a *felt1*, a *tart2* tartományában a *felt2* … feltétel teljesül. Ha egyetlen feltételünk van, akkor használhatjuk helyette a ***DARABTELI***, az üres cellák megszámolására pedig a ***DARABÜRES*** függvényt is.

A ***SZUMHATÖBB****(tart; tart1; felt1; tart2; felt2; …)* egy adattábla *tart* tartományában összeadja azokat az elemeket, amelyekre a *tart1* tartományában a *felt1*, a *tart2* tartományában a *felt2* … feltétel teljesül. Hasonlóan használhatjuk az ***ÁTLAGHATÖBB***, a ***MAXHA*** és a ***MINHA*** függvényeket.

14. példa: Adatok kiemelése feltételes formázással

Ha fel akarjuk hívni a figyelmet bizonyos adatokra, akkor érdemes azokat a táblázatban eltérő formázással kiemelni. A táblázatkezelők a megadott feltételeknek megfelelő adatokat az előírt formátummal automatikusan ki tudják emelni, az adatok módosulása esetén pedig automatikusan megváltoztatják. Ezt a funkciót **feltételes formázásnak** nevezzük. Az egyszerűbb feltételeket általában kiválaszthatjuk a menü segítségével is, de az összetettebb feltételeket képlettel kell megadni, így most mi is ezzel a lehetőséggel fogunk megismerkedni.

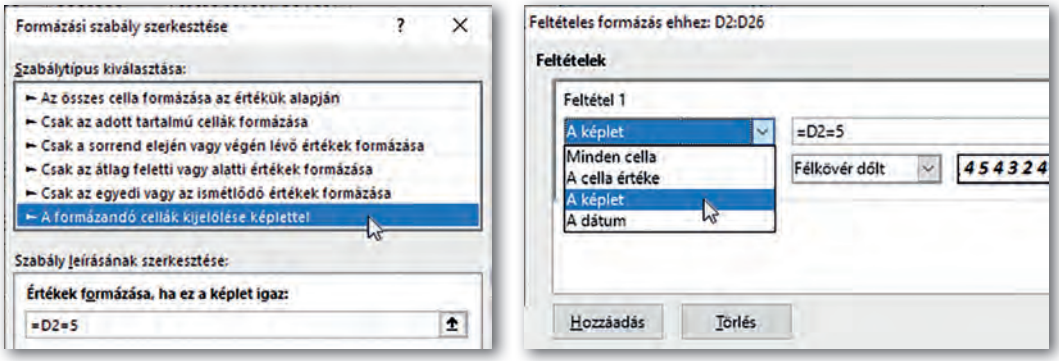
Emeljük ki az osztály adatait tartalmazó táblázatban a jeleseket félkövér és dőlt betűstílussal!

Jelöljük ki a *D2:D26* tartományt, majd válasszuk a *Kezdőlap* > *Feltételes formázás* > *Szabályok kezelése* (illetve a *Formátum* > *Feltételes formázás* > *Kezelés*) pontot! A megjelenő ablakban kattintsunk az *Új szabály* > *A formázandó cellák kijelölése képlettel* pontra (illetve a *Hozzáadás* gombra kattintva a megjelenő ablakban gördítsük le a *Feltétel 1* alatti listát, és kattintsunk *A képlet* lehetőségre)! Itt adjuk meg a képletet a tartomány első sorára vonatkoztatva: *=D2=5*, majd a *Formátum* gombra kattintva (illetve az *Alkalmazandó stílus* melletti lista legördítésével) állítsuk be a megfelelő formátumot! A táblázatkezelő program a megadott képlettel a teljes tartományt kitölti, vagyis a 3. sorban már az *=D3=5* feltételt vizsgálja, és így tovább.

Megjegyzések: Kicsit zavaró lehet, hogy a képletben két egyenlőségjel van, de az első csak kötelező bevezető eleme egy képletnek, nincs más szerepe. A *LibreOffice Calc* használata esetén az alkalmazandó formátumot – ha az nem szerepel a listán – a *Formátum* > *Stílusok* > *Új stílus* menüponttal hozhatjuk létre úgy, hogy egy cellát előbb megformázunk az előírt módon, majd annak kijelölése után ebben a pontban egy új nevet adunk a stílusnak.

Kicsit bonyolultabb a helyzet, ha az angolos tanulók sorait teljes egészében meg szeretnénk formázni, például egy halvány háttérszínnel. Ekkor előzetesen a teljes *A2:E26* adattáblát ki kell jelölnünk, míg az alkalmazandó képlet egy vegyes cellahivatkozást tartalmaz: *=$C2="angol"*. Mivel a keresett szó a *C* oszlopban van, ezért a *C* oszlopazonosítót rögzítenünk kell, a sorazonosítót azonban nem szabad, hiszen a képletnek a tartomány 2. sora alatti sorokba másolva is működnie kell.

A formátumot a táblázatkezelők a kijelölt tartományra állítják be, de a feltétel vonatkozhat azon kívüli adatokra is. Például kiemelhetjük dőlt betűvel azok nevét, akik 10 óránál többet hiányoztak, ha a kijelölt tartomány az *A2:A26*, míg a feltétel: *=E2>10*.



► A feltételes formázás képletének megadása (balra Microsoft Excel, jobbra LibreOffice Calc)

Feladatok

1. A 11. példában szereplő táblázat *J* oszlopában állítsuk elő a felhasználók monogramját! Egészítsük ki a monogramot egy véletlenszerű nagybetűvel! (A karakterek kódja 65 és 90 közé esik, egy adott kódú karaktert a táblázatba a *KARAKTER* függvénnyel állíthatunk elő.)
2. Melyik tanuló hiányzott a legtöbbet a 9. c osztályban az *osztstat* fájl adatai alapján? Írassuk a tanuló nevét a *G16*-os, mulasztott óráinak számát a *H16*-os cellába!
3. Az angolosok vagy a németesek évfolyamdolgozata sikerült-e jobban? Határozzuk meg az *osztstat* fájl adatainak felhasználásával mindkét átlagot, majd egész mondatos választ írassunk ki a képernyőre! (Például: *A németesek értek el jobb eredményt.*)
4. Hogyan határozhatnánk meg a fiúk, illetve a lányok átlagos hiányzását csupán az *ÁTLAG* és a *HA* függvények felhasználásával?
5. Határozzuk meg képlet segítségével a németes lányok számát, együttes hiányzását és átlageredményét az *osztstat* fájl adatainak felhasználásával!
6. Emeljük ki piros színű betűkkel az *osztstat* táblázatban a lányok és kék színű betűkkel a fiúk sorait!
7. Neverstate-ben a rendőrség rendszeresen ellenőrzi az autók sebességét. Egyik alkalommal az ábrán látható adatokat mérték.
   a) Tudjuk, hogy lakott területen 50 km/h, országúton 80 km/h, autópályán 110 km/h a megengedett legnagyobb sebesség. Írassuk képlettel a *D* oszlopba a megengedett legnagyobb sebesség értékeket a *B* oszlopban lévő adatok alapján!
   b) A büntetés mértéke a megengedett sebességet meghaladó minden megkezdett 30 km/h többlet esetén 100$, például aki lakott területen 120 km/h-val megy, az 300$-t fizet. Írassuk ki képlet segítségével ezeket az összegeket az *E* oszlopba!

   (Mindkét feladatban használhatunk segédtáblát is.)

	A	B	C
1	**rendszám**	**terület**	**mért**
2	AA 333 B	lakott	45 km/h
3	DE 131 AD	országút	180 km/h
4	JA 144 SO	országút	50 km/h
5	IT 995 KT	pálya	250 km/h
6	TA 201 TT	országút	110 km/h
7	LE 333 LE	országút	88 km/h
8	KI 001 CH	lakott	72 km/h
9	BR 444 ER	országút	83 km/h
10	FC 010 SJ	országút	52 km/h

Fájlok kezelése, megosztása

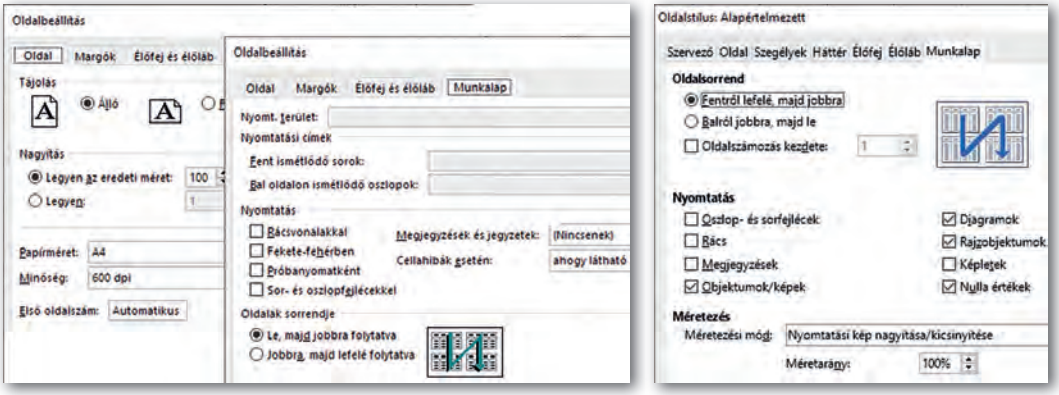
Adatok importálása

A táblázatkezelő programok esetén az adatbevitel nem feltétlenül azok beírásával történik, hanem sok esetben az adatok már más formátumban rendelkezésre állnak.

Egy gyakori formátum a formázatlan szöveg *(txt)*, amikor a cellák tartalmát egy adott soron belül tabulátorjelek választják el egymástól. Egy másik megoldás a *csv* (Comma-separated values, azaz vesszővel tagolt értékek) formátum, ahol egy sorban a cellák adatait vessző (a magyar változatban pontosvessző) választja el. Ezek a formátumok képleteket nem tartalmaznak. A legtöbb táblázatkezelő közvetlenül képes az ilyen fájlokat importálni. Ilyenkor ügyeljünk arra, hogy mentéskor a táblázatkezelő formátumában mentsünk, különben a képletek és a formázási beállítások elvesznek!

Egy szövegszerkesztő programban készült táblázat adatait legegyszerűbben vágólapon keresztül tudjuk átemelni, de ezt a módszert használhatjuk a tabulátorokkal tagolt szöveg átvitelére egy egyszerű editorból, például a *Jegyzettömbből* is.

Az adatok gyakran az internetről származnak. A vágólap ebben az esetben is működik, azonban az adatokat formázás nélkül érdemes beszúrni, például a *Kezdőlap* > *Beillesztés* > *Irányított beillesztés* > *Szöveg* (illetve a *Szerkesztés* > *Irányított beillesztés* > *Formázatlan szöveg*) lehetőséggel. Számok esetén problémát okozhat a tizedespont, illetve az ezres tagolásra használt nem törhető szóköz karakter. Ezeket a *Kezdőlap* > *Keresés és kijelölés* > *Csere* (illetve *Szerkesztés* > *Keresés és csere*) menüponttal cserélhetjük le, a tizedespontot tizedesvesszőre, a nem törhető szóközt pedig üres szövegre.



▶ Ugyanazokat a beállításokat érjük el, más módon (balra Microsoft Excel, jobbra: LibreOffice Calc)

Nyomtatás

A táblázat kinyomtatása általában nem egyszerű feladat, mivel ritkán fordul elő, hogy az adatok elrendezése igazodik a lap méretéhez. Másrészt táblázatkezelés közben nem szoktunk a papíralapú megjelenítés egyéb eszközeire: tájolás, margók, élőfej és élőláb tartalma, rácsvonalak, oszlop- és sorfejlécek megjelenítése stb. figyelni. Ezek beállítására a táblázatkezelő programok különböző lehetőségeket kínálnak, amelyeket például a *Fájl* > *Nyomtatás* > *Oldalbeállítás* (illetve a *Formátum* > *Oldal*) ponttal érhetünk el. Gyakran lehetőségünk van a táblázatkezelés logikáját követő normál elrendezés helyett közvetlenül a nyomtatási képet használni, például *Microsoft Excelben* a *Nézet* > *Lapelrendezés* menüpontra kattintva.

A legfontosabb beállítás a táblázat átméretezése úgy, hogy az kinyomtatva is áttekinthető legyen. Ehhez egyrészt megadhatjuk a táblázat átméretezésének arányát százalékban kifejezve, de gyakran azt is, hogy a táblázat kinyomtatva hány oldal széles és hány oldal magas legyen. Még ilyenkor sem biztos, hogy a laphatárok jó helyre kerülnek, ezért az egér húzásával egyenként módosíthatjuk az oldaltörések helyét, például a *Nézet* > *Oldaltörés előnézet* (illetve a *Nézet* > *Oldaltörés*) menüponttal.

Nagy méretű táblázatok esetén gyakori az is, hogy nem szeretnénk a teljes munkalapot kinyomtatni, hanem annak csupán egy részét. Ezt a nyomtatandó terület kijelölése után a *Lapelrendezés* > *Nyomtatási terület* > *Nyomtatási terület kijelölése* lehetőséggel állíthatjuk be (illetve közvetlenül nyomtatás előtt adhatjuk meg a *Fájl* > *Nyomtatás* > *Tartományok és példányszám* > *Kijelölt cellák* beállítással).

15. példa: Táblázat közös használata

Utolsó példánkban egy osztálybulit szervezünk. Osztályfőnöki órán már megállapodtunk ennek helyéről és időpontjáról, most azt szeretnénk egyeztetni, hogy ki jön el, és aki jön, mit hoz (üdítő, sós süti, édes süti).

Készítsünk táblázatot a mintának megfelelően, mentsük el egy felhőalapú tárhelyre, és osszuk meg a hivatkozást az osztályban! A megosztott táblázat a hivatkozás ismeretében szerkeszthető a böngészőből megnyitva webes alkalmazással vagy a mobiltelefonra telepített mobil alkalmazással is. A megosztott táblázatot egyszerre többen is szerkeszthetik, ilyenkor tiszteletben kell tartanunk a többiek munkáját, nehogy véletlenül töröljük vagy felülírjuk azt.

Sajnos asztali táblázatkezelő esetén a táblázat letöltés nélkül gyakran nem szerkeszthető, ilyenkor a módosított táblázatot vissza kell tölteni.

	A	B	C	D	E
1		Jövök	Mit hozok?		
2			Üdítő	Sós süti	Édes süti
3	Alma Aina	nem			
4	Barack Bardó	x	x		x
5	Citrom Cicerő	x		x	
6	Cseresz Nyeste	x			
7	Dió Dina	x			x
8	Eper Erik	x			x
9	Füge Fürtike	x			
10	Galagonya Gala	nem			
11	Grép Gerda	x		x	
12	Josta Jolán	x			
13	Kivi Kitti	x		x	
14	Körte Kotóny	x	x		
15	Licsi Liem	nem			

Szöveg vagy képlet megadása

1 2 3 4 5 6 7 8 9 0

► Megosztott táblázat szerkesztése (Android, Google Táblázat)

Feladatok

1. Budapest kerületei
   a) Keressük meg az interneten a budapesti kerületek adatait tartalmazó táblázatot, és vágólapon át illesszük be az általunk használt asztali táblázatkezelő programba formázási adatok nélkül!
   b) A táblázatot formázzuk meg, és nyomtassuk ki PDF-fájlba egyoldalasra!

Felhőszolgáltatások

Az informatikának mindig fontos kérdése volt, hogy hogyan, milyen eszközökön tároljuk az adatainkat. A hálózati szolgáltatások kiterjedt használata ebben a kérdésben is jelentős változást hozott. Egyre gyakoribb, hogy adatainkat nem egy adathordozó eszközünkre, hanem a felhőbe mentjük el.

A **felhő (cloud)** egy cég által biztosított online környezetet jelent. Sokféle szolgáltatást nyújthat. Biztosíthat szerverszolgáltatást, virtuális futtatókörnyezetet, online programokat és adattárolási lehetőséget. Az online tárhelyünkre bármilyen adatunkat feltölthetjük, és bármikor, bárhonnan el is érhetjük. Ennek egyetlen feltétele, hogy legyen hálózati hozzáférésünk.

Milyen előnyei lehetnek a felhőben történő adattárolásnak?

Az itt tárolt fájlok **bármilyen, hálózathoz csatlakozó eszközről bármikor hozzáférhető**k. Ez azt jelenti, hogy egy számítógépen létrehozott fájl telefonról, tabletről, másik gépről is elérhető a tárhelyre való bejelentkezéssel. A legtöbb tárhelyszolgáltató lehetővé teszi, hogy egy letölthető egyszerű alkalmazás segítségével kezelni tudjuk a tárhelyen lévő adatokat. Ez sokszor ahhoz hasonló, mintha olyan tárhelyet használnánk, ami az eszközünkhöz tartozik. Az online tárolót úgy látjuk, mint a gépünkbe beépített egyik meghajtót. Sok esetben két helyen is megőrzi a fájlokat a rendszer. Ha nem tudunk azonnal szinkronizálni, az adatok feltöltése később is történhet. A felhőben tárolt adatainkat tehát helytől függetlenül, például otthon, az iskolában, útközben mobileszközről elérhetjük.

Sokszor egyéb kényelmi szolgáltatást is ad a rendszer. Megjegyzi, hogy melyik fájllal dolgoztunk utoljára és hol tartottunk. Egy másik eszközön belépve felajánlja a megnyitását, sőt az olvasást, a szerkesztést pontosan annál a résznél folytathatjuk, ahol az előző hozzáféréskor jártunk.

A felhőben lévő fájlokat nagyon könnyen **megoszthatjuk**. Megosztáskor a megadott felhasználóknak különböző jogosultságot adhatunk a fájlhoz való hozzáférésre. Lehetséges, hogy csak megtekinthetik, vagy ezen felül szerkeszthetik is. A jogokat adhatunk csak egyes személyeknek, vagy bárkinek, aki a linket ismeri.... Érdemes ezt mindig átgondolni. Ez alkalmas lehet nagyobb mennyiségű adat átvitelére vagy a fájlok közös szerkesztésére, csoportmunkára is. Az online tárhelyeken általában lehetőség van a **verziókövetésre** is. Ez

azt jelenti, hogy a mentett fájloknak nemcsak az utolsó verzióját tárolják, hanem az előző néhányat is. Így nem okoz gondot, ha egy előző állapotot szeretnénk visszaállítani.

A felhőben való tárolást sok esetben ingyen igénybe vehetjük. Ha nagyobb tárolókapacitást, egyéb szolgáltatásokat szeretnénk, akkor lehet költsége a rendszer használatának, de ez sokszor még mindig kevesebb, mint a megfelelő tárolóeszköz vásárlásának és üzemeltetésének költsége. Különösen igaz lehet ez az állítás cégek esetén. Nem kell foglalkoznunk azzal sem, hogy a hardveregységeink meghibásodnak vagy elavulnak. Tehát az ilyen szolgáltatás sok esetben költségkímélő.

Felmerül az a kérdés is, hogy lehetnek-e veszélyei a felhőalapú tárolásnak, lényeges probléma, hogy elveszhetnek-e az adataink. Érdemes tudni, hogy a szolgáltatók az adatokat többszörözve (redundánsan) tárolják, ezért kicsi a valószínűsége a véletlen elvesztésüknek. A tapasztalatok szerint nagyon ritkán és rövid időre fordul elő szolgáltatáskiesés.

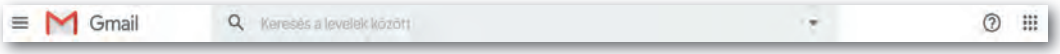
Fontos kérdés az is, hogy ki férhet hozzá az adatainkhoz. A felhőszolgáltatást biztosító cégek általában nagy, több éve biztonságosan működő szolgáltatók. Nem engedhetik meg maguknak, hogy a náluk tárolt adatokhoz illetéktelenek hozzáférjenek. Sok biztonsági intézkedést vezetnek be, ilyen például a **kétfaktoros azonosítás** használata. Ez azt jelenti, hogy a belépéshez nemcsak az azonosítónkat és jelszavunkat kell megadni, hanem egy másik adatot is. Ezt az adatot a rendszer egy előre rögzített saját eszközünkre, például a telefonunkra küldi akkor, amikor észleli a bejelentkezési szándékunkat. Azt is gyakran figyelik, ha szokatlan helyről jelentkezünk be, és erről értesítést is küldenek. Gondoskodnak a felhőben lévő adatok vírus és egyéb kártevők elleni védelméről is. Természetesen a biztonsághoz az is szükséges, hogy mi, felhasználók, megfelelően kezeljük a jelszavainkat és az eszközeink biztonságát.

Egyre több felhőalapú szolgáltatás közül választhatunk. Az alapverzióik legtöbbször ingyenesek. Ilyen szolgáltatások a Dropbox, a Google Drive, az Apple iCloud, a Microsoft OneDrive, a Box.

A felhőszolgáltatások azonban nem csak az online tárolásra szorítkoznak. Sok egyéb szolgáltatás is igénybe vehető. Észrevehetjük, hogy a tárolt fájlokat sok esetben online szoftverrel is meg lehet nyitni, és lehet letöltés nélkül online szerkeszteni.

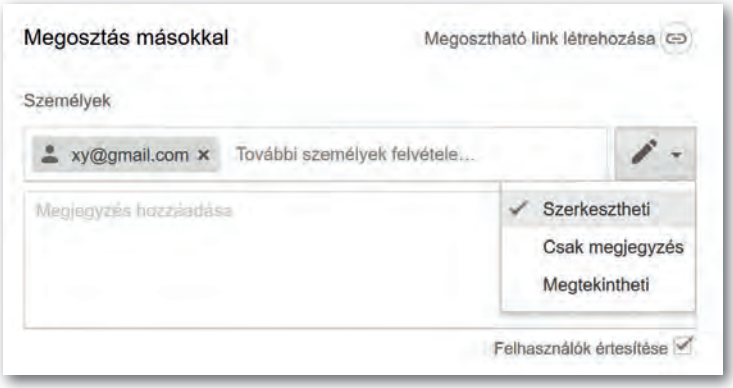
Ismerkedjünk meg a két ismert felhőszolgáltatás alapjaival!

Aki rendelkezik Gmail-postafiókkal, az regisztrált a Google Drive szolgáltatásra is. Ha bejelentkezünk a Gmail-fiókunkba, válasszuk a Google alkalmazásokat, azon belül pedig a Drive-ot:

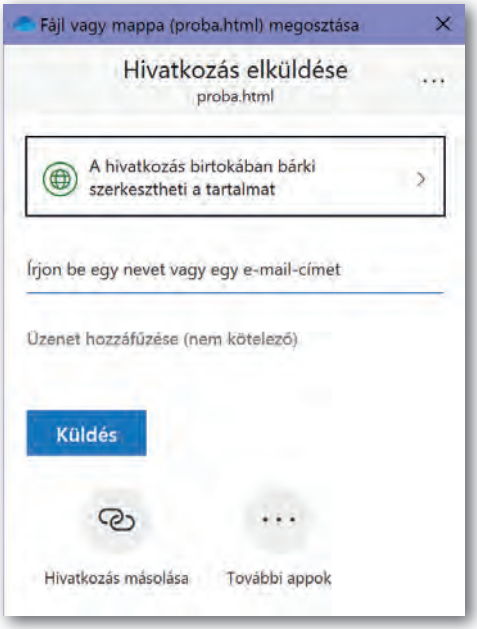


A Drive tárhelyét több célra tudjuk használni. Feltölthetünk és létre is hozhatunk dokumentumokat. Ehhez a Google online szoftvereket biztosít. Ha a bal oldali *Új* gombra kattintunk, választhatunk a programok közül. Lehet szövegszerkesztővel *(Dokumentumok)*, Táblázatkezelővel *(Táblázatok)*, Prezentációkészítővel *(Diák)* és más programokkal, pl. *kérdőív*-készítővel dolgozni. Ezek a programok nem a saját eszközünkön, hanem a felhőben futnak. Hasonlítanak az ismert irodai alkalmazásokhoz, de azoktól jelentősen különböznek is. Ha ismerjük az asztali szoftverek kezelését, akkor ezeket is tudjuk használni. Megfigyelhetjük azonban, hogy kevesebb funkciójuk van, bár folyamatosan fejlesztik, újabb lehető-

ségekkel látják el őket. Nagyon jól használhatók olyan esetben, amikor a telepített alkalmazások nem érhetők el (például otthoni munkavégzés), vagy a feladathoz nem szükséges az összetettebb funkciókkal rendelkező szoftver. A létrehozott dokumentumokat elsősorban a Drive-ra menthetjük.



Lehetőségünk van az itt tárolt dokumentumainkat megosztani. A **megosztás** a dokumentumra vagy könyvtárra jobb egérgombbal kattintva választható ki. **A megosztásnak különböző szintjei vannak**. Jogot adhatunk arra, hogy a kiválasztott személyek csak megtekinthessék, vagy megjegyzésekkel is elláthassák, illetve velünk együtt szerkeszthessék a dokumentumot.



A másik igen elterjedt felhőszolgáltatás, a OneDrive hasonló lehetőségeket biztosít számunkra. A OneDrive asztali alkalmazás a Windows 10 esetében az operációs rendszerbe beépített, korábbi verziók esetében le kell tölteni, de ezt az Office telepítésekor is meg lehet tenni. Alapesetben a OneDrive-regisztráció is ingyenes. Az online tárhely teljes mértékben együttműködik a Microsoft Office alkalmazásaival, különösen az Office 365-tel.

Ezt a felhőszolgáltatást használva szintén kapunk online szoftver használati lehetőséget. A felhőben közvetlenül megnyitva a dokumentumok az Office online alkalmazásban nyílnak meg, és ott szerkeszthetők. Itt is van lehetőségünk megosztásra, sőt, ha van OneDrive szolgáltatásunk, akkor ezt az asztali Office programokban is megtehetjük.

Ha telepítve van a OneDrive a gépen, a megfelelő meghajtón jobb egérgombbal a dokumentumra kattintva kiválaszthatjuk a *Megosztás*t. A továbbiakban beállíthatjuk, kivel és hogyan kívánjuk megosztani a dokumentumot.

Kérdések, feladatok

1. Milyen fájlokat érdemes felhőben tárolni, és milyeneket nem? A választ indokoljuk!
2. Milyen előnye lehet a dokumentumok online szerkesztésének?
3. A továbbiakban egy összetett feladatot fogunk megoldani csoportmunkában. A feladat egy prezentáció közös szerkesztése lesz!
   a. Alakítsunk 2–4 fős csoportokat!
   b. Minden csoport válasszon egy témát!
   c. Gyűjtsünk anyagot a témában!
   d. Hozzunk létre egy prezentációt Google Diákban vagy Office Online-ban, és osszuk meg a csoport tagjaival!
   e. Tervezzük meg, osszuk fel a témát az egyes csoporttagok között! A csoport minden tagja készítsen a prezentációba 2-3 diát!
   f. Az elkészült munkát osszuk meg a tanárunkkal is!
4. Elemezzük a fenti ábrát! Milyen információkat olvashatunk le róla a felhőszolgáltatásokkal kapcsolatban?

► Ilyen szerverközpontokban tárolják a felhőben lévő adatokat.

Online kommunikáció

Online kommunikációs eszközök csoportosítása

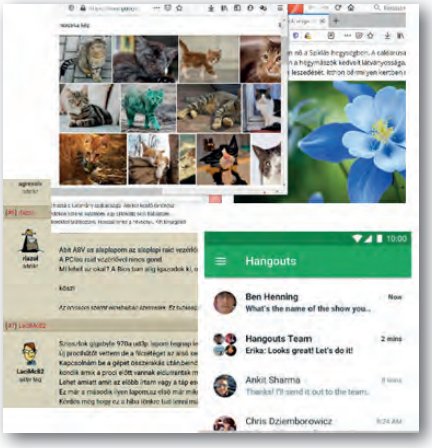
Az elmúlt évszázadban a kommunikáció formája gyökeresen átalakult. Előbb a vezetékes telefon, aztán a rádióadások, majd a televízió megjelenése nagymértékben átalakította az emberek közötti mindennapos kommunikációt, az információ áramlását. Legalább ekkora jelentőséggel bír a mobiltelefonok, az internet, az okostelefonok megjelenése. Mindegyikük újabb lehetőségek, kommunikációs formák megjelenésével járt. Online kommunikáció nélkül ma már elképzelhetetlen a mindennapi életünk.

Az online vagy digitális kommunikáció az információk digitális eszközökön keresztül történő cseréjét jelenti.

Az online kommunikációra használható eszközök és szolgáltatások köre napról napra rohamosan bővül. Egyre fontosabb, hogy képesek legyünk közöttük eligazodni és jól használni őket.

A **kommunikációs szolgáltatások** számos csoportba sorolhatók:

- weboldalak,
- elektronikus levelezés,
- azonnali üzenetküldés, chat,
- fórumok,
- blogok, videóblogok,
- közösségi szolgáltatások,
- tudástárak,
- levelezőlisták, hírlevelek,
- csoportos üzenetküldés,
- IP-telefonálás,
- kép- és videómegosztók.



A kommunikáció időbeli lefolyása szerint lehet szinkron vagy aszinkron.

Az első esetben a kommunikáció lezajlásakor mindkét fél online jelen van a hálózaton. A második esetben nem kell mind a két félnek egyszerre jelen lenni a kommunikációs csatornán, az egyik fél gyakran offline.

Szinkron kommunikációs forma például az azonnali üzenetküldés, aszinkron például az e-mail.

A kommunikáció iránya szerint lehet egyirányú vagy kétirányú, attól függően, hogy mindkét fél küldhet-e üzeneteket.

Az **azonnali üzenetküldő** vagy **chat** alkalmazásokkal gyors szöveges üzeneteket válthatunk hálózaton keresztül. Alapfunkciója szerint a kommunikációban részt vevő felek valós időben (szinkron módon) beszélgetnek egymással, de legyakrabban a rendszer tárolja is az üzeneteket, később is elolvashatjuk őket. Gyakran lehetőség van videóüzenetre és csoportos üzenet küldésére is.

A hagyományos weboldalakkal elletétben, ahol a tartalmat a weboldal üzemeltetője szolgáltatja, ma gyakran használunk **web 2.0 szolgáltatásokat.** Az ilyen weboldalaknál a szolgáltató csak a keretrendszert biztosítja, az oldal tartalmát maguk a felhasználók hozzák létre bejegyzéseikkel. A közösségi szolgáltatások, tudástárak, blogok, fórumok mind ide tartoznak.

A **fórum** olyan internetes közösség, ahol egy meghatározott témában cserélhetünk véleményt, információt a fórum tagjaival. A fórum hozzászólásait szabadon olvashatjuk, de ahhoz, hogy a fórumra hozzászólást írhassunk, általában regisztrálni kell. A hozzászólásokat adminisztrátor ellenőrizheti, megjelenésüket szabályozhatja.

A **blog** egy olyan oldal, ahol a blog tulajdonosa időről időre újabb bejegyzéseket tesz közzé. A bejegyzések lehetnek szövegesek, vagy akár videóbejegyzések is. A blog látogatói hozzászólhatnak a bejegyzésekhez.

A **tudástárak** vagy **wiki**k online lexikonok, amelyek a hagyományos lexikonokhoz hasonlóan szócikkeket tartalmaznak. Legismertebb közülük a Wikipédia.

A **levelezőlista** szintén egy közös fórum, amelyet a listagazda hoz létre, és ő veszi fel a tagjait a listára. A tagok által a listának címzett leveleket minden tag megkapja, és válaszolhat rá.

Sokféle **közösségi szolgáltatás**t ismerünk. Közös bennük, hogy azokkal az oldalra regisztrált tagokkal, akikkel ismeretség köt össze minket, valamilyen tartalmat oszthatunk meg. Ez lehet szöveg, kép és videó. Szolgáltatásuk világszerte egyre népszerűbb, némelyiknek már több milliárd felhasználója van.

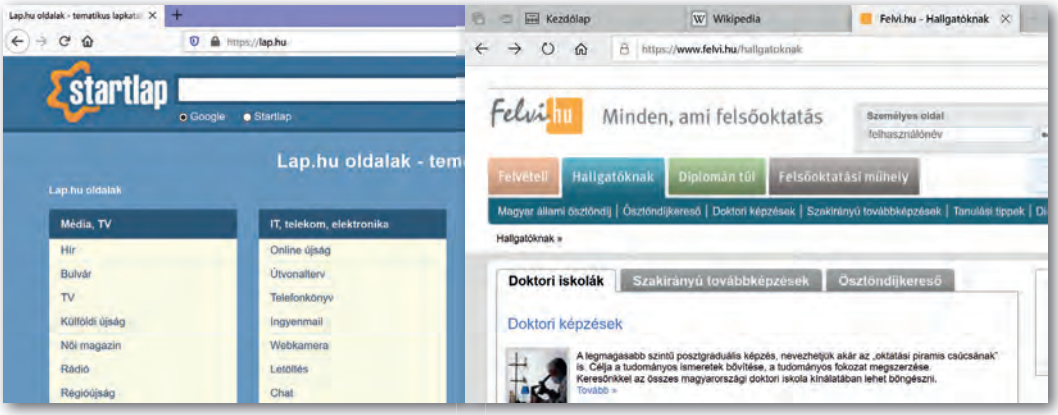
Kérdések, feladatok

1. Alakítsunk 2–4 fős csoportokat! Válasszunk egy megfelelő mobiltelefonos alkalmazást (pl. Viber, WhatsApp), és indítsunk vele csoportos videóbeszélgetést egymás között!
2. Milyen szolgáltatások logói találhatók a képen? Jellemezzük röviden a szolgáltatásokat, melyik milyen célra használható! Keressünk információt az interneten ahhoz, amelyiket nem ismerjük!
3. Keressünk a leckében felsorolt szolgáltatástípusokra további számítógépes vagy mobiltelefonos alkalmazást!

A világháló

A www (world wide web) az internet legnépszerűbb szolgáltatása. Ennek segítségével weboldalakon található információkat tekinthetünk meg, kereshetünk. A weboldalak valójában olyan dokumentumok, amelyeket HTML nyelven (hypertext markup language) készítettek el. Az összetartozó weboldalakat leggyakrabban rendszerezve webhelyekbe rendezik. A weboldalak egyik legfontosabb ismérve, hogy rajtuk hiperlinkek (kereszthivatkozások) találhatók. A hiperlinkek, röviden linkek segítségével más oldalakra hivatkozhatnak címtartalmukkal, rájuk kattintva másik oldalra léphetünk át.

A weboldalak megtekintéséhez böngészőprogramot használunk. A dokumentum webcímét a böngésző címsorába beírva a program lekéri, értelmezi és megjeleníti azt. A címsorban láthatjuk a szükséges és használt protokollt. A weboldalak megtekintéséhez általában a *http* protokollt vesszük igénybe. Ha azonban olyan adatokat cserélünk a weboldallal, amelyeket titkosan kell kezelni, akkor fontos, hogy ezt csak *https* protokollal tehetjük meg. Ilyen eset például, amikor banki vagy személyes adatokat tartalmazó oldalra jelentkezünk be.



▶ Mozzilla Firefox és Microsoft Edge

A böngészők alapszolgáltatásai a konkrét programtól függetlenül hasonlók.

A megtekintett oldalak közt előre- és visszaléphetünk, az oldal betöltését leállíthatjuk vagy frissíthetjük. Megtekinthetjük a böngészés előzményeit, a fontosnak ítélt oldalak címét elmenthetjük a Kedvencek/Könyvjelzők közé.

Böngészés közben nagyon fontos figyelnünk a biztonsági kérdésekre. Tapasztalhatjuk, hogy ha nem először látogatunk el egy weboldalra, akkor bizonyos adatokat már tud rólunk. Lekérdezhetjük a böngészőben, mikor melyik weboldalakat kerestük fel. Ezek az információk hasznosak lehetnek, de nem mindig csak azok. Érdemes a **böngésző adatvédelmi beállításait** időnként átnézni, a már nem szükségesen tárolt adatokat törölni. Szintén itt állíthatjuk be a kiegészítők, a felugró ablakok kezelését. Ha nem szeretnénk, hogy a webhelyek követni tudjanak minket, hogy tárolják az előzményeket, érdemes **inkognitó** ablakot használni. Különösen érdemes erre figyelnünk, ha nyilvános számítógépnél dolgozunk.

Keresés a világhálón, keresési stratégiák

A világhálón több mint másfél milliárd weboldal található, és számuk napról napra rohamosan növekszik. Keresési lehetőségek nélkül lehetetlen lenne áttekinteni a rajtuk található óriási információmennyiséget, megtalálni azt, amire szükségünk van. Ez magyarázza a keresőszolgáltatást nyújtó weboldalak használatát és népszerűségét. A keresőszolgáltatások kétféle típusát különböztethetjük meg.

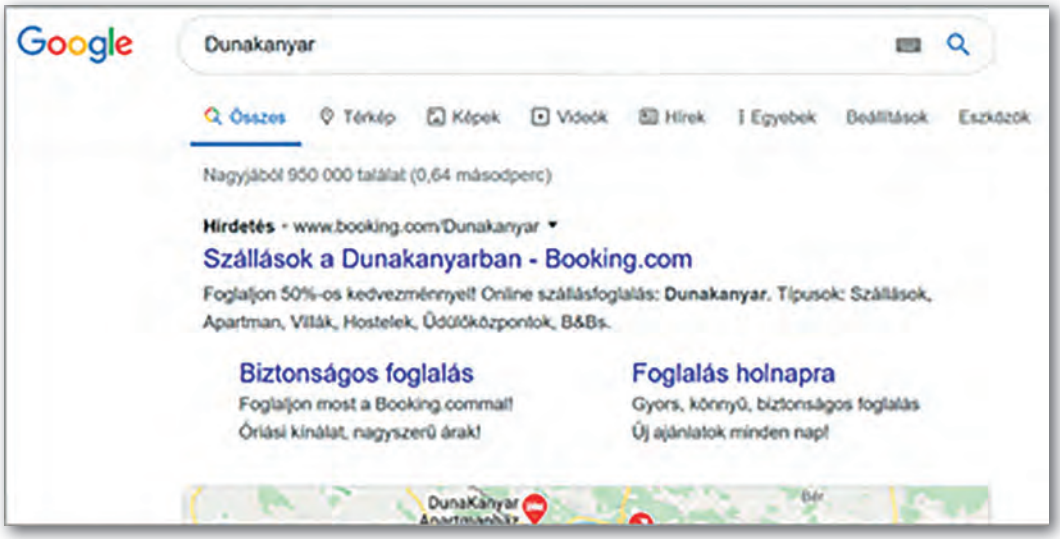
Kulcsszavas vagy index szerinti keresés

Az ilyen elven működő keresőoldalon a keresett tartalom szempontjából jellemző szavakat – kulcsszavakat – írhatunk be. A szerver ezek alapján jeleníti meg a keresésnek leginkább megfelelő **találatokat listába** rendezve.

Az ilyen weboldalakon **keresőmotorok** működnek. Kettős feladatuk van. Folyamatosan pásztázzák a weboldalakat új tartalom után kutatva, és indexelik azokat. Az eredményeket nagy adatközpontokban tárolják. Amikor felhasználóként megadjuk a kulcsszavakat, valójában nem a teljes világhálón, hanem ezekben az adatbázisokban keresik meg a számunkra leginkább megfelelő találatokat.

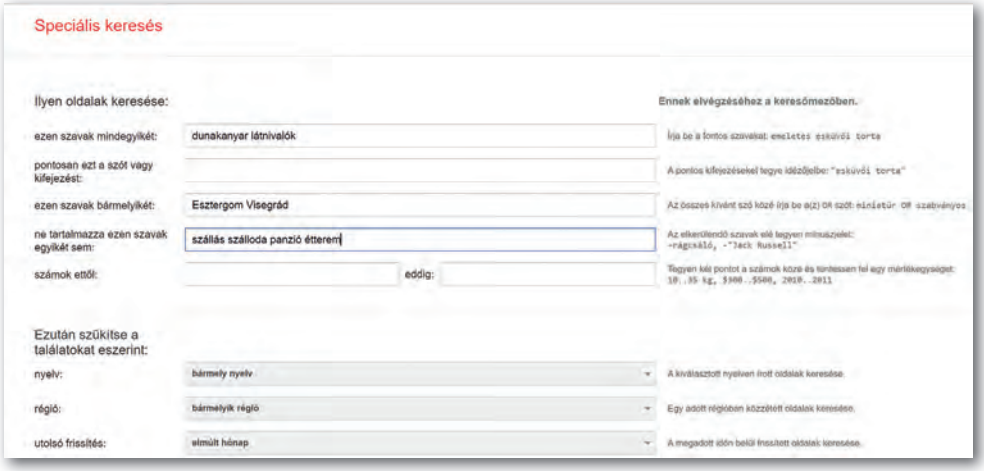
A legismertebb kulcsszavas kereső a Google. Népszerűségét főképp a jól felépített keresőalgoritmusának köszönheti. Szintén ebbe a típusba sorolható például a Bing és a Yahoo is.

1. példa: Keressünk olyan oldalakat a Google kereső segítségével, amelyek a Dunakanyarról szólnak!



A keresőkifejezéstől függően a találatok száma igen nagy, akár milliós nagyságrendű is lehet, de az is előfordulhat, hogy nem kapunk találatot. Ezért fontos, hogy a találati listát képesek legyünk szűkíteni vagy bővíteni. Ahhoz, hogy könnyebben megtaláljuk, amit keresünk, érdemes minél pontosabban megadni a kulcsszavakat. A lista pontosításához további eszközök állnak rendelkezésünkre:

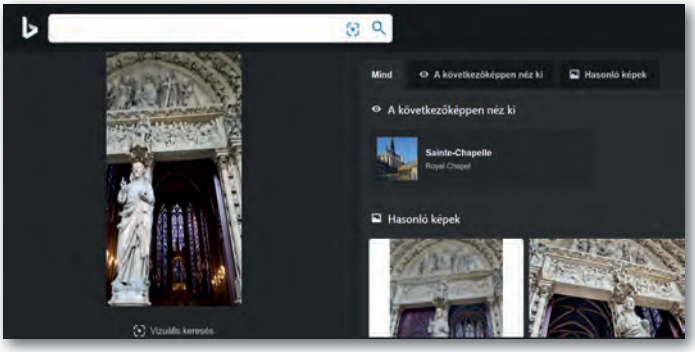
- Ha egy több szóból álló kifejezést keresünk, tegyük idézőjelek közé, így azok az oldalak, ahol a szavak nem egymás után találhatók, nem jelennek meg.
- Ha bizonyos szavakat ki szeretnénk zárni, jelenítsük meg a keresőkifejezésben úgy, hogy egy – jelet teszünk elé.
- Használjunk logikai műveleteket (AND, OR), ha mindkét kifejezésre, vagy legalább az egyikre szeretnénk keresni.
- Megadhatjuk a keresés eredményének fájltípusát, nyelvét, illetve tudunk képeket keresni.



Google keresőben a *Speciális keresés* választásával is megtehetjük ezeket.

Ha sikerül jó keresési paramétereket megadni, a találati listánk első oldalára jó eséllyel a legmegfelelőbb weblapok kerülnek. Érdemes azonban tudnunk, hogy a találati lista első néhány eleme a fizetett hirdetéseket tartalmazza. Bár ezt a szerver feltünteti, nem mindig figyelünk rá.

Előfordul, hogy egy képet látva azt szeretnénk kideríteni, hogy mit vagy kit ábrázol. A keresőprogramok ma már ebben is segíthetnek nekünk. Sok keresőben van lehetőség vizuális keresésre. Ez azt jelenti, hogy egy kép vagy fotó alapján tudunk hozzá hasonlókat keresni. Ez alkalmas lehet arra, hogy megkeressük azt az információt, amelyre keresőkifejezést nem is tudnánk megadni.



▸ Vizuális keresés

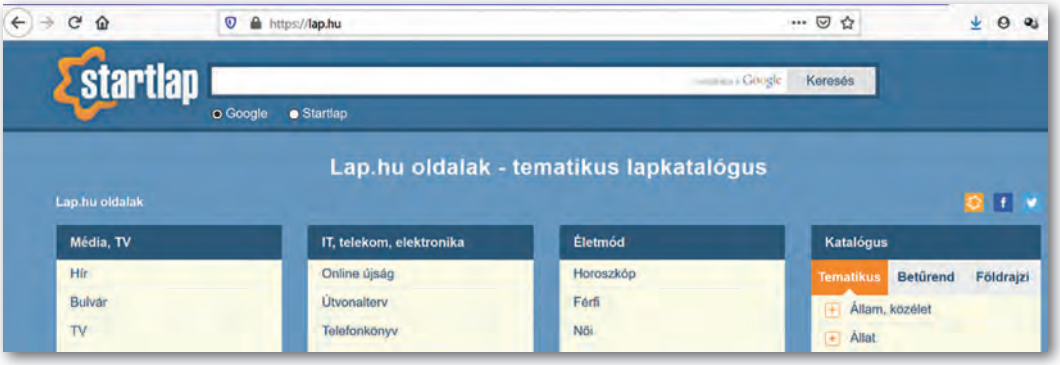
A találati listákat érdemes a találatok hitelessége szerint vizsgálni. Ezzel kapcsolatban támpontot adhat például az oldal utolsó frissítésének dátuma, a források pontos megjelölése, követhetősége, a tartalmi és formai minőség, a megfelelő helyesírás.

Tematikus keresők

2. példa: Keressük meg a legutóbb kihúzott ötöslottó-számokat a lap.hu oldalon! Figyeljük meg a kereső felépítését! Milyen lépésekben juthatunk el a keresett információhoz?

A tematikus keresők a weboldalakat témákba – katalógusokba – gyűjtik, az oldalon ennek megfelelő rendezésben jelennek meg. A témákon belül egyre szűkülően további altémákat találunk, míg végül eljutunk a keresett oldalhoz, információhoz. Bár kevesebbszer használjuk őket, sok esetben gyorsan és pontosan juthatunk el a kívánt tartalomhoz. Fontos tudni, hogy a tematikus keresőknek nem céljuk a teljes világháló feltérképezése, de erre nincs is mindig szükségünk.

A fent leírt keresési stratégiák nemcsak keresőoldalakon, hanem egy-egy weblapon belül is fellelhetők. Például a webáruházak oldalán általában tudunk mindkét eljárással árucikket keresni.



Kérdések, feladatok

1. Keressük meg az interneten, hogy hány magyar Nobel-díjas fizikus van, és kik ők! Tudjuk meg azt is, hogy hol adják át a Nobel-díjat, és töltsünk le egy képet az épületről! Használjunk kulcsszavas keresést!
2. Tematikus kereső segítségével keressük meg, milyen e-könyvet forgalmazó oldalak vannak Magyarországon!
3. Nézzük meg a böngészőprogramunkban az előzményeket! Miért lehet hasznos, és miért lehet problémás, hogy a böngésző ezeket az adatokat tárolja?
4. Keressük fel a Magyar Elektronikus Könyvtár oldalát (http://mek.oszk.hu)! Mutassuk meg, hogy az oldal melyik részén van lehetőségünk tematikus és melyiken kulcsszavas keresésre! Próbáljuk mindkét módszerrel megkeresni a geofizikával foglalkozó könyveket! Milyen különbségeket láthatunk a találati listákban?

Elektronikus levelezés

Az e-mail az internet legrégebbi szolgáltatásai közé tartozik A postai úton küldött levélhez hasonlítható, napjainkra sok szempontból át is vette annak szerepét. Ma már sok hivatalos ügy intézésére használhatunk elektronikus levelet. Szöveges üzenet mellett fájlokat is küldhetünk a címzetteknek.

Ahhoz, hogy elektronikus levelet tudjunk küldeni vagy fogadni, szükségünk van egy **elektronikus postafiók**ra. Ezt és a hozzá kapcsolódó e-mail-címet a szolgáltató nyújtja számunkra, amelynél regisztráltunk vagy előfizettünk. A postafiók fogadja és tárolja a leveleinket. Mérete a szolgáltatótól és az igényelt szolgáltatástól függően eltérő lehet. A leveleket időközönként elolvassuk, ekkor letölthetjük a szerverről, vagy döntésünk szerint meg is őrizhetjük a szerveren.

E-mail-címmel mind a küldő, mind a fogadó félnek rendelkeznie kell. Az e-mail-cím két részből áll, amelyet a @ karakter választ el egymástól. Az első rész a felhasználót, a második a kiszolgálót azonosítja.

Az elektronikus leveleink kezelésére kétfajta szoftvert használhatunk. Dolgozhatunk **webes vagy asztali levelezőrendszer**rel.

A webes rendszereket böngészőprogramon keresztül használjuk. A szolgáltatók egy részénél ingyenesen regisztrálhatunk, így jutunk hozzá az e-mail-címünkhöz és a postafiókunkhoz. Levelezni a böngészőprogram segítségével tudunk bejelentkezés után. Ilyen ingyenes levelezőszolgáltató például a Gmail, Freemail, Indamail, Hotmail, Citromail stb.

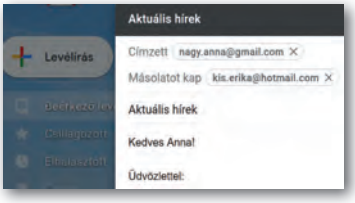
Az asztali levelezőprogramokat a gépünkre kell telepíteni. Első használat előtt, a webes levelezővel ellentétben, be kell állítanunk a postafiókunkat. Leveleinket a program letölti a számítógépre, nem csak a szerveren találhatók meg. Ilyen levelezőprogram például a Microsoft Outlook, Mozilla Thunderbird, IncrediMail. Ezek általában több vagy más szolgáltatást nyújtanak, mint a webes levelezők.

Az elektronikus levéllel végezhető legfontosabb műveletek

Levélírás

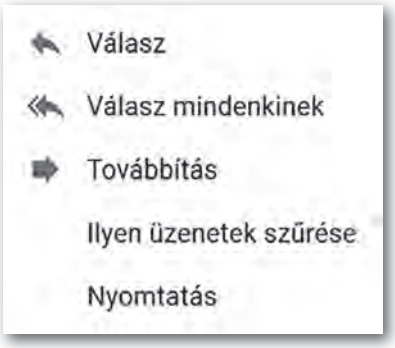
Elektronikus levél írásakor mindig meg kell adnunk a levél **címzett**jét. Lehetőség van arra, hogy **másolati címzett**et (CC: carbon copy) és **titkos másolati címzett**et (BCC: blind carbon copy) is megadjunk. Ez utóbbit a többi címzett nem látja, de a másolati címzetteket igen.

A levelezőprogramok nem tiltják meg, hogy elhagyjuk a levél tárgyát, ennek ellenére ezt mindig illik megadni, ez a címzett felé alapvető udvariasság. A tárgy elhagyása miatt a levelet kéretlennek ítélhetik, ezért nem biztos, hogy célba ér vagy el is olvassák. A levél szövegének kitöltése és az aláírás mellett a levélhez fájlokat is csatolhatunk. A **csatolt fájl** méretét, esetleg típusát a levelezőszerverek korlátozzák, érdemes tudni, hogy a két fél esetében ez a korlát mekkora. Amennyiben a csatolmány túl nagy lenne, a saját online tárhelyünkön meg tudjuk osztani, vagy igénybe tudunk venni óriáslevél-küldő szolgáltatást. A legtöbb levelezőben beállíthatjuk a levél **prioritás**át, ezzel jelezhetjük a címzettnek, hogy az üzenet sürgős, valamint kérhetünk **kézbesítési és olvasási visszaigazolás**t.



Műveletek beérkező levelekkel

A beérkező leveleket a megfelelő mappában láthatjuk. Olvasás után a levélre válaszolhatunk vagy továbbíthatjuk egy másik címzettnek. Fontos, hogy a válasz- vagy továbbküldött levélben azt a tartalmat hagyjuk csak meg, amely a másik fél számára még információt hordozhat. A beérkező leveleket rendezhetjük feladó, beérkezési idő, tárgy szerint. Az üzeneteket megjelölhetjük, címkézhetjük, áthelyezhetjük másik mappába. A leveleket törölhetjük, vagy a fontosakat archiválhatjuk, és ki is nyomtathatjuk őket.



Névjegyek

A levelezőprogramok lehetőséget adnak arra is, hogy az ismerőseink elérhetőségeit névjegyekben tároljuk. Ma ez gyakran összekapcsolható az egyéb helyeken tárolt névjegyekkel.

Az alapszolgáltatásokon felül mindkét fajta levelezés nyújthat számunkra további lehetőségeket. Gyakran kezelhetünk például személyes határidőnaplót, amelybe elfoglaltságainkat bejelölhetjük. Sok webes és asztali levelező segítségével könnyen kapcsolódhatunk más szolgáltatásokhoz, például felhőalapú tárolás, azonnali üzenetküldés.

Levelezés és biztonság

Mint minden hálózaton érkező objektum, az elektronikus levél is hordoz biztonsági kockázatokat.

A hasznos leveleink mellett nagyon gyakran kapunk számunkra kéretlen leveleket ismeretlen címzettektől. Az ilyen levelet **spam**nek nevezzük. A spam általában valamilyen terméket, weboldalt, szolgáltatást reklámoz.

Problémát jelenthet, ha sok haszontalan levél érkezik a postafiókba, amelyek keverednek a valóban nekünk szóló, fontos üzenetekkel. A levelek válogatása időt, figyelmet igényel. A kéretlen levelek emellett lehetnek veszélyforrások is. Gyakran érkeznek velük vírusok, kémprogramok, adathalász üzenetek. A csatolmányokkal kapcsolatban mindig érdemes óvatosnak lenni, de ismeretlen feladótól érkezett üzenet csatolmányát nem ajánlott megnyitni.

A legtöbb szolgáltató és levelezőprogram nyújt olyan szolgáltatásokat, amellyel a rosszindulatú kóddal ellátott levelek és a spamek ellen védekezhetünk. Ezek a spamszűrő szolgáltatások. Megkeresik és külön mappába gyűjtik a levélszemétnek vélt leveleket. Érdemes figyelni arra, hogy előfordul, hogy olyan levelek is a spamek közé kerülnek, amelyeknek nem ott lenne a helyük, de a rendszer annak véli őket.

D A digitális eszközök használata

Mielőtt elkezdenénk…

„Ha egy fészekben van 10 kakukktojás, és egy fürjtojás, akkor melyik a kakukktojás?”

Ez a fejezet egy kakukktojás a könyvben. Az itt leírtak nem önálló tanulási egységei a tananyagnak, hanem ide akkor kell lapozni, ha más témakörhöz kapcsolódóan felmerül egy kérdés az eszközhasználattal kapcsolatban. Ezt a fejezetet először címek szintjén érdemes gyorsan áttekinteni, hogy tudjuk, mik azok a területek, amikről szükség esetén itt olvashatunk. Ha feladataink elvégzése közben felmerül egy kérdés, amiről itt van leírás, akkor az adott részt tanulmányozzuk, és használjuk fel.

Az egyes témák után ebben a fejezetben is található *Feladatok, kérdések* rész. Ha a téma felkeltette az érdeklődésünket, akkor az itt leírtaknak érdemes lehet máshol is utánanézni. Ezek nem kötelező részei a tananyagnak, ebben a könyvben a kérdésekre adandó válaszok nem olvashatók, de sok érdekességre lelhetünk, míg megtaláljuk azokat.

Egy kis történelem

Az informatikai eszközök története különböző szempontok szerint egészen eltérő időszakokat eredményez. Ha úgy nézünk ezekre az eszközökre, hogy mikroprocesszorokkal működnek, akkor történetük ott kezdődött, amikor megalkották az első mikroprocesszorokat. Így csak néhány évtizeddel ezelőttről beszélhetünk az informatikai eszközök történelme kapcsán. De ha azt tekintjük kiindulásnak, hogy az ember már nagyon régen törekedett olyan szerkezetek megalkotására, amelyek a távollétében az általa megadott folyamatokat elvégzik, akkor egészen az ősemberek csapdakészítéséig visszamehetünk az időben. Képzeljük el azokat az ősembereket, akik gödröt ástak, és ágakkal, gallyakkal fedték el, hogy az arra járó vadat elejthessék. Egy tökéletesen időzített automatát alkottak. Pontosan akkor működött, amikor a vad arra járt. Soha nem ejtette el az állatot korábban vagy később. Pont akkor, amikor arra járt. És ezt tette az ember távollétében. Ha innen közelítjük meg az automaták, az informatikai eszközök történetét, akkor százezer években mérhetjük azt.

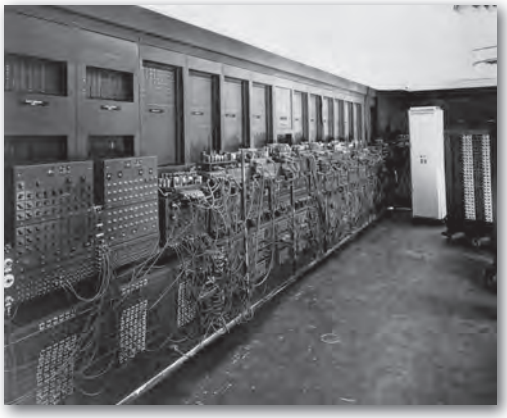
Ha kevésbé szélsőségesen szeretnénk megközelíteni, akkor az informatikai eszközök történetét a számolás automatizálásának a történetével indíthatjuk. Az ókor különböző civilizációiban jelent meg az abakusz, mint a számolást könnyítő eszköz.

► Ősi abakusz

Mondhatjuk, hogy ezek voltak az első digitális számolóeszközök. Így néhány ezer évre nyúlik vissza az informatika története.

Megközelíthetjük onnan is, hogy a középkor végének és az újkor elejének számos neves tudósa mechanikus számológépet készített. Az 1600-as években Wilhelm Schickard, Blaise Pascal és Gottfried Wilhelm Leibniz is készített, tökéletesített ilyen eszközt. Így már csak néhány száz évre szűkítettük az informatika történetét.

A műszaki fejlődés a háborúk idején mindig lökésszerűen megugrik. Ez történt a II. világháborúban is. A különböző harci eszközök fejlesztése és használata nagy mennyiségű számítás elvégzését igényelte. Például egy új löveg elkészítése után szükség volt arra, hogy a lövegkezelő rendelkezzen egy olyan táblázattal, amiből kiolvashatja, hogy a löveg milyen állásában mekkora távolságra lehet vele lőni. Amikor rendre készültek az újabb és újabb lövegek, megnövekedett az igény a számítási kapacitások iránt, hogy a szükséges táblázatokat minél hamarabb és pontosabban elkészíthessék. Erre és hasonló problémákra építették az Egyesült Államokban az első tisztán elektronikus, általános célú digitális számítógépet, az ENIAC-ot az 1940-es években. Ezzel értünk el az informatika történetének szűkítésében oda, ahonnan már mondhatni egyenes ágon leszármaztathatóak a mai informatikai eszközeink, a laptopunk, a telefonunk, a fitnesz karperecünk, az intelligens mosógépünk, az okostévénk. Ezek az eszközök működésük, felépítésük alapján nagyon sok mindenben megegyeznek az ENIAC működésével, felépítésével. Persze tudjuk, hogy az ENIAC teremnyi méretű berendezés volt, ezzel szemben egy okosóra három–négy nagyságrenddel gyorsabban végzi a számítási műveleteket, de mégis az alapelvek szintjén rengeteg az azonosság.



► ENIAC az első általános célú elektronikus digitális számítógép

Feladatok, kérdések

1. Mi motiválhatta Blaise Pascalt a mechanikus számológépe megépítésében? Érdemes több független forrást is keresni, hogy lehetőség szerint hiteles információkhoz jussunk.
2. Gyűjtsünk adatokat az ENIAC számítógépről! Mekkora volt az alapterülete? Mekkora volt a tömege? Mekkora volt az áramfogyasztása? Hány összeadási műveletet tudott másodpercenként elvégezni? Mai értékre átszámítva mennyibe került az előállítása?
3. Hasonlítsuk össze egy mai számítógép processzorában található tranzisztorok számát az ENIAC gépben található elektroncsövek számával!

A modern digitális eszközök működése

Nézzük azokat az alapelveket, amikben egységesnek tekinthetjük a mai informatikai eszközöket. Ehhez vissza kell nyúlnunk, az 1940-es évek közepéig, amikor az Amerikai Egyesült Államokban egy tudóscsoport azon dolgozott, hogy létrehozzon egy teljesen elektronikus, programozható számítógépet, az ENIAC-ot. A készülék korszakalkotó volt, de a tudósok, akik dolgoztak rajta, menet közben már látták, hogy mit és hogyan kellene módosítani, hogy egy még hatékonyabb, és univerzális gépet hozhassanak létre. Az ENIAC tapasztalatai alapján a Pennsylvaniai Egyetemen a kutatócsoport megállapításait a magyar származású Neumann János – aki a következő elektronikus számítógép (EDVAC) megépítését végző projekt tanácsadója volt – vetette papírra. Ezért mi magyarok szeretjük ezeket az elveket Neumann-elveknek nevezni, de fontos tudnunk, hogy ez egy csapat matematikus és mérnök közös munkája volt.

A teljesség igénye nélkül vegyük sorra ezeket az elveket:

► Elektroncső

- **A számítógép legyen teljesen elektronikus működésű!** Az ENIAC és az azt követő években a többi számítógép is elsősorban elektroncsövekből épült fel. Az így elkészített berendezések számolási sebessége akár az ezerszerese is volt a korábbi mechanikus alkatrészeket is tartalmazó gépeknek. Így a sebesség szempontjából is egyértelmű volt, hogy ez a jövő. Ha megnézzük a mai digitális eszközök számítási sebességét, akkor azt látjuk, hogy ezt a sebességet sokmilliószorosan túlléptük azóta.
- **Az eszköz használja a kettes számrendszert a műveletvégzéseiben!** E mögött egy nagyon praktikus mérnöki gondolat húzódik meg. A kettes számrendszerben csak kétféle számjegy létezik (0 és 1), ami elektronikai eszközökkel könnyebben kezelhető állapotot jelent, mintha 10 értéket kellene minden helyiértéken megkülönböztetni, ahogy az a tízes számrendszerben szükséges. Mérnökileg egy alacsony és egy magas feszültségszint kezelése nagyobb toleranciát, hibatűrést, könnyebben megépíthető áramköröket jelent. A kettes és tízes számrendszer közötti átváltás egész számok esetében problémamentes, törtek esetében a véges darabszámú helyiértékek miatt adódnak átváltási, kerekítési problémák, amelyekre programozáskor figyelemmel kell lennünk. A kettes számrendszer használata az elektronikai összetevőkkel egyszerűbb áramköröket eredményez, mintha ugyanezt tízes számrendszerrel kellene megtenni, így ez az elv mind a mai napig meghatározó az informatikai eszközeinkben.
- **A gépnek legyen belső memóriája, ami az adatok és a programutasítások tárolását egyaránt elvégzi!** Ennek egyik következménye az, hogy a programutasításokat adat-

ként kezelve, azok módosíthatók, tehát a program képes akár önmagát is megváltoztatni. Az első számítógépek esetén néhány tíz, néhány száz adat tárolását oldották meg. Napjainkban több milliárd adatot tárolhatunk egy számítógép memóriájában.

