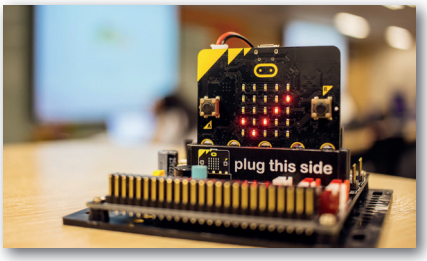
Programozzunk micro:biteket!

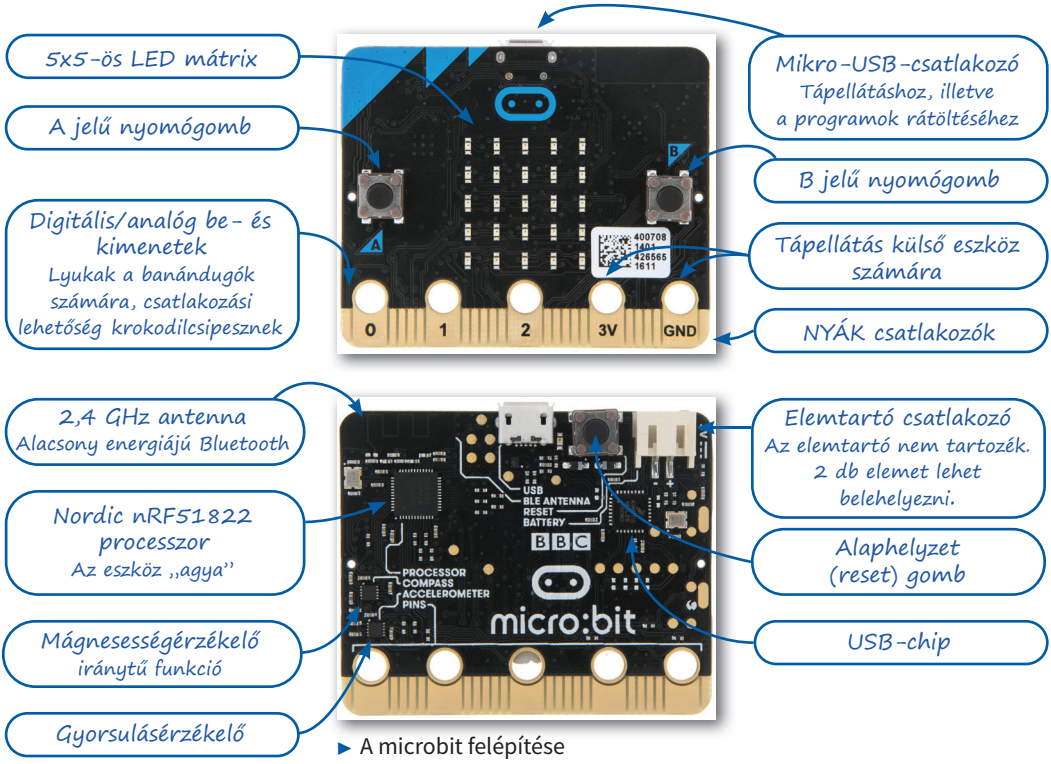
A következőkben egy olyan eszközt fogunk megismerni, amely valódi szenzorokkal van felszerelve, és blokkprogramozási környezetben is programozhatjuk. Ez nem más, mint a micro:bit.



A micro:bit egy oktatási célra kifejlesztett, egyetlen lapkán megvalósított miniszámítógép. Található rajta egy 5x5-ös LED kijelző, amelyen számokat, szövegeket és különböző ikonokat, animációkat jeleníthetünk meg.

Sokféle érzékelővel el van látva. Rendelkezik gyorsulásérzékelővel, hőmérséklet-érzékelővel, fényérzékelővel, irányérzékelővel.

Van két gombja (A és B jelű), amelyek megnyomására reagálni tud. A be- és kimeneti csatlakozói lehetővé teszik, hogy más eszközökkel is össze lehessen kötni. A Bluetooth kapcsolatnak köszönhetően pedig a micro:bitek egymással is képesek kommunikálni.



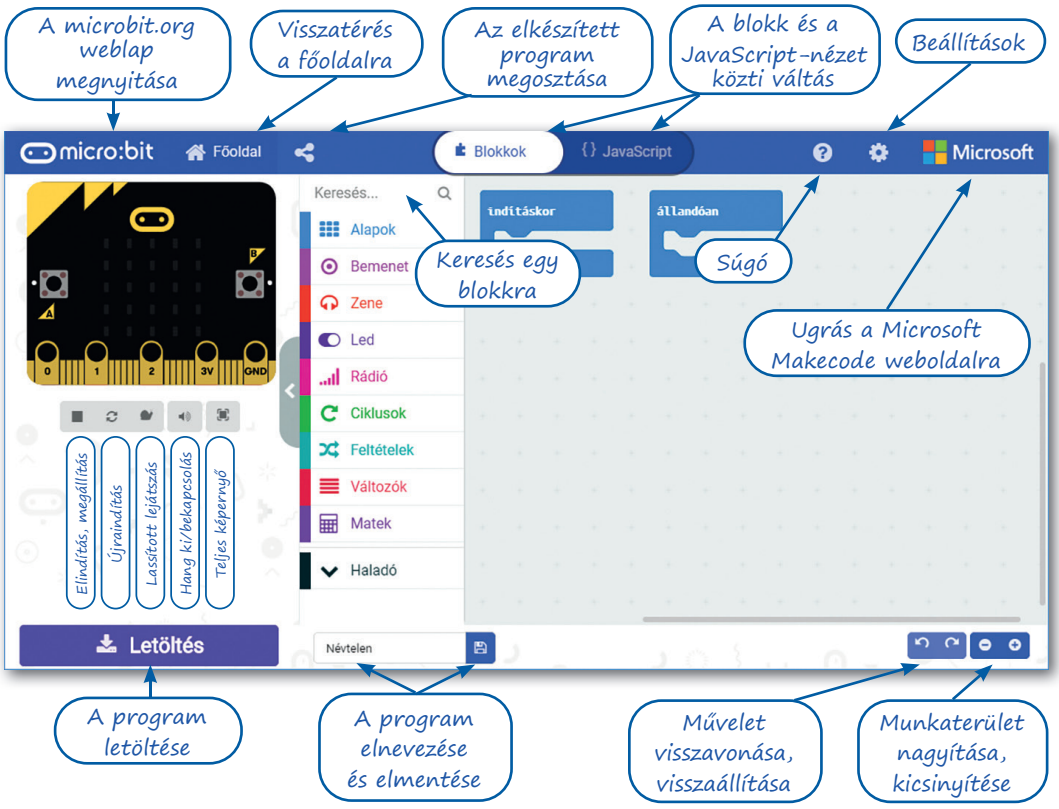
► A microbit felépítése

Az eszközt sokféle programozási nyelven lehet programozni, köztük több blokkprogramozási nyelven is. De nagy különbség van abban, hogy az egyes környezetekben milyen funkciókat érhetünk el. A legtöbb funkciót a MakeCode környezet biztosítja. Ennek van online elérhető (https://makecode.microbit.org/) és letölthető változata is.

Ismerkedés a felülettel

A *MakeCode* alkalmazás felülete nagyon hasonlít a korábban látott blokkprogramozási környezetekhez. A fő különbség, hogy most nem egy szereplőt irányítunk a képernyőn, hanem egy fizikailag is létező eszközre készítünk programokat.

Az elkészült programot egy szimulátor segítségével ki is próbálhatjuk, de ha rendelkezünk micro:bittel, akkor a programot az eszközre letölthetjük, és maga az eszköz fogja végrehajtani azt.



► A Makecode programozási felülete

A beépített szenzorok miatt a micro:bit képes reagálni a környezetből érkező hatásokra. Ilyen lehet, amikor valamelyik irányba megdöntjük az eszközt, megnyomjuk valamelyik vagy mindkét gombját, ha sötét helyre visszük, ha észak felé fordítjuk, ha megrázzuk, és így tovább.

► Néhány esemény, amelyekre az eszköz reagálni képes

Mire ügyeljünk?

Amikor a micro:bitekkel dolgozunk, ügyeljünk a következőkre!

- Mindig tiszta asztalon dolgozzunk! Különösen veszélyesek az eszköz számára a rövidzárlatot okozó tárgyak, mint pl. gemkapocs, tűzőgépkapocs, körző, egy ceruza kitört grafithegye, kifolyt üdítőital stb.
- Az elektrosztatikus kisülés akár tönkre is teheti az eszközt, ezért fontos, hogy mielőtt az eszközhöz hozzáérnénk, érintsük meg a számítógép házát, vagy a tanteremben lévő radiátor vagy fűtési csőhálózat felületét!
- Az eszköz úgy legyen elhelyezve az asztalon, hogy ne eshessen le, illetve véletlenül se lehessen lesodorni azt!
- Az egyes gesztusok kipróbálása során (pl. rázás) figyeljünk arra, hogy biztosan tartsuk az eszközt a kezünkben, ne ejtsük azt le!
- A tevékenységek során se az eszközben, se a társainkban ne tegyünk kárt!
- Elemről csak addig működtessük az eszközt, ameddig feltétlenül szükséges, egyébként használjuk az USB-kábelt!
- Ne kössünk az eszköz kivezetéseihez olyan külső eszközöket, amelyek pl. nagyobb tápfeszültséget igényelnek, mint amit az eszköz biztosítani tud! Ez akár tönkreteheti az eszközt, és az esetleges túlmelegedés miatt égési sérüléseket is okozhat!
- Az eszköz érzékelőit megzavarhatják külső tényezők, pl. erős mágneses tér, fémtárgyak stb. Ha a szenzorok nem megfelelő értékeket mérnek, vizsgáljuk meg, hogy nincs-e a környezetben valamilyen zavaró tényező!

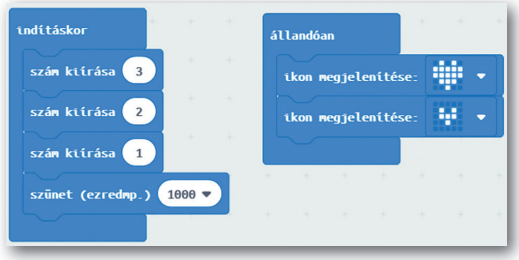
Készítsünk animációt!

Kezdjük az eszközzel való ismerkedést azzal, hogy a LED kijelzőn jelenítsünk meg különböző ábrákat és animációkat!

A munkaterületen alapesetben két blokkot is találunk. Az *indításkor* blokk tartalma egyszer, a program elindulásakor hajtódik végre. Az *állandóan* blokk tartalma állandóan ismétlődik, csakúgy, mint a korábban látott végtelen ciklus esetén.

Mindkét blokk kék színű, ez egyben arra is utal, hogy melyik kategóriába tartoznak. Ez most az Alapok kategória.

Feladatok

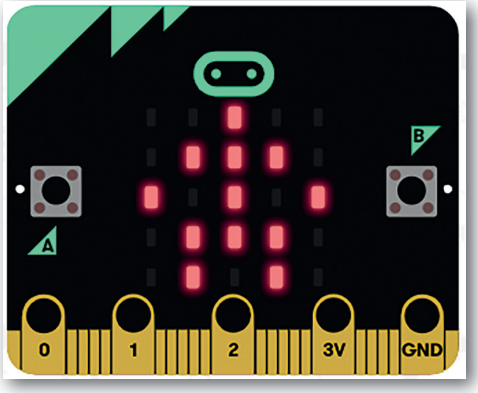


▶ Mit csinálnak a blokkok?

1. Próbáljuk ki, hogy az itt látható blokkok hatására mi fog történni a szimulátorban/eszközön! Mit tapasztalunk?
2. Bővítsük úgy a kódot, hogy indításkor az eszköz írja ki a keresztnevünket! Az ehhez szükséges blokk szintén az Alapok kategóriában található. Próbáljuk ki a programot!
3. Láthattuk, hogy a programban nemcsak szívecskét, hanem sok más ikont is meg lehet jeleníteni. Cseréljük le a szív ikonokat több, általunk választott ikonra. Bővítsük az állandóan blokk tartalmát úgy, hogy az animáció legalább öt ikonból (fázisból) álljon!
4. Nincs olyan ikon, amire szükségünk van? Semmi gond, rajzoljunk egyet kedvünk szerint! Készítsünk olyan animációt, amelynek minden fázisát mi rajzoltuk meg. Az ehhez szükséges blokkot szintén az Alapok kategóriában kell keresni.
5. A micro:bit logója pont a LED kijelző tetejénél van elhelyezve, és úgy néz ki, mintha egy robot feje lenne, két szemmel. Használjuk ki ezt, készítsünk egy robotot, amely tud integetni!

A *Bemenet* kategóriában találunk olyan vezérlőblokkokat, amelyekkel megoldható, hogy a blokkok akkor hajtódjanak végre, amikor megnyomtuk az *A* vagy a *B* gombot, vagy mindkettőt egyszerre.

Készítsünk olyan animációt, amelyben az *A* gombot lenyomva a robot az egyik karjával integet, a *B* gomb hatására pedig a másikkal! Ha mindkét gombot egyszerre megnyomjuk, akkor pedig csináljon valami egyedi mozgást, amit mi találunk ki. Az integetés legalább kétszer ismétlődjön. Keressük meg a megfelelő blokkot ehhez a Ciklusok kategóriában. Az integetés után a robot vegye fel a kiindulási helyzetét! Próbáljuk ki a projektet! Mentsük el a programot, hogy következő alkalommal folytathassuk!

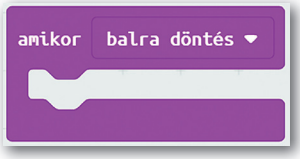


Használjuk az érzékelőket!

Korábban megismerkedtünk azzal, hogy a LED kijelzőn hogyan jeleníthetünk meg ikonokat, animációkat, és már az *A* és *B* gombokat is használtuk. Most továbblépünk úgy, hogy a micro:bit érzékelőiben rejlő lehetőségeket is kihasználjuk.

Feladat

Az *A* és *B* gombokkal működtetett animációs projektet fejlesszük tovább úgy, hogy ne csak a gombokra reagáljon a micro:bit. Az eszköz balra döntésekor integessen a robot az egyik kezével, jobbra döntéskor pedig a másikkal! Amikor megrázzuk az eszközt, akkor pedig mindkét karját emelje fel, majd engedje le!



► A Bemenet kategória egyik blokkja

Tipp: Ezeket a gesztusokat akár a szimulátorban is kipróbálhatjuk a virtuális micro:bit mozgatásával, illetve a Rázás (shake) gomb megnyomásával.

Készítsünk számlálót, ami egyben dobókocka is!

A micro:bitet akár elemről is lehet működtetni, a rátöltött programot pedig egész addig képes végrehajtani, míg egy másik programot rá nem töltünk. Ez azt jelenti, hogy a számítógéptől függetlenül, akár a szabadban is használni tudnánk.

A következőkben egy olyan alkalmazást készítünk, amellyel egyszerűen megszámolhatunk különböző dolgokat, sőt akár dobókockaként is működhet. Ezt az alkalmazást sokféle célra felhasználhatjuk:

- Megmérhetjük, hogy hány autó haladt el az iskola melletti útszakaszon adott idő alatt. Ha látunk egy autót, növeljük meg a számlálót.
- Egy meccsen megmérhetjük, hogy hány labdaérintés kellett ahhoz, hogy gól szülessen. A gól után lenullázhatjuk a számlálót, és folytathatjuk a mérést.
- Bármilyen társasjátéknál nyilvántarthatjuk, hogy hányszor nyertünk. Sőt, igazi dobókocka helyett használhatjuk a micro:bitet is.
- A tanórai feladatoknál véletlenszerű csoportbeosztást tudunk csinálni (nem csak digitáliskultúra-órán!). Akik azonos számot dobnak a virtuális dobókockával, azok egy csoportba kerülhetnek egy feladat megoldásakor.

Feladat

Gondoljuk át, hogy milyen más célra tudnánk használni egy ilyen eszközt, a saját érdeklődési körünkhöz, hobbinkhoz kapcsolódóan! Beszéljük meg, hogy kinek milyen ötlet jutott eszébe!

Hogyan tároljuk a számláló értékét?

A terv az, hogy egy olyan alkalmazást készítsünk, amelyben ha megnyomjuk a *B* gombot, akkor mindig növekedjen eggyel a számláló a micro:bit kijelzőjén. Az *A* + *B* gomb hatására pedig nullázódjon le a számláló. Az *A* gomb megnyomásakor véletlenszerűen jelenjen meg egy szám 1 és 6 között, mintha egy dobókockával dobtunk volna.

Azt már megismertük, hogy az *A* és *B* gombok lenyomását figyelő vezérlőblokkot hogyan kell használni. De hogyan tudnánk eltárolni, sőt növelni egy számot, valamint azt megjeleníteni a kijelzőn?

Változók használata

Ahhoz, hogy számokat, szövegeket el tudjunk tárolni egy program során, meg kell ismerkednünk a változókkal.

Az adatokat a program végrehajtásakor úgynevezett **változókban** tárolhatjuk. Az adat lehet szám, szöveg, sőt más, összetettebb adat is. A változó onnan kapta a nevét, hogy a program végrehajtása során a tartalma **változhat, módosulhat**.

A változóknak egyedi nevet kell adnunk. A változó értékére pedig a változó neve alapján hivatkozhatunk.

Nézzük meg a számláló alkalmazás kapcsán, hogy a gyakorlatban hogyan használhatnánk a változókat.

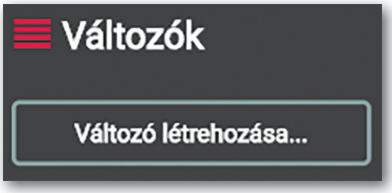
A programunkat azért készítjük el, hogy megszámoljunk különböző dolgokat. Ezt az adatot egy változóban fogjuk eltárolni.

Amikor **elnevezzük a változót**, akkor érdemes olyan nevet adni neki, amely alapján később is jól tudjuk majd azonosítani. Most *számláló* néven hozzunk létre egy változót!

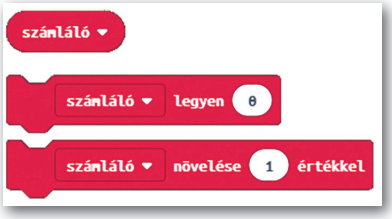
Miután ezt megtettük, a *Változók* között megjelenik egy ovális alakzat, benne a megadott névvel. Ezt a blokkot tudjuk használni akkor, ha le akarjuk kérdezni, hogy éppen milyen értéket tartalmaz a változó.

De nem csak ez a blokk jött létre, hanem a változó értékének beállítására szolgáló blokk is. Az érték szintén egy ovális alakzatban van elhelyezve. Ide számot is írhatunk majd, de akár szöveg is lehet benne, sőt különböző matematikai műveletek eredménye is.

Szintén létrejön egy blokk, amely a számláló értékét eggyel megnöveli. (Ha negatív előjelet használunk, akkor akár csökkenthetjük is az értéket.)



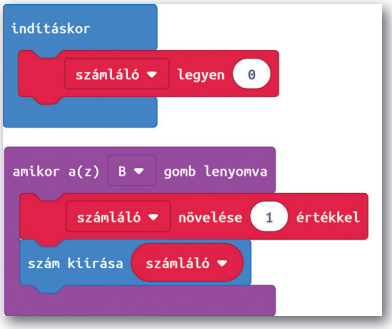
► Változó létrehozása a MakeCode felületen (Változók kategória)



► A létrehozott változóhoz tartozó blokkok

Feladatok

1. Készítsük el az itt látható programot, és próbáljuk ki a működését!
2. Módosítsuk úgy a programot, hogy az *A* + *B* gomb hatására is nullázódjon le a változó értéke!
3. Fejlesszük tovább az alkalmazást úgy, hogy az *A* gomb hatására egy véletlenszerűen meghatározott szám kerüljön bele a dobókocka nevű változóba, majd jelenítsük meg a változó értékét. Nézzünk körül a Matek kategóriában, hogy melyik blokkal lehetne véletlenszámot létrehozni!



► A számláló program egy részlete

Készítsünk egy játékot!

Van egy eszközünk, amelyen különböző ikonokat jeleníthetünk meg, illetve reagál arra, ha megrázzuk, megdöntjük valamelyik irányba. Ezen egyszerű elemek felhasználásával már akár játékokat is készíthetünk.

Tegyük fel, hogy igazságosan akarjuk eldönteni, hogy mi vagy a testvérünk ehesse meg az utolsó szelet csokit. Vagy a véletlenre szeretnénk bízni, hogy kinek kell kivinnie a szemetet a kukába. Ilyenkor akár kő, papír, olló játékot is játszhatunk. De erre a játékra akár a micro:bitet is megtaníthatjuk.

Ennek kapcsán pedig olyan új dolgokkal is megismerkedünk, amelyeket más játékok készítésénél is fel tudunk használni.

► A kő, papír, olló játék kézjelei

Gondoljuk át, hogy mi lehet egy kő, papír, olló játék algoritmusa! Valójában annyi történik, hogy véletlenszerűen kiválasztjuk, hogy három lehetőségből (kő, papír, olló) melyiket fogjuk a kezünkkel mutatni. Ez nagyon hasonló ahhoz, mintha egy 1 és 3 közötti számra gondolnánk, és a szám értékétől függően más-más kézjeleket mutatnánk.

Azt, hogy egy változó értékétől függően más-más dolog történjen, azt elágazással tudjuk leírni. De mi az az elágazás?

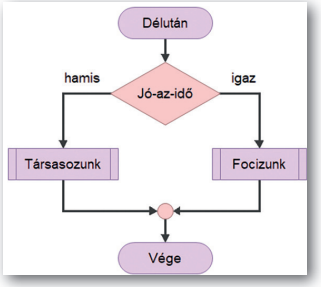
Elágazások a hétköznapi életben

Elágazásokkal a hétköznapi életben is gyakran találkozunk. Az alábbi mondatok mind ezt szemléltetik:

Ha hazaérsz 4 óráig, (**akkor**) vidd el a kutyát sétálni!
Ha tüzet észlelsz, (**akkor**) nyomd meg a tűzjelzőt!
Ha látsz egy hullócsillagot, (**akkor**) kívánj valamit!

Ezek a mondatok egyszerű elágazásra mutatnak példát. Csak arra az esetre adtunk meg utasításokat, ha a feltétel igaz. Ha hozzátesszük azt is, hogy különben mi történjen, akkor már kétirányú elágazásról beszélhetünk:

Ha délután jó az idő, (**akkor**) focizunk, **különben** társasozunk.
Ha jobban érzed magad, (**akkor**) menj át a nagyihoz, **különben** maradj itthon pihenni.



► Kétirányú elágazás folyamatábrán szemléltetve

Az **elágazásban** egy utasítás vagy egy utasításcsoport végrehajtását feltételhez tudjuk kötni. **Egyszerű elágazás** esetén az utasítások akkor hajtódnak végre, ha a megadott feltétel **igaz**.

Kétirányú elágazásban már azt is megadjuk, hogy milyen utasítások hajtódjanak végre akkor, ha a feltétel nem igaz, vagyis hamis.

Feladat

Fogalmazzunk meg mi is olyan mondatokat, amelyek egyirányú, illetve kétirányú elágazásnak felelnek meg!

Elágazások és változók az algoritmusokban

Amikor algoritmizálunk, érdemes a változókkal és elágazásokkal kapcsolatban az itt látható megfogalmazást használni.

A *kő, papír, olló* játékunk algoritmusa így alakul:

```
Program
   gondoltszám:=véletlenszám(1 és 3 között)
   Ha gondoltszám=1 akkor kő kirajzolása
       különben
         Ha gondoltszám=2 akkor papír
         kirajzolása
             különben olló kirajzolása
         Elágazás vége
   Elágazás vége
Program vége
```

Láthatjuk, hogy az 1 és 3 között véletlenszerűen meghatározott számot egy olyan **változóban tároljuk el**, amelynek neve: gondoltszám. Amikor a változónak értéket adunk, akkor nemcsak egyenlőségjelet használunk, hanem egy kettőspontot is teszünk az egyenlőségjel elé.

Ha a program algoritmusának első sorát fel kellene olvasnunk, akkor így tehetnénk meg: *„A gondoltszám változó értéke legyen egyenlő egy 1 és 3 közötti véletlen számmal.”*

Az elágazásokat a következő mondatszerű leírásokkal adhatjuk meg:

Egyszerű elágazás esetén:

```
Ha feltétel akkor
     utasítás(ok)
elágazás vége
```

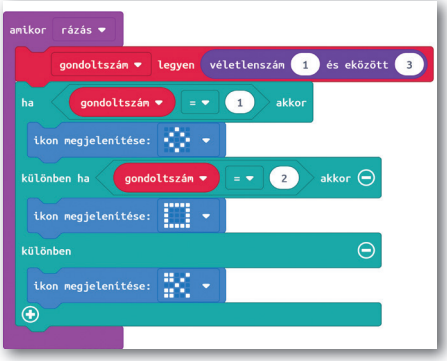
Kétirányú elágazás esetén:

```
Ha feltétel akkor
   utasítás(ok)
különben
   utasítás(ok)
elágazás vége
```

Háromirányú elágazás esetén:

```
Ha feltétel akkor utasítás(ok)
   különben ha feltétel2 akkor utasítás(ok)
                        különben utasítás(ok)
            elágazás vége
elágazás vége
```

Játékra fel!



► A programkód a MakeCode környezetben

Feladatok

1. Próbáljuk ki az itt látható kódot! Játsszunk le pár játszmát párokban!
2. Fejlesszük tovább a játékot úgy, hogy nyilván lehessen tartani, hogy hány játszmát nyertünk! A *B* gombbal lehessen növelni a számlálót, az *A* gombbal lehessen nullázni!
3. Játsszunk tojás, fióka, sas, főnix játékot! Kezdetben mindenki tojás állapotból induljon. Aki megnyer egy játszmát, fióka lesz. Ők egymás ellen játszanak párokban, majd aki nyer, sassá változik. A sasok egymás ellen fognak játszani. Aki nyer a sasok közül, az főnixmadárrá változik. Az nyer az osztályban, aki először eléri a főnixmadár szintet!
4. Hogyan lehetne továbbfejleszteni a játékot? Ötleteljünk párokban, és készítsük el a továbbfejlesztett alkalmazást!

Gyakorlás, saját ötletek megvalósítása

Most már egy új, kézzelfogható eszközt is megismertünk, amelynek kijelzőjén számokat, szöveget, ikonokat és animációkat is meg tudunk jeleníteni. Az eszköz szenzorai segítségével érzékelni tudja a környezetét, és akár gesztusokkal (pl. rázás) is irányítani lehet.

Ezek alapján újabb izgalmas programokat lehet megvalósítani.

Alkossunk együtt!

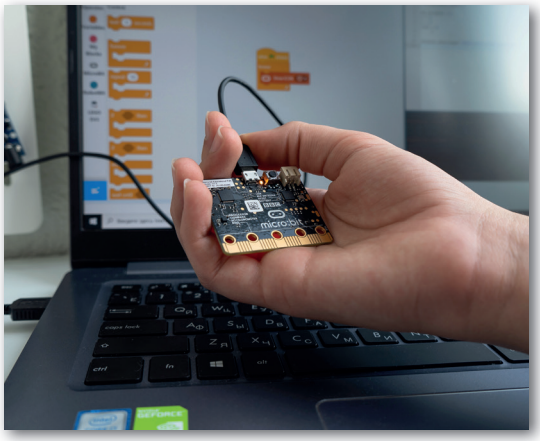
Alkossunk három-négy fős csoportokat! Gondoljuk át, hogyan lehetne továbbfejleszteni a korábban megvalósított számláló és dobókocka alkalmazást úgy, hogy minél több társasjátéknál lehessen használni segédeszközként! Gyűjtsük össze, hogy az általunk játszott és kedvelt társasjátékoknál milyen segédeszközöket kell használni, és azokat hogyan lehetne kiváltani micro:bittel!

Tervezzük meg az alkalmazást, majd valósítsuk is meg! Törekedjünk arra, hogy mindenki kapjon olyan részfeladatot, amiért ő a felelős!

Ha elkészültünk a munkánkkal, mutassuk be egymásnak a fejlesztéseteket!

Otthon pedig, amikor összeül a család egy jó kis társasjátékra, lepjük meg őket azzal, hogy a micro:bitek is szerepet kapnak a játékban!

Egyéni feladat



1. Készítsünk olyan animációt, melynek szereplője egy általunk kiválasztott állat. Az *A* és *B* gombok megnyomásakor, valamint a különböző események hatására (pl. rázás, döntés) más-más animáció játszódjon le (pl. menjen balra, jobbra, ugorjon egyet stb.).
2. Készítsünk egy programajánló alkalmazást, amely különböző tevékenységeket ajánl! Ha unatkozunk, csak megrázzuk a micro:bitet, és a megjelenő ikon alapján már akár cselekedhetünk is (pl. zenehallgatás, sport, kirándulás stb.)!
3. Készítsünk egy időjárás-előrejelző alkalmazást. Az *A* gomb hatására véletlenszerűen napsugár, esőfelhő vagy hópihe alak jelenjen meg. A *B* gomb hatására pedig jelenjen meg számként, hogy a micro:bit hőmérséklet-érzékelője hány fokot érzékel.

Páros feladat

Készítsünk egy számkitalálós játékot! A játék során azt kell megtippelnünk, hogy a micro:bit milyen számra fog gondolni. Aki eltalálja, az kap egy pontot. A játékot felváltva lehet játszani. A program úgy működjön, hogy a micro:biten a *B* gomb megnyomásával lehessen növelni a kijelzőn megjelenő szám értékét, az *A* gomb hatására pedig egy véletlen szám jelenjen meg 1 és 9 között.