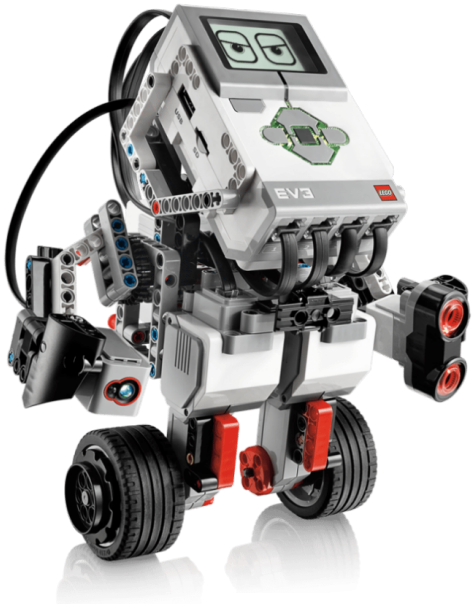
Belépő a tudás közösségébe

Lego Mindstorms EV3 robotok programozása

Szakköri segédanyag tanárok számára

Barbalics Dóra Krisztina, Solymos Dóra



SZÉCHENYI 2020
MAGYARORSZÁG KORMÁNYA
Európai Unió
Európai Szociális Alap
BEFEKTETÉS A JÖVŐBE

Lego Mindstorms EV3 robotok programozása

Szerzők

Barbalics Dóra Krisztina, Solymos Dóra

Felelős kiadó

ELTE Informatikai Kar

1117 Budapest, Pázmány Péter sétány 1/C.

ISBN szám

ISBN 978-963-489-040-9

A kiadvány „A felsőoktatásba bekerülést elősegítő készségfejlesztő és kommunikációs programok megvalósítása, valamint az MTMI szakok népszerűsítése a felsőoktatásban”
(EFOP-3.4.4-16-2017-006) című pályázat keretében készült 2018-ban.

Tartalomjegyzék

1. Bevezető

A LEGO© Education közel 37 éve dolgozik együtt tanárokkal és oktatási szakértőkkel, hogy olyan eszközöket fejlesszenek ki, amivel a tanulás játékos tapasztalatszerzés lehet. Digitális és fizikális oktatási eszközök széles kínálatával próbálják a diákokat kreatív gondolkodásra és problémamegoldásra ösztönözni, mellyel akár jövőjüket is megalapozhatják.[1]

A tanításra és a gyakorlati tanulásra kínált megoldásuk érdeklődést kelt a gyerekekben a matematika, a természettudományok, a műszaki tudományok és az informatika (a magyar szakirodalomban is bevett mozaikszóval: MTMI) iránt.
Ez különösen fontos napjainkban, ugyanis a fejlett országokban megfigyelhető általános jelenség, hogy az MTMI területei iránti érdeklődés csökken a fiatalok körében. Mindez aggodalomra adhat okot, hiszen a tudásalapú gazdaságban nagy szükség van az MTMI készségekkel rendelkező szakemberekre. Hiányuk a fejlődés gátja lehet.

A LEGO© szakemberei úgy gondolják, hogy az ismeretszerzés és az elméleti oktatás a 21. századi szakértelemmel kombinálva, aktív és együttműködő élethosszig tanulókat (angolul: lifelong learners) nevel. A cég az oktatókkal együtt, azt a célt tűzte ki maga elé, hogy eszközeivel képessé tegyen minden diákot arra, hogy azok sikereket érjenek el a tanulásban és felkészüljenek a jövő kihívásaira.

Tananyagunkkal ehhez és a programozás népszerűsítéséhez szeretnénk további segítséget nyújtani 12 kilencvenperces szakköri óra bemutatásával, melyet a 10-15 éves korosztálynak ajánlunk.

A szerzők

[1] `https://education.lego.com/en-us/about-us`

2. A szakkör felépítése

A szakköri anyagot a LEGO© Mindstorms EV3 robottal ismerkedő tanároknak és diákoknak (főként 10-15 éveseknek) készítettük. A szakkör 12 db 90 perces órából áll, melyeken a diákok kettesével használnak egy-egy felépített robotot.

Eszközszükséglet 10 diákra és egy tanárra:

- 11 db asztali számítógép vagy laptop egérrel
- 5 db LEGO© Mindstorms EV3 összerakva
- 5 vagy 10 db USB kábel a számítógép és a robot összekötéséhez
- 1 db projektor

Tananyagunk főként informatika tanárok számára készült, így az alapvető programozással kapcsolatos fogalmakra (pl.: ciklus, többágú elágazás) nem térünk ki. Ezért arra szeretnénk kérni Titeket, hogy ezeket a fogalmakat az adott szakköri alkalmon a tananyagban szereplő feladatok és példák segítségével magyarázzátok el a diákoknak.