- **A gép legyen univerzális, azaz ne egy speciális feladatra készüljön, hanem a programok segítségével különböző feladatokat legyen képes ellátni!** A mai informatikai eszközökben ezt a tulajdonságot teljesen természetesnek tekintjük például akkor, amikor a telefonunkra letöltünk egy új programot, ami olyan feladatokat lát el, amire korábban a telefonunk nem volt képes.
- **A számítógép legyen soros végrehajtású, azaz a program utasításai egymás után, időben sorban történjenek!** A soros végrehajtás teljesen praktikus okokból került az alapelvek közé. Ennek megvalósítása egyszerűbb volt, mint a párhuzamos programvezérlés, ez csökkentette az egyébként sem egyszerű berendezés bonyolultságát. Ez az elv az, ami a mai eszközöknél már sokszor nem teljesül. Egy mai korszerű számítógép többmagos processzorral és egy komoly videokártyával rendelkezik, így ezek az elvégzendő műveleteket egyszerre, párhuzamosan hajtják végre, és nem kapcsolják ki magukat addig, amíg a másik eszköz végzi a számításokat.

Feladatok, kérdések

1. Neumann János (John von Neumann) milyen iskolákat végzett, mely tudományterületeken ért el jelentős eredményeket?
2. Keressünk adatokat arról, hogy egy mai mobiltelefon és egy tíz évvel ezelőtti laptop számítási teljesítménye hogyan viszonyul egymáshoz!

A digitális eszközök főbb egységei

Sokféle digitális eszközzel vesszük magunkat körbe. Ezek jelentősen különbözőnek tűnnek, de ha megnézzük a belső felépítésüket, a funkcionális összetevőiket, akkor láthatjuk, hogy jobban hasonlítanak egymásra, mint elsőre gondolnánk.

Milyen részekből is áll egy asztali számítógép, egy laptop, egy okostévé, egy mobiltelefon, egy tablet, egy intelligens távirányító, egy e-book olvasó, egy okosóra, egy fitnesz karkötő?

► Laptop

Perifériák

Az elsők, amikkel a használat közben találkozunk, a **be- és kimeneti perifériák**. Ezek azok a részek, amiken keresztül mi információt tudunk adni az eszköznek, és amelyeken keresztül az eszköz tud válaszolni.

Szinte minden digitális eszköznek van egy **kijelző**je, **képernyő**je. Ez általában egy színes, grafikus megjelenítő. Itt egyik értékként a *képátló hosszát* szokták megadni, jellemzően inch-ben (inch: hüvelyk, 1 hüvelyk = 2,54 cm). Ez mobiltelefonnál lehet például 6” (~15,2 cm), egy asztali monitornál 24” (~61 cm), egy tévénél 55” (~140 cm). Másik jellemző érték a *felbontás*, azaz, hogy vízszintesen, függőlegesen hány képpontot tud megjeleníteni egymás mellett a kijelző. Egy monitor esetében lehet például 1 920 × 1 080 (FullHD), ami azt jelenti, hogy vízszintesen 1920 képpontot, függőlegesen 1080 képpontot tud megjeleníteni. Ez összesen 1 920 · 1 080 = 2 073 600 képpont a kijelzőn. Egy 4K-s tévé esetén ehhez képest mind vízszintesen, mind függőlegesen dupla annyi képpont van egy sorban, illetve oszlopban. Tehát 3 840 × 2 160 a felbontás, ami több mint 8 millió képpontot jelent a képernyőn. A legtöbb kijelző színes, a pixelgrafika fejezetben tárgyalt *RGB* színkódolást használ. Van néhány eszköz, aminek a kijelzője *fekete-fehér*, vagy csak a szürke néhány árnyalatát tudja megjeleníteni.

► Hagyományos könyvek és e-book olvasó

Ennek oka lehet, hogy sokkal olcsóbb egy egyszínű kijelző, és sok esetben elegendő is. Másik ok a technológia lehet. Az e-book olvasók az e-papír technológiája miatt csak *szürkeárnyalat*os megjelenítésre képesek.

Több olyan informatikai eszköz van, ami kijelző helyett, vagy mellett képes **hang** segítségével is információt adni nekünk. Itt a legegyszerűbb a *sípoló*, *csipogó*, *bippegő* hang, ami például a be- vagy kikapcsolást, egy érték elérését jelezheti, vagy egysze-

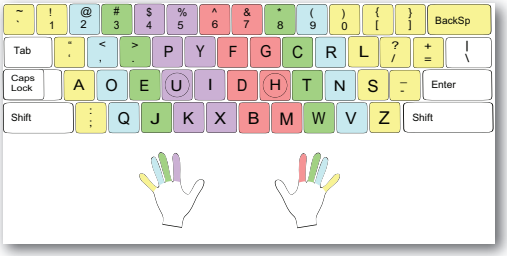
► Okostévé

rű figyelmeztetést adhat. Ennél összetettebb hangokkal is rendszeresen találkozunk, amikor zenét játszunk le, vagy például a digitális asszisztensünk tájékoztat az aktuális időjárásról, vagy futás közben az okostelefonunk egyik programja elmondja, hogy milyen sebességgel tettük meg az utolsó kilométert. A hangok esetén van, amikor elegendő, hogy hallunk egy információt, de zene esetén már szeretjük a *sztereó* hangzást, azaz azt, hogy a jobb és a bal oldalon akár eltérő lehet a hang. Ekkor már nem elegendő egy *hangszóró*, legalább kettőre van szükség. A számítógépes játékoknál, filmeknél a hangzás térbelisége, a mély hangok kiemelése (például egy akciójelenetben) még fontosabbá válhat. Ekkor már két hangszóró nem is elegendő. Ilyenkor a hangszórók körülvehetnek minket, és teljessé tehetik a *térbeli hangzás*t. Egy profi rendszerben akár 7–8 *hangcsatorna* is elkülönülhet.

Az utóbbi időben egyre több eszközben jelenik meg a *rezgőmotor*, amivel a zsebünkben lévő telefon, a csuklónkon lévő okosóra vagy fitnesz karkötő ad diszkrét jelzéseket.

Láthattuk, hogy az érzékszerveink közül a látás, a hallás, a bőrérzékelés használata már mindennapinak számít az informatikában a **kimeneti perifériák**nál, azaz azoknál az összetevőknél, amiknek a feladata az, hogy információt közöljenek velünk.

A kimeneti perifériák után vegyük sorra, hogy milyen módon tudunk mi információt adni egy digitális eszköznek. Egy számítógép esetén elsőként a *billentyűzet* juthat az eszünkbe. Ennek segítségével szövegeket tudunk beírni, utasításokat tudunk adni, vezérelni tudjuk a programok működését, irányítani tudjuk kedvenc karakterünket egy játékban. A billentyűzet használatakor fontos, hogy milyen nyelvhez készült a billentyűzet, mert ha magyar billentyűzethez szoktunk, akkor egy angol vagy francia billentyűzeten a speciális írásjelek megtalálása jelentős időbe kerülhet. Ha egy angol billentyűzet elé ültetnek minket, akkor a magyar ékezetes karakterekkel leszünk bajban. Találkozhattunk már azzal a helyzettel, hogy egy billentyűzeten egyes gombokat lenyomva nem az jelent meg a kijelzőn, mint ami a gombon volt. Ennek oka, hogy a számítógépes program határozza meg azt, melyik billentyű milyen karaktert is jelentsen. A *billentyűzet nyelve* az operációs rendszer beállításai között módosítható. Ezért amíg nem tudunk vakon gépelni, akkor dolgozhatunk jól egy billentyűzettel, ha a rajta lévő feliratok és az operációs rendszer billentyűzetre vonatkozó beállításai megegyeznek.



► Dvorak billentyűkiosztás

Ahogy a szövegszerkesztés fejezetben is olvashattuk, a billentyűzetkiosztás még a mechanikus írógépeken kialakított elrendezést követi. Az írógépeken úgy tették egymás mellé a billentyűket, hogy a szövegben gyakran egymás mellé kerülő betűk távol legyenek, mert az írást végző mechanikus karok így akadhattak legkevésbé össze. Egy számítógép

billentyűzete, vagy egy okostelefon képernyőjén megjelenő virtuális billentyűzet esetén ez az elrendezés így már nem indokolt. Sok kutatás foglalkozott azzal, hogy milyen elrendezés mellett lehetne sokkal gyorsabban gépelni. Az ezek alapján létrehozott billentyűzetek kísérleti jelleggel megjelentek, de elterjedni nem tudtak. Ennek elsődleges oka a megszokás, az attól való eltérés nehézsége.

Egy számítógép esetén a grafikus felhasználói felület kezelésének másik fontos eszköze az **egér**. Ha pontosabban és általánosabban akarunk fogalmazni, akkor a **mutatóvezérlő**, mivel ez lehet akár egér, érintőpárna, trackball (hanyattegér), pöcökegér.

Melyiket hol használjuk, mik az előnyei?

Egér: a legelterjedtebb mutatóvezérlő eszköz. Már biztosan találkoztunk több fajtával is, amiken eltérő számú gomb volt. A régebbieknek az alján golyó volt, az újabbakon optikai érzékelő figyeli a mozgást.

Érintőpárna (touchpad): a laptopokon láthatjuk a billentyűzet alatti területen. Ha a laptopot hordozható gépként használjuk, akkor ez a mutatóvezérlő mindig ott van velünk, nem igényel külön helyet a gép mellett.

► Érintőpárna

Pöcökegér: ez egy kisméretű botkormány a billentyűzetbe elhelyezve. Aki sokat gépel, és néha van csak szüksége a grafikus kurzor mozgatására, az a keze felemelése nélkül irányíthat ezzel a kis eszközzel. Kell egy kis idő, hogy megszokjuk, de utána gépelés közben mindig kézre esik. Programozók közül sokan kedvelik.

► Pöcökegér

Trackball: szokták hanyattegérnek is nevezni, mert a régi egerek alján lévő, mozgás érzékeléséhez szükséges golyó itt fentre került. Ezt a golyót kell az ujjunkkal mozgatni, aminek hatására mozog a grafikus kurzor. Előnye az egérrel szemben, hogy ezt nem az asztalon kell mozgatni, és akkora helyre van csak szüksége, amekkora maga az eszköz. Sokan idegenkednek tőle, de a grafikusok között sokan használják.

► Trackball – hanyattegér

Ha már grafikusoknál tartunk, érdemes megemlíteni a *digitalizáló táblák*at is, amelyek egy kis táblán érintésérzékeny felülettel rendelkeznek, és egy speciális digitális íróeszköz tartozik hozzájuk. Ezen a felületen rajzolva a kép közvetlenül a számítógépben keletkezik, ahol a digitális toll a szoftver beállításaitól függően működhet például

▶ Táblagép (tablet)

ceruzaként, tollként, ecsetként, radírként. A nagyon precíz nyomásérzékelőnek köszönhetően a vonalrajzolás intenzitása gyengébb vagy erősebb ceruzarajzot eredményezhet, vagy az ecsetvonásokat tudja jól visszaadni.

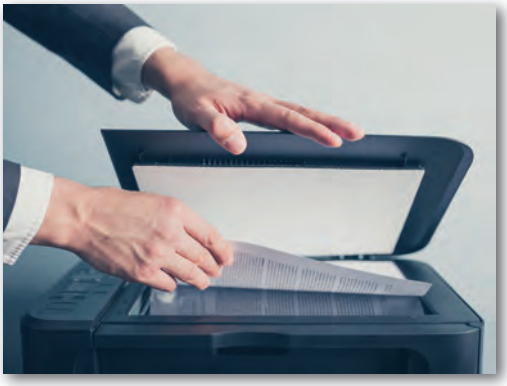
Vannak olyan digitális eszközeink, amelyek grafikus felhasználói felülettel rendelkeznek, de se billentyűzetet, se egeret nem szoktunk hozzájuk kapcsolni. Ilyen például az okostelefon és a tablet (táblagép). Ezeknél a szöveg bevitelét és a grafikus kurzor pozícionálását a kijelzőbe épített érintésérzékeny felületen végezhetjük el. A gépeléshez ilyenkor egy virtuális billentyűzetet használunk.

Az eddig felsorolt bemeneti perifériákat mind a kezünkkel vezéreltük. Mozgással még más különböző információ beviteli lehetőségeink vannak. Gondoljunk arra, hogy amikor a mobiltelefont elforgatjuk 90 fokkal, akkor azt a képernyőn megjelenő program is képes követni. Például a böngészőben 90 fokkal elfordul a szöveg, vagy a lejátszott film is fordul. A telefonban, tabletben különféle mozgásérzékelők vannak, melyek képesek érzékelni az elmozdulás irányát, sőt akár a sebességét, gyorsulását is. Ugyanilyen érzékelőket használnak a fitnesz karkötőkben a mozgás érzékelésére, a lépés számlálására. Ezek a mozgási adatok adják meg feldolgozás után, hogy egy kirándulással, egy futással mennyi energiát égettünk el, mennyit nassolhatunk ezért cserébe anélkül, hogy tartanunk kellene az elhízástól.

A digitális eszközeink egyre nagyobb részét tudjuk már hanggal is vezérelni. Ehhez az adott nyelven „értő” szövegfelismerőre van szükség. Az egyszerűbb utasításoktól, mint például a mobiltelefonnál „Hívd Anyut!”, az összetett feladatokig, mint a szövegdiktálás, vagy a digitális asszisztensnek adott keresési feladatok, lehetőségünk van már akár magyarul is, hanggal vezérelni berendezéseinket. Nem meglepő módon ehhez egy mikrofonra van szükség, és természetesen arra a programra, ami képes a hangrezgésekből létrehozott elektromos impulzusokat, digitális jeleket szövegként, utasításként kezelni, feldolgozni.

A digitális eszközeink elsődlegesen a kijelzőn keresztül kommunikálnak velünk. Viszont nekünk is van lehetőségünk képeket bejuttatni az eszközbe. Ehhez egy kamerára van szükségünk. A *kamera* lehet a mobiltelefonba vagy laptopba beépített, de lehet egy számítógéphez külön csatlakoztatott *webkamera*, vagy akár egy *digitális fényképezőgép*, amit a számítógépünkhöz kapcsolunk. A kamera képénél fontos szempont, hogy milyen minőségű képet tud előállítani. Ezt sok összetevő mellett a kamera felbontása is erősen meghatározza. Egy laptopba integrált 1280×720 képpont felbontású kamera képe kevesebb képpontot biztosít, mint amennyit egy FullHD felbontású monitor képes megjeleníteni. Egy mobiltelefonba integrált 32 megapixeles (32 millió képpontos) kamera képe négyszer annyi képpontot tartalmaz, mint amennyit egyszerre meg tud jeleníteni egy 4K felbontású tévé. Miért van ekkora felbontásra szükség? Ha képernyőn szeretnénk a teljes képet meg-

jeleníteni, akkor nincs szükség ekkorára. Ha viszont szeretnénk egy részletet kinagyítani, kiemelni, akkor már fontos, hogy vannak olyan képpontok, amikkel ez megtehető. Ha nyomtatni szeretnénk, akkor nagyobb felbontású képre lesz szükségünk.



► Lapolvasó

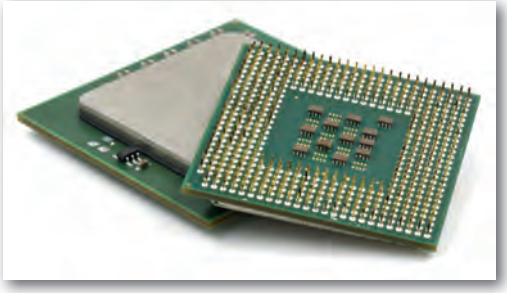
Ha dokumentumok, papírlapok tartalmát kell digitalizálnunk, akkor nem mindig tökéletes megoldás a digitális fényképezőgép vagy a mobiltelefon használata. Erre a feladatra sokszor megfelelőbb egy *lapolvasó* (scanner) használata. A lapolvasó felbontását nem úgy adják meg, hogy összesen hány képpontot képes rögzíteni, hanem úgy, hogy egy adott távolságon belül hány képpontot tud megkülönböztetni. Jellemzően a felbontást DPI (dot/inch: pont hüvelykenként) értékben adják meg, ami megmutatja, hogy 2,54 cm-en hány pontot tud megkülönböztetni a lapolvasó. Jellemző érték a 600 DPI, de egy otthonra is megfizethető árú berendezés akár 4800 DPI értéket is tudhat.

A képek digitalizálásának több módja és eszköze van. Gondoljunk csak a vonalkódolvasókra vagy a digitális röntgengépekre. Itt mindig a beolvasott képi információ feldolgozása lesz a fontos feladat, mint például egy bolti pénztárban a vonalkódolvasóval a vonalak által jelzett számsor értelmezése, a leolvasott termék adatbázisban tárolt árának hozzáadása a számlához, esetleg a raktárkészlet azonnali módosítása, szükség esetén az új termékek automatizált beszerzésének indítása.

Központi feldolgozóegység

Az eddigiekben azokról az eszközökről volt szó, amikkel a használat közben elsőként találkozunk, azaz a ki- és bemeneti perifériákról. Ezek csak a kapcsolatot teremtik meg a számítógép és a külvilág között. Nézzük, hogy mi van a „motorháztető” alatt!

A számítógép utasításokat hajt végre. Ezeknek az utasításoknak olyanoknak kell lenniük, amiket a gép belső alkatrészei megértenek, végre tudnak hajtani. Mi is hajtja végre ezeket az utasításokat? A digitális eszközünkön belül található egy, az utasítások végrehajtásáért felelős rész, ami egyrészt képes elvégezni a számítási műveleteket, másrészt képes a memóriában adatok formájában tárolt utasítássornak megfelelően vezérelni a számítógép működését. Ez a központi feldolgozóegység, angolul *central processing unit (CPU)*. Ez egy mikroprocesszor, amiben tranzisztorok segítségével létrehozott áramkörök felelősek a vezérlésért, a számítási műveletek elvégzéséért. A központi feldolgozó egységet röviden processzornak is szoktuk nevezni. A *processzor* napjainkban akár több, egymás mellett párhuzamosan működő egységből is állhat. Ebben az esetben ezeket a részeket processzor-magoknak nevezzük. Egy otthoni használatra szánt processzornak lehet 4–8

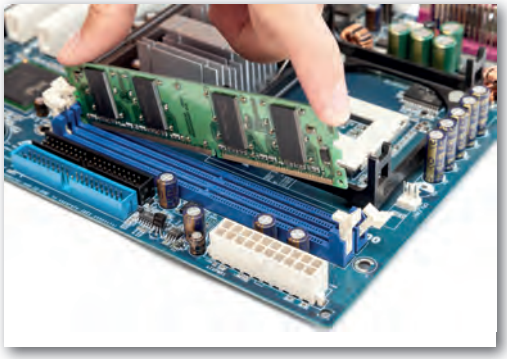


► CPU

magja, de speciális grafikai feladatok esetén ez akár 16 mag is lehet egy asztali számítógépben. Mivel ezek a magok külön-külön képesek egyidőben műveleteket végezni, ezért a Neumann-elvek között említett soros, azaz szigorúan egymás utáni feladatvégrehajtás már nem teljesül, illetve meghaladott.

Memória, RAM

A számítógép memóriája az, amiről mindenki hallott már, mindenki sejti, mi az, de sok esetben egy digitális eszköz leírásában zavaróan keverednek fogalmak, kifejezések, ami nem segíti a megértést. Szedjük sorra, hogy mi is az a memória, milyen lényeges fajtái vannak.



▶ Memória (RAM)

A számítógépnek szüksége van egy olyan adattároló részre, ami tárolja az éppen futó program utasításait, adatait, vagy azoknak legalább egy éppen feldolgozás alatt álló részét. Ennek egy olyan tárolónak kell lennie, amiből az adatok tetszőleges sorrendben, gyorsan elérhetők. Ez a napjaink digitális eszközeiben a **RAM** (Random Access Memory: véletlen elérésű memória). Ebben több milliárd bájtnyi adatot tudunk eltárolni. Egy asztali számítógépnél ez lehet 8, 16, vagy akár 32 GB is. Mobiltelefonoknál is találkozunk 10–12 GB-os értékekkel. Minél nagyobb ez az érték, annál több programot tudunk egyidőben futtatni, vagy annál nagyobb memóriaigénnyel rendelkező program képes akadásmentesen dolgozni. Ilyen lehet például egy videószerkesztési feladat. Az adatok a RAM-ban addig maradnak meg, amíg a RAM folyamatos áramellátást kap. Ha egy táblagép vagy mobiltelefon kijelzőjét kikapcsoljuk, akkor az az eszköz még működik, az akkumulátor kapacitásától függően akár napokig képes az adatokat megtartani. Ha lemerül az akkumulátor, akkor a nem mentett adatok elvesznek.

Háttértárak

Hova menthetjük az adatokat, ahol megmarad akkor is, ha már nincs áramellátás? A még mindig széles körben elterjedten használt **háttértár**-típus a merevlemezes meghajtó (HDD: Hard Disk Drive), ami egy zárt dobozban lévő, mágnesezhető lemezekre rögzíti az adatokat. Ez nem számít ma már elég gyorsnak, és a technológiából adódóan nem is nagyon lehet arra számítani, hogy érdemlegeset fejlődjön ezen a területen. Előnye az, hogy viszonylag alacsony áron biztosítja nagy mennyiségű adat tárolását. Jellemzően otthoni számítógépbe 2–4 TB kapacitású elegendő, de létezik 10–16 TB-os tárterületű is, ami például nagy mennyiségű videó tárolására alkalmas.

▶ Merevlemezes meghajtó (HDD)

Az egységnyi adat tárolásának költsége a merevlemezhez képest drágább az **SSD**-k (Solid-state drive: szilárdtest-meghajtó) esetén. Ezek a félvezető alapú tárolók nem tartalmaznak mozgó alkatrészeket, az adatok írása és olvasása akár több mint hússzorosa is lehet a HDD-k sebességének. Az ilyen típusú tárolók hasonlóan működnek, mint a korábban bemutatott memóriák, ezért több helyen ezeket is egyszerűen RAM-ként jelzik. Felmerülhet a kérdés, hogy amennyiben a memóriához hasonlóan működnek, de nem veszítik el tartalmukat az áramellátás kimaradása esetén, akkor miért nem ezeket használjuk a digitális eszközeink belső memóriájának. A válasz viszonylag egyszerű: több nagyságrenddel lassabbak, így program futtatására a mai sebességelvárások mellett nem lennének alkalmasak. A mobileszközökben – mint táblagép és mobiltelefon – ilyen típusú háttértárakat alkalmaznak. Ott ezek mérete elérheti a GB-os kategóriát, bár az eszköz árára ennek egészen jelentős hatása van. A mobil eszközök leírásánál sokszor a belső memóriát és a háttértárat is RAM-ként jelölik, ami nem segíti az eligazodást. Ha látunk két számértéket a RAM jelzés után, akkor szinte biztos, hogy ezek közül a kisebb érték a belső memória, míg a nagyobb számérték a háttértár.

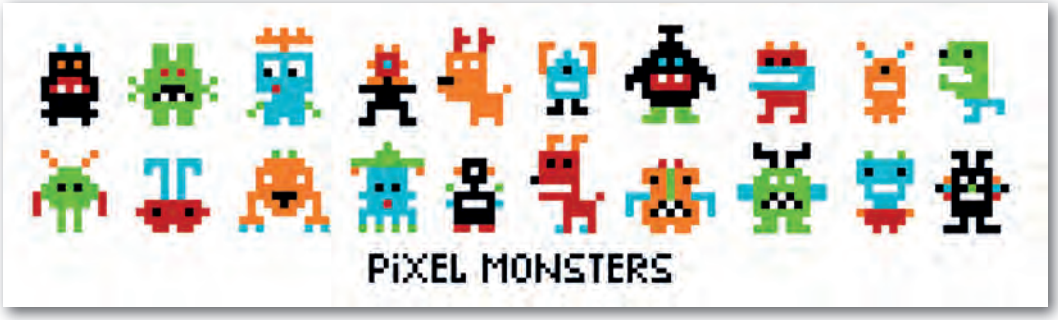
► Félvezető alapú tároló (SSD)

Egyéb kiegészítők

A fentebb leírt összetevőkből összeállítható nagyon sokféle digitális eszköz, de van pár olyan funkció, amihez külön speciális áramköröket, mikroprocesszorokat terveznek. Ennek oka, hogy az adott szolgáltatás ne terhelje a CPU-t, vagy adott területen speciális szolgáltatást tudjanak biztosítani.

Ma már minden eszköznél elvárt, hogy az internetre legyen kapcsolva. Az okosotthon koncepció szerint a lakásban lévő légkondicionáló, fűtésvezérlésért felelős termosztát, robotporszívó, hűtőgép és egyéb berendezések legyenek elérhetők interneten keresztül, azokat a mobiltelefonunkról vezérelhessük, legyünk bárhol a világon. Ehhez arra van szükség, hogy mindegyik eszközünk képes legyen hálózati kapcsolatot létesíteni vezetékes vagy vezeték nélküli módon. Ezt egy speciális, erre kifejlesztett áramköri elemmel oldják meg, amit sok esetben *hálózati kártyá*nak hívnak, holott nem feltétlenül van kártya alakja. Az elnevezés használata az 1980-as évek elejére nyúlik vissza, amikor az IMB PC (Personal Computer: személyi számítógép) létrehozásakor az volt az alapelv, hogy ezek a számítógépek az alapfelépítésük mellett legyenek bővíthetőek speciális feladatokat ellátó komponensekkel. Ezeket hívták bővítőkártyáknak. Amennyiben a hálózathoz vezeték nélküli kapcsolattal képes csatlakozni egy eszköz, szükséges egy antenna is, amit sokszor az eszköz dobozán belül helyeznek el, mivel a mérete ezt lehetővé teszi. Egy mobiltelefon képes többféle vezeték nélküli kapcsolatot létrehozni. Gondoljunk arra, hogy az utcán történő telefonáláshoz a telefontársaság átjátszóállomásához kell kapcsolódni, otthon wifikapcsolaton keresztül érjük el az internetet, a vezetéknélküli hangszóróhoz, fitnesz karkötőhöz pedig Bluetooth segítségével tudunk kapcsolódni.

Amikor egy filmet nézünk, egy számítógépes játékkal játszunk, a képi hatás mellett fontos számunkra, hogy a hangzás is teljes legyen, ne csak a kijelző irányából jöjjön, hanem



► 8 bites játék szereplői – alacsony grafikus teljesítmény mellett is lehetnek hőseink, antihőseink. :-)

vegyen minket körbe. Ehhez speciális többcsatornás hangrendszerre van szükségünk, aminek vezérlését egy hangkártya fogja elvégezni. Ez határozza meg, hogy hány hangszórót tudunk csatlakoztatni, milyen speciális hangeffektusokkal tudjuk kiegészíteni a hangzást.

A képi megjelenítésnél elvárjuk a nagy felbontást, az élethű színeket, az akadásmentes mozgást, a 3D élményt. A nagy számítási teljesítményre felkészített videokártya fogja biztosítani, hogy az operációs rendszer ablakkezelése látványos legyen, és egy videójátéknál a szereplők játékhelyzethez igazodva, de lehetőleg élethűen jelenjenek meg. Egy komoly, játékosoknak szánt videokártya ára egy számítógépben lehet akkora összeg, amennyiért egy szerényebb, de a legtöbb feladatra bőségesen elegendő grafikus képességgel megáldott komplett számítógépet lehet kapni.

Egy hordozható eszközben – laptop, táblagép, mobiltelefon – az utólagos bővítés lehetősége nagyon korlátozott. Egy laptopban cserélhető esetleg a memória vagy a háttértár nagyobbra, de például a processzor, vagy a beépített vezetékes hálózati csatlakozás nem cserélhető nagyobb teljesítményűre. Egy táblagépben vagy mobiltelefonban ennyi fejlesztési lehetőség sincs. A legtöbbször a háttértár bővíthető egy-egy SD-memóriakártyával.

Feladatok, kérdések

1. Mit jelent, ha egy kerek kijelzős okosórához azt írják, hogy a felbontása 480 × 480?
2. Hogyan működik az e-papír, miben különbözik egy OLED kijelzőtől?
3. Egy e-book olvasóba hány könyv fér el?
4. Mit jelent, hogy egy hangrendszer 2.0, 2.1, 5.1, vagy 7.1 jelzést kap?
5. Mi a giroszkóp feladata?
6. Nézzünk utána, hogy egy laptop CPU-ja mekkora alapterületű, és hány tranzisztort tartalmaz!
7. Mit jelentenek egy okostelefonnál a következő jelölések: RAM 8 GB / 512 GB, 6,2”, 64 MP / 12 MP, 2,8 GHz Octa-core CPU, IP68?

Operációs rendszerek

Az elektronikus számítógépek első generációját azok a tudósok használták elsősorban, akik a gép megépítésében részt vettek, vagy legalábbis értették a gép működését, felépítését. Az általuk készített programok teljesen az adott számítógép fizikai összetevőire, hardverére épültek. Később egyre többen kezdtek használni számítógépeket, és egyre nagyobb igény merült fel egy olyan felhasználói felületre, amin keresztül az ember könnyebben tud a számítógéppel kommunikálni. Ez kezdetben kizárólag karakteres felületen történt. A személyi számítógépeknél az 1980-as években megjelentek a grafikus felhasználói felületek, amelyek már széles kör számára tették könnyebben elérhetővé a számítógépek szolgáltatásait.

```
raerek@laptop:/$ ls -l

drwxr-xr-x  1 root root    4096 Oct 28  2018 bin
drwxr-xr-x  1 root root    4096 Jul 25  2018 boot
drwxr-xr-x  1 root root    4096 Apr 26 10:39 dev
drwxr-xr-x  1 root root    4096 Apr 26 10:39 etc
drwxr-xr-x  1 root root    4096 Oct 29  2018 home
-rwxr-xr-x  1 root root  591344 Jan  1  1970 init
drwxr-xr-x  1 root root    4096 Jul 25  2018 lib
drwxr-xr-x  1 root root    4096 Jul 25  2018 lib64
drwxr-xr-x  1 root root    4096 Jul 25  2018 media
drwxr-xr-x  1 root root    4096 Apr 20 11:40 mnt
drwxr-xr-x  1 root root    4096 Jul 25  2018 opt
dr-xr-xr-x  9 root root       0 Apr 26 10:39 proc
drwx------  1 root root    4096 Jul 25  2018 root
drwxr-xr-x  1 root root    4096 Apr 26 10:39 run
drwxr-xr-x  1 root root    4096 Feb 24 16:07 sbin
drwxr-xr-x  1 root root    4096 Jul 19  2018 snap
drwxr-xr-x  1 root root    4096 Jul 25  2018 srv
dr-xr-xr-x 12 root root       0 Apr 26 10:39 sys
drwxrwxrwt  1 root root    4096 Apr 20 11:41 tmp
drwxr-xr-x  1 root root    4096 Jul 25  2018 usr
```

► Linux ls parancs és eredménye – karakteres felhasználói felület

A **karakteres felület**en jellemzően utasításokat, parancsokat adhatunk ki, amiknek eredményeként szöveges válaszokat kapunk a képernyőn. Ehhez ismernünk kell a kiadható utasításokat, azok használatának módját, paramétereit, és néha az eredmény értelmezése is csak megfelelő előismeretek birtokában lehetséges. Ezzel szemben a mai **grafikus felhasználói felület**ek kialakításánál törekednek arra, hogy könnyen érthető, kevés előismerettel is használható legyen, a még esetleg nem ismert funkciók is intuitív módon felfedezhetők legyenek. A felhasználók széles körének készített operációs rendszerek szinte kivétel nélkül grafikus felhasználói felülettel rendelkeznek. Számítógépen ilyen például a *Windows*, a *macOS*, a *Chrome OS* és a különböző *Linux*ok. A mobiltelefonokon, tableteken például az *Android*, az *iOS*, valamint az *iPadOS*.

A különböző grafikus felhasználói felületek használata attól lesz a felhasználóknak könnyű, hogy nagyon sok elemükben azonos módon működnek, így az egyik megismerése segíti a másik használatát. Nézzünk pár példát! A felhasználói felületen az alkalmazások *ablak*ban futnak, azaz a kijelző jól meghatározott területét használják. Persze ez adott esetben lehet akár az egész képernyő is. Az ablakokban általában *menü*ket találhatunk, a menüpontokhoz almenüpontok tartozhatnak. Ha valamit be kell írnunk, akkor *beviteli mező*t kell kitöltenünk, döntéseinket *gomb*okkal jelezhetjük. Amennyiben egy rendszerben egyidőben több programot futtathatunk, akkor lehetőségünk van közöttük váltani, vagy sok esetben akár több program képét is láthatjuk egymás mellett. Az említett és más hasonló tulajdonságok miatt mondhatjuk, hogy könnyű ezeknek a rendszereknek a használatát elsajátítani. Ha valaki egy teljesen új rendszert szeretne készíteni, akkor figyelembe kell vennie, hogy a felhasználóknak ezek a már megszokott felületek, jelentősen eltérni ettől nagy bátorság és kockázat.

A grafikus felhasználói felület kezelésének egyik szinte elengedhetetlen eszköze az *egér* vagy más *mutatóvezérlő eszköz*. Az operációs rendszerek viszonylag egységesek az egér-

használatban. Vegyük sorra, mik az alapvető egérműveletek azon túl, hogy az egér segítségével a kis nyilat lehet mozgatni a képernyőn.

Bal gombbal *egy szimpla kattintás*: ez a kiválasztás, a kijelölés, ami lehet például egy gomb, egy felirat, egy állomány. Az érintőpárnán vagy érintőképernyőn ezt egy koppintással érjük el.

Bal gombbal *dupla kattintás*: ez a kiválasztott állomány megnyitása, a kiválasztott program elindítása, mappáknál a mappába belépés. Az érintőpárnán vagy érintőképernyőn dupla koppintással érjük el.

Jobb gombbal kattintás: ez jellemzően a helyi menüt hozza elő, ami az adott környezetben leginkább releváns műveletekhez ad gyors hozzáférést. Ezt érintőpárnán a legtöbb eszköznél kétujjas koppintással, míg érintőképernyőn ujjunkat hosszabban ott tartva érhetjük el.

Vonszolás, azaz amikor az egér bal gombját lenyomjuk, és miközben nyomva tartjuk, mozgatjuk az egeret: ez egy objektum, ablak áthelyezését teszi lehetővé. Az érintőpárnán vagy érintőképernyőn ez dupla koppintással érhető el úgy, hogy a második koppintás után nem emeljük el az ujjunkat, hanem mozgatjuk a felületen. A vonszolást a gomb vagy érintőfelület elengedésével fejezhetjük be.

Egyes programokban a többszörös kattintásnak is van funkciója. Próbáljuk ki például a szövegszerkesztőben, mi történik, ha egy szövegen belül egy szóra egyszer, kétszer, háromszor vagy négyszer kattintunk!

Természetesen a balkezes egérhasználathoz a jobb és a bal gomb felcserélhető az operációs rendszer beállításaiban.

Az egér mellett a billentyűzetnek is jelentős szerepe van az operációs rendszer kezelésében. Azért, hogy könnyen megtanulható legyen a különböző programok használata, a billentyűkombinációk sokszor ugyanazt, vagy nagyon hasonló feladatokat látnak el. Nézzünk néhány gyakran használt gyorsbillentyűt!

Ctrl + C: másolás. Ez sok esetben a Ctrl + Insert segítségével is elvégezhető.

Ctrl + V: beillesztés. Sokszor lehet helyette Shift + Insert.

Ctrl + X: kivágás. Esetleg Shift + Delete.

Ctrl + A: mindent kijelöl (A: all, magyarul összes).

F1: a súgót nyitja meg.

F2: átnevezés. A fájlkezelőben a fájl vagy mappa átnevezésére szolgál, a táblázatkezelőben pedig a kijelölt cella szerkesztésére.

Alt + F4: ablak bezárása

Ctrl + F4: egy ablak egy fülének bezárása, például böngészőben.

F5: frissítés. Amikor egy ablakban megjelenített tartalomról tudjuk, hogy az már megváltozott, de még a gép nem jelenítette meg, akkor ezzel frissíthetjük a tartalmat. Például a fájlkezelőben egy hálózati mappa megtekintésekor lehet jó, vagy a böngészőben, ha szeretnénk a weboldalt ismét betölteni, frissíteni.

F10: a program menüjének elérése.

Ctrl + S: mentés (S: save, magyarul mentés).

CTRL + P: nyomtatás (P: print, magyarul nyomtatás).

Azzal, hogy a különböző programok hasonlóan viselkednek, ezáltal „kézre esnek”, a szolgáltatásiak megvalósítják a könnyű, kényelmes használhatóságot, azaz a **szoftver er-**

gonómiát. Ehhez még hozzátartozik a felület logikus elrendezése, átláthatósága, a program által megvalósítandó funkciók logikus elérése.

Az operációs rendszer feladata a felhasználóval való kapcsolattartás, és a programok futtatásához szükséges környezet biztosítása, a programok futtatása. A leggyakrabban használt operációs rendszerek lehetővé teszik a programok párhuzamos futtatását, a váltogatást közöttük. Ezt ma már teljesen természetesnek tekintjük.

Egy számítógépnél megszokott, hogy a gép indulásakor ki kell választani, hogy melyik felhasználó fogja használni, azaz minden felhasználónak elkülönített *felhasználói fiók*ot hozhatunk létre. Ezzel elérhető, hogy legyen olyan tárterület a gépen, amit csak az egyik vagy csak a másik felhasználó érhet el. Ez az adatvédelem szempontjából nagyon fontos. Hasonlóan a tárterület biztosításához, felhasználónként azt is szabályozhatjuk, hogy kinek van lehetősége például programokat telepíteni, kinek csak futtatni. Akár az is megadható, hogy melyik felhasználó melyik programot indíthatja el.

A mobiltelefonokat jellemzően mindig csak egy személy használja, így ott a több felhasználó felvétele a rendszerbe nem alapvető elvárás. Vannak olyan mobiltelefonok, ahol van lehetőség több felhasználói fiókot is létrehozni, és vannak olyanok is, amelyeken egy felhasználónak az üzleti és magánjellegű tevékenységeihez kapcsolódó adatokat – mint például a fényképeket, e-maileket – lehet jelszóval védetten elkülöníteni.

Hasznos szolgáltatások

Az operációs rendszerek azon kívül, hogy lehetőséget biztosítanak a programok futtatására, rengeteg segédprogramot tartalmaznak, amelyek nem elengedhetetlen részei az operációs rendszernek, de a digitális eszközünk használatát megkönnyítik. Ilyen program például egy **számológép**, egy egyszerű **képszerkesztő**, egy **stopperóra**, ami nem feltétlenül szükséges, de hasznos.

Nézzünk pár példát arra, hogy mik az operációs rendszer hasznos, a mindennapi munka szempontjából nélkülözhetetlen segédprogramjai.

Védekezés a digitális kártevők ellen

Sokat hallunk a számítógépes vírusokról, férgekről, kémprogramokról, agresszív reklámprogramokról, arról, hogy ezek mennyi kellemetlenséget, problémát okozhatnak. Ezeket a programokat együttesen **rosszindulatú szoftverek**nek is szoktuk nevezni. A rosszindulatú szoftverek célja lehet a fájlok, adatok törlése, módosítása, a fájlok titkosítása annak érdekében, hogy a dekódolásért zsarolhassanak, jelszavak, bankkártya-adatok megszerzése, a megfertőzött gép használata illegális tevékenységekre, mint például a spam küldés.

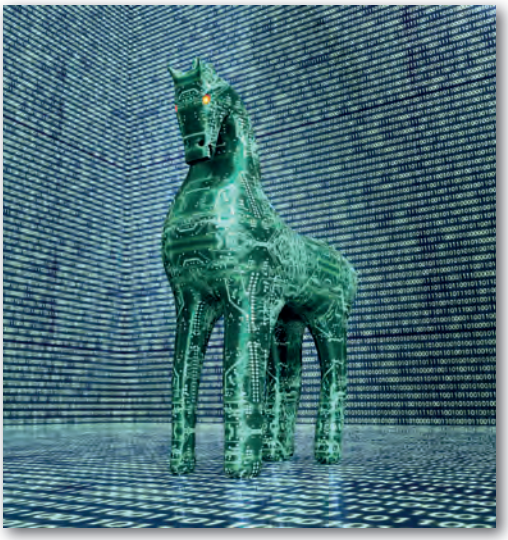
Vegyük sorra a digitális kártevők csoportjait, és azt, hogy hogyan védekezhetünk ellenük. A leggyakrabban talán a **számítógépes vírusok** elnevezéssel találkozunk. Ezek olyan programok, amelyek másik programhoz, vagy a rendszerbetöltésért felelős tárterületre írják magukat, ezzel elérve, hogy a digitális eszközünk, illetve a programok normális működésekor elinduljanak, és a rendszerben

► Vírustalálat

szinte tetszőleges műveletet végezhessenek. Ebből az egyik lépés az, hogy igyekeznek önmagukat újabb fájlokhoz másolni, ezzel biztosítani a szaporodást. A vírus elnevezés azért találó, mert ezek a programok is csak parazitaként, más programhoz kapcsolódva képesek szaporodni, mint a biológiai vírus egy élőlény sejtjében. Számítógépes vírus leggyakrabban nem megbízható forrásból származó programokkal jut a digitális eszközre. Ellenük a *tudatos, odafigyelő számítógéphasználói magatartás* mellett a *víruskereső és -irtó program*mal tudunk védekezni. Ez sok esetben az operációs rendszer mellé adott segédprogramként áll a rendelkezésünkre. Sokakban él az a tévhit, hogy vírusos csak a Windows lehet, ezzel szemben viszont fontos tény, hogy minden digitális eszközünk ki van téve ennek a kockázatnak, legyen az Windows, Linux, macOS rendszert futtató számítógép, vagy akár a táblagépünk, mobiltelefonunk.

A **féreg** (angolul: worm) a vírussal szemben nem más programokhoz kapcsolódva képes terjedni, hanem ezt teljesen önállóan teszi. Általában a számítógépes hálózaton terjed úgy, hogy kihasználja a rendszerekben fellelhető programhibákat, biztonsági réseket. Védekezni vírusirtó programokkal lehet ellene, valamint azzal, hogy az operációs rendszerünket és a gépünkre telepített programokat rendszeresen frissítjük, mert a frissítések sok esetben a fejlesztők által megismert biztonsági réseket foltozzák be, ezzel is megakadályozva a rosszindulatú programok terjedését. A férgek hálózati terjedésének megakadályozásában még nagy segítséget jelentenek a *tűzfal program*ok, amik a számítógépes hálózaton keresztüli forgalmat szűrik. A tűzfal programok is szinte kötelező segédprogramjai az operációs rendszereknek.

A **trójai program** a nevét a görög mitológiából ismert trójai falóról kapta. A trójai egy olyan program, ami másnak mutatja magát, mint ami valójában. Gyakori, hogy keresünk egy probléma megoldására egy programot az interneten, és nem megbízható forrásból, ellenőrizetlen programot telepítünk a gépünkre. A trójai program elvégzi azt a feladatot, amire beszereztük, de mellette olyan tevékenységeket is végez, amik kárt okozhatnak. Ilyen lehet például a gépen található állományok titkosítása, és ezáltal elérhetetlenné tétele. Miután egy ilyen program a tárolt adatok jelentős részét titkosította, megjelenít egy üzenetet, hogy hova milyen módon kell a „váltságdíjat” befizetni azért, hogy a feloldókódot megkapjuk. A lekövethetetlenség érdekében napjainkban kriptovalutában (például Bitcoin-ban) várják az összeget a szoftver készítői, de az, hogy fizetünk, még semmi garanciát nem jelent arra, hogy az adatainkhoz valóban hozzá fogunk utána férni. A trójai programok sok esetben nem ilyen látványosan fejtik ki hatásukat, hanem csak egy „hátsóajtót” nyitnak a gépen, azaz lehetőséget biztosítanak a szoftver készítőinek, hogy a gépünkre újabb programot juttassanak, a gépünk feletti irányítást magukhoz



► Trójai faló

vehessék. Sok esetben ilyenkor a gépünk látszólag végzi a dolgát, csak kicsi lelassulást érzünk, esetleg az internetelérés sebességével leszünk elégedetlenek. Eközben a gépünk a háttérben ezerszámra küldi más felhasználók postaládáiba a kéretlen reklámleveleket, más néven spam-eket. Védekezni tudatos géphasználattal lehet, azaz ellenőrizetlen forrásból nem telepítünk semmit, valamint a vírusirtó programok nyújtanak még segítséget.

A **kémprogram**ok (angolul: spyware) olyan, főleg interneten terjedő szoftverek, melyeknek feladata felhasználói adatok megszerzése, mint például személyazonosító adatok, bankkártya-adatok, jelszavak, amelyek felhasználásával általában bűncselekményeket hajtanak végre. Ilyenek lehetnek mások nevében kötött szerződések, kötelezettségvállalások, bankszámlák megcsapolása, illetve a jelszavak felhasználásával akár nemkívánatos üzenetek küldése, a felhasználó ismerősei elérhetőségének megszerzése kéretlen reklámok továbbítása céljából.

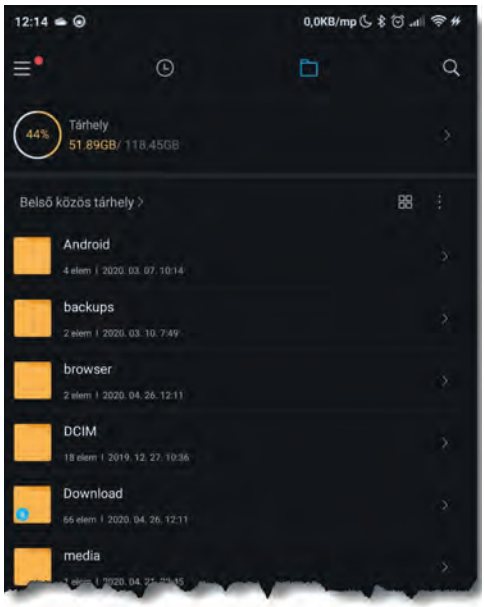
A **rosszindulatú szoftverek elleni védekezés**nek több módja van, amelyek mindegyikét érdemes megtenni. Az elsődleges a tudatos géphasználat. *Megbízhatatlan forrásból nem telepítünk szoftvert*, hiába csábít, hogy egy egyébként drága szoftvert majd ingyen használhatunk. A számítógépes kártevők egy része e-mail mellékletként érkezhet meg hozzánk akár egy ismerősünk e-mail címéről. Ezért legyünk mindig elővigyázatosak a mellékletek megnyitásakor. A digitális eszközünkön – legyen az számítógép vagy akár mobiltelefon – legyen *víruspajzs program*, ami minden állományt megvizsgál használat előtt. Mivel a kártevők újabb változatai naponta jelennek meg, fontos, hogy a vírusirtó program naprakész legyen, ezért rendszeresen töltsük le a *vírusdefiníciós adatbázis*át. Ezt általában a vírusirtók automatikusan megteszik, csak engedélyeznünk kell számukra ennek végrehajtását. A rosszindulatú programok sokszor a programokban található biztonsági hibákat használják ki. A szoftvergyártók az ismertté vált hibákat rendszeresen javítják, és *frissítések kiadásával* juttatják el a felhasználókhoz. Ezeket a frissítéseket a megjelenésük után a lehető leghamarabb telepítsük eszközeinkre! Amennyiben az operációs rendszerünkön van *szoftveres tűzfal*, azt tartsuk bekapcsolva. Ha valamilyen program a tűzfal miatt nem működik rendesen, akkor ne a tűzfal kikapcsolása legyen a megoldás, hanem keressük meg, hogy milyen beállítások mellett lesz a programunk működőképes úgy, hogy közben a rendszerünk védett marad.

Készüljünk fel arra, hogy számítógépünk rosszindulatú támadás áldozatává válik. Ha más megoldás nincs, akkor az eszközünkön vissza kell állítani az operációs rendszer gyári állapotát. Ezzel mind a feltelepített programjainkat, mind az állományainkat (képeinket, dokumentumainkat) elveszítjük. Ezért készítsünk rendszeresen *biztonsági mentés*t adatainkról, programjainkról, és ezt tartsuk lehetőleg fizikailag külön az eszközünktől. Lehet egy külső tárolón, vagy akár egy felhős tárterületen. Ezt rendszeresen végezzük el, de ehhez akár beállíthatunk automatizált mentést is.

► Biztonsági mentés külső meghajtóra

Fájlkezelés

A digitális eszközeinken állományokat tárolunk, amelyek kezelését egy **fájlkezelő** megkönnyíti. Itt mappákba rendezve láthatjuk az állományokat. Magunk is hozhatunk létre új mappákat, és áthelyezhetjük állományainkat, hogy később könnyen megtaláljuk azokat. Például e könyv készítése közben a könyvbe kerülő képállományokat fejezetenként külön-külön mappákba gyűjtöttük, hogy a kiadványszerkesztéssel foglalkozó kollégák számára jól kezelhető legyen. Érdemes a különböző projektek anyagait, azon belül a munkaanyagokat, részben vagy egészben feldolgozott fájlokat elkülönítetten tárolni. Így amikor a végeredményt be kell adni, akkor könnyen össze tudjuk gyűjteni a beadandó állományokat, ha viszont még újabb módosítási igények merülnek fel, akkor vissza tudunk nyúlni az eredeti forrásokhoz. Amennyiben nem csak a saját eszközünkön szeretnénk tárolni az állományokat, esetleg meg szeretnénk osztani másokkal is az iskolában, akkor szükségünk lesz egy fájlszerver elérésére. Sok iskolában a helyben üzemeltetett szerveren lehet tárolni a tanuláshoz kapcsolódó anyagokat. Ilyenkor mindenkinek egyedi felhasználói azonosítóval és jelszóval kell rendelkeznie. Ezzel biztosítható, hogy mindenki a saját állományaihoz, mappáihoz hozzáférjen, a többiekét ne lássák, viszont legyenek olyan tárterületek, amit a diákok egymással, a tanárral is közösen használhatnak. A mai operációs rendszerek a helyi fájlkezelés mellett a hálózati fájlkezelést is alapszolgáltatásként biztosítják, általában a hálózaton tárolt adatok elérése nem, vagy csak kis mértékben tér el a helyi tárolásnál megszokottól.



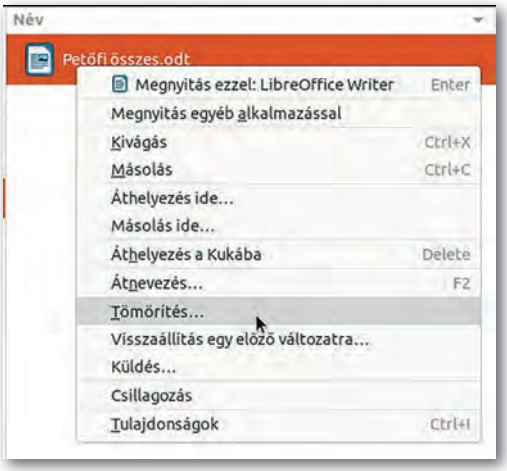
▶ Fájlkezelő Android alatt

Tömörítés

Rendszeresen előfordul, hogy elkészült munkánkat másokkal kell megosztanunk. Ilyenkor probléma lehet az állomány mérete, vagy esetleg az, hogy egy mappán belül sok állományt, esetleg mappát kellene átadnunk. Ebben az esetben segíthet a tömörítés. Elsőként gondoljuk végig, mit is jelent a tömörítés. Amikor adatokat akarunk tárolni, akkor az elfoglal valamekkora helyet a memóriában vagy a háttértáron. Hogyan lehet ezt tömörebben, kisebb helyen tárolni? Ehhez fontos tudnunk, hogy hogyan tároljuk az adatokat. A számítógépes grafika fejezetben a pixelgrafikus képszerkesztésnél volt már pár szó a tömörítésről. A tömöríthetőséget grafikus példával könnyű szemléltetni.

Képzeljünk el egy olyan rajzot, ahol a kép felső fele kék, alsó fele zöld színű. Ekkor tárolhatjuk minden egyes képpont színét egy számértékkel. Így a kép méretével egyenes arányban nő a tárolandó adat mennyisége. Ezzel szemben tárolhatnánk az egymás utáni azonos színű képpontok esetében azt, hogy hány darab milyen színű képpont következik. A példánkat megnézve látható, hogy ezzel a tárolással az eredeti tárolási mérethez képest jelen-

tősen kisebb tárterületre lesz szükségünk, de szükség esetén képpontról képpontra vissza tudjuk állítani az eredeti képet. A kép mérete kisebb lesz, cserébe számítási kapacitásokat igényel a folyamat. Ez a tárolási mód veszteségmentes, azaz teljesen pontosan visszaállítható az eredeti adathalmaz. A példa alapján elképzelhetjük, hogy nem csak képeken, hanem más állományokon is van lehetőségünk ismétlődő, vagy gyakran megjelenő mintázatokat találni, amik rövidített formában helyettesíthetők.



► Tömörítés – Ubuntu fájlkezelő

Amikor egy vagy több állományt szeretnénk másnak átadni, és ezért tömörítjük, akkor egyértelműen veszteségmentes tömörítésre van szükségünk. Az operációs rendszerbe integrált, vagy külön programként megjelenő fájltömörítők így működnek. A leggyakrabban használt ilyen tömörítési formátum a **ZIP**. Az operációs rendszerek fájlkezelő alkalmazásaiban általában a fájlokhoz vagy mappákhoz kapcsolódóan előhívható helyi menüben is megjelenik ez a lehetőség, ahogy az a képen is látható. Amennyiben több állományt vagy mappát jelölünk ki, akkor a tömörítés eredményeként egy darab tömörített állományt kapunk. Ez akkor is jól jöhet, ha sok állományt kellene továbbítanunk, mert így egy csomagban tehetjük ezt meg. Ebben az esetben a tömörített állományban a mappaszerkezet is tárolódik, azaz kibontás után ugyanabban a struktúrában lesznek elérhetőek az állományok, mint ahogy eredetileg a tömörítés előtt voltak. Fontos tudnunk a veszteségmentes tömörítés kapcsán, hogy egy már tömörített állomány tömörítésével már csak nagyon korlátozott mértékben tudunk helyet megtakarítani, ha egyáltalán lehet. Amennyiben olyan állományt próbálunk tömöríteni, ami már eleve (veszteséges vagy veszteségmentes) tömörített formátum, akkor az eredeti mérethez nagyon közeli méretet kapunk eredményül. Tömörített formátumok például a képek esetén a JPG, a PNG, hangok esetén az MP3, a FLAC, videók esetén az MP4, a MOV. A Microsoft Word által használt DOCX, és a LibreOffice Writer által használt ODT is már tömörített formátum, ráadásul a ZIP állományok létrehozásához használt algoritmussal tömörítettek.

Felhőszolgáltatások

Felhőszolgáltatásoknak hívjuk azokat az informatikai megoldásokat, ahol egy üzleti vállalkozó a saját hardvereszközein üzemelteti a szolgáltatásokat, és a felhasználó elől az üzemeltetés részletei rejtve maradnak. Ilyenkor a felhasználó nem tudja, és nem is fontos tudnia, hogy fizikailag melyik számítógépeken biztosítják számára a szolgáltatást. A felhasználó annyit tapasztal, hogy az internethez kapcsolódva bárhol is van, mindig azonos módon éri el a szolgáltatásokat. A *helyfüggetlenség* mellett fontos, hogy *méretezhetőség*, más néven *skálázhatóság* jellemzi, azaz az igényeknek megfelelően képes növekedni. Amennyiben egy vállalkozás ilyen szolgáltatást vesz igénybe, és megnő a vállalkozás forgalma, akkor a szerverközpontokban üzemelő szolgáltatás több vagy nagyobb teljesítményű szerverekre tud automatizált formában költözni. Ennek természetesen megnövekvő költségei lesznek, de

nagyobb forgalomból várhatóan nagyobb bevétel is következik. Mivel a szerverek üzemeltetését erre szakosodott szervezet végzi, a szolgáltatás *rendelkezésre állás*a is magasabb, kevesebb rendszerhibából adódó időkieséssel kell számolni. Ez átlagosan éves szinten legfeljebb néhány perc. Egy szerverközpontban a feladatok a nagyszámú gép között jól elosztha-tók, így egy cég igényeinek megfelelő szolgáltatás sokkal költséghatékonyabb, mintha azt saját eszközökkel, saját alkalmazásban lévő rendszergazdával kellene megoldania.

A felhőszolgáltatásoknak az egyik nagy előnye, hogy több lehetőséggel támogatják a csapatmunkát. Sokszor az alapszolgáltatásokat ingyenesen elérhetővé teszik a szolgáltatók, csak a nagyobb tárterületért, speciális funkciók eléréséért, esetleg a reklámmentességért kell havidíjat fizetni.

Nézzük a teljesség igénye nélkül, hogy egy projekthez a felhőszolgáltatásokból mit vehetünk igénybe. Legyen első a kommunikáció!

Lehetőségünk van *levelezőszolgáltatás*t igénybe venni. Sokak által használt a Gmail, az Outlook, a Microsoft365, a ProtonMail, a Yahoo Mail.

A levelezés mellett a kapcsolattartás rövid, azonnali szöveges üzenetekben is zajlik, ez a *csevegés* (angolul chat). Gyakran használt csevegő szolgáltatás a Messenger, a Viber, a Skype, a WhatsApp. Ha hang alapon szeretnénk kommunikálni, akkor azt általában a csevegőprogramokkal is megtehetjük.

*Videóhívás*okra is lehetőséget szoktak biztosítani a csevegőprogramok, de vannak olyan szolgáltatások, amik kifejezetten ebben erősek. Ilyen például a Microsoft Teams, a Webex Meetings, a Zoom, a Google Meet.

▶ Videókonferencia

Ha állományokat szeretnénk tárolni, másokkal megosztani, közösen használni, akkor a felhős tárhelyszolgáltatásokat vehetjük igénybe, például a Microsoft OneDrive, a Google Drive, a Dropbox. Ezeknél lehetőségünk van automatikus szinkronizálás beállítására. Ha beállítjuk, akkor a telefonunkon készített fényképek automatikusan a felhős tárterületünkre másolódnak, így nem kell külön gondoskodnunk azok biztonsági másolatáról. Ilyenkor beállíthatjuk, hogy a mobilhálózati adatforgalom esetén is engedélyezett legyen a szinkronizálás, ami külön költségeket jelenthet, vagy csak a wifikapcsolat esetén induljon a másolás. Ehhez hasonlóan a számítógépünkön tárolt állományokhoz is beállíthatjuk a szinkronizálást, amivel a biztonsági mentésünk lesz biztosított. Azon túl, hogy a magunk számára eltárolhatunk állományokat, lehetőségünk van azokat másokkal megosztani. A megosztásnál meghatározhatjuk, hogy azt mindenki elérheti-e, vagy csak az általunk megadottak. Beállíthatjuk, hogy csak megtekinteni tudják, vagy lehetőségük legyen módosítani is azokat. Van olyan szolgáltatás, ahol még időkorlátot is be lehet állítani, például 3 napig elérhető a linken, utána már nem lesz megosztott. Ha egy projektfeladat kapcsán a mobiltelefonunkkal készítünk egy videót vagy fényképeket, akkor azokat a telefonunkról a felhős tárterületre másolhatjuk, majd azt a csapatunk többi tagjával megoszthatjuk. Ha egy megosztott mappába újabb állományokat teszünk, akkor azt a többiek azonnal elérik.

Ha *dokumentumokat* szeretnénk *szerkeszteni* a többiekkel egyidőben, akkor használhatjuk például a Google Dokumentumok vagy a Microsoft365 szolgáltatást.

A felsorolt felhőszolgáltatásokban közös, hogy igénybevételük *eszközfüggetlen*, azaz laptopon, táblagépen, okostelefonon egyaránt elérhetőek, sok esetben még alkalmazást sem kell telepíteni, elegendő egy böngészőből csatlakozni a szolgáltatás weboldalához. Itt egy felhasználói azonosítót és jelszót kell általában megadni. A biztonság növelhető az úgynevezett kétfaktoros hitelesítéssel, amikor a bejelentkezéshez nem elegendő e két adat, hanem egy fizikai eszközre is szükség van. A leggyakoribb megoldás, hogy a bejelentkezéskor kapunk a mobiltelefonunkra egy egyszer használatos kódot, amit a bejelentkezési felületen meg kell adnunk, vagy csak a mobiltelefonunkon nyugtázni kell a bejelentkezési kísérletet. Erre azért van szükség, mert egy felhasználónév és jelszó párost megszerezhetnek például egy kémprogram segítségével, de ennek segítségével nem tudnak a nevünkben bejelentkezni, ha a telefonunk mindeközben nálunk van.

Feladatok, kérdések

1. Nézzük meg, hogy mobiltelefonunkat be tudjuk-e úgy állítani, hogy azt kölcsönadva csak a telefonálás szolgáltatást tudják használni, az üzeneteinket, fényképeinket ne tudják megnézni, a telepített programokat ne tudják elindítani, nevünkben ne tudjanak üzenetet küldeni!
2. Hogyan lehet egy mobiltelefonra letöltött fájlt másik mappába áthelyezni?
3. Milyen módokon tudunk egy mobiltelefonnal készített fényképet a teremben lévő osztálytársunknak átküldeni? Milyen lehetőségeink vannak nagyobb távolság esetén, például, ha iskolaidő után már otthon vagyunk, és egy közös projekthez szükséges megosztanunk egy általunk készített képet?
4. Milyen műveleteket lehet végezni a szövegszerkesztő programban a funkcióbillentyűk segítségével? Nézzük meg a Shift, a Ctrl és az Alt billentyűkkel együttes használatot is!
5. Hogyan lehet egy tömörített állományt jelszóval védetté tenni?
6. Egy képeket és szöveget is tartalmazó DOCX vagy ODT állomány kiterjesztését írjuk át ZIP-re, majd bontsuk ki! A megkapott könyvtárak és állományok között keressük meg a szövegszerkesztővel elkészített állományban lévő szöveget és képeket!
7. Milyen felhőszolgáltatásokat érdemes használni egy 3-4 fős csapatban megoldandó projekt kapcsán? A különböző projektek a használandó eszközökben is eltérhetnek, így érdemes többféle projekt esetében is végiggondolni a választ.