Kiadványunk tematikája rendhagyónak számít abban a tekintetben, hogy a hangsúlyt nem a programozással kapcsolatos alapfogalmakra tesszük, hanem inkább a robot megismerésére, használatára. A fogalmak megismerése mintegy melléktermé ennek a folyamatnak, így a gyerekek észre sem veszik, hogy milyen nehéz dolgot tanulnak.

Az általunk bemutatott tematika csak egy a sok közül, nyugodtan módosítsátok saját habitusotok, szájízetek szerint.
A tananyagot a jövőben szeretnénk bővíteni, ezért minden visszajelzést szívesen fogadunk a `https://www.facebook.com/tethalo/` facebook oldalunkon.

2.1. A könyv felépítése, jelölések

Minden szakköri alkalom egy újabb elem megismerésével kezdődik, majd annak gyakorlásával folytatódik. Általában egy órán kétféle új dologgal ismerkednek meg a diákok és kezdik el gyakorolni a használatát. Mindketten több feladatot készítettünk egy-egy órához, mint amennyi biztosan beleférne egy alkalomba, de ez nyilván csoportfüggő.

Volt szerencsém olyan csoporttal dolgozni, ahol nem győztem pluszfeladatokat adni, de még azok is kevésnek bizonyultak, így mindig egy EV3 robot sumo videó megnézésével zártuk az órát. Aminek legalább annyira örültek a fiúk, mint egy újabb feladatnak.

Visszatérve a tananyagra, a 2-9. alkalmak egy-egy Kahoot![2] teszttel indulnak.
A Kahoot egy online, ingyenes feladatsor-készítő program. Négy különböző feladattípus közül választhatunk. Mi minden szakköri alkalom elejére egy kvíz jellegű feladatsort ajánlunk, így téve izgalmassá a már tanult anyagok ismétlését.
Ehhez nem kell más, mint egy tanári gép, melyen a feladatokat ki tudjuk vetíteni, illetve a diákoknak egy-egy gép, tablet vagy okostelefon melyen a helyes választ jelölő színes téglalapot ki tudják választani. Lehetőséget ad a program csoportmunkára is. Az egyéni verziótól ez csupán abban különbözik, hogy a diákok nem becenevet, hanem csapatnevet választanak a teszthez való csatlakozáskor, illetve együtt döntik el, melyik legyen a megjelölt válasz.
Kvízünkből versenyt is készíthetünk! Ehhez be kell állítanunk a programban, hogy pontozza a válaszadókat. A pontszámban szerepet játszik a helyes válasz eltalálása, illetve másodlagos tényezőként a gyorsaság is.
Feladatsorunkat akár képekkel, videókkal is színesíthetjük. Ezeket a szerkesztőben tudjuk beilleszteni, a kitöltés során pedig a kérdés szövege alatt fognak megjelenni.

A Kahoot tesztek kérdéseit egy narancssárga szegélyű dobozba helyeztük (olyanba, mint például ez itt). Ezek a „feladatok" sem kötelező jellegűek, csak lehetőségek.

Törekedtünk arra, hogy mindent szemléltessünk, hogy akár az alapoktól is könnyen, önállóan elsajátítható legyen a robot használata, így elég sok ábrát helyeztünk el a szövegben. Minden ábra rendelkezik egy sorszámmal. Ha egy szövegben találkoztok ilyennel (pl itt: 1. ábra), akkor a sorszámra kattintva átugorhattok a képre. (Utána felfelé görgess, hogy visszatalálj ide! :))

Hasonlóan „linkesítettük" a feladatokat is, azonban ott a feladat szövegére kell kattintanotok, hogy a megoldáshoz érjetek. Ahhoz, hogy visszaugorjatok a feladat eredeti helyére a megoldástól, újra a feladat szövegére kell kattintanotok. A tájékozódást még egyszerűbbé téve, a feladatok a szakköri alkalmaknál normál stílusúak, míg a megoldásoknál dőltek.

[2] `https://kahoot.com/`

Még nem ejtettem szót a lila/rózsaszín szegélyű dobozainkról. Nos, ezekben saját tapasztalatainkat, véleményünket szeretnénk megosztani Veletek, melyet már rögtön itt feljebb is olvashattatok.

Szerzők:
Solymos Dóra: Bevezető, A szakkör felépítése, Szakköri alkalmak (1-5)
Barbalics Dóra Krisztina: Kahoot tesztek, Szakköri alkalmak (6-12)

2.2. Az általunk használt tesztrobot

A szakköri alkalmak során az 1. ábrán is látható robotot szoktuk használni, mert elég sok érzékelőt bele tudunk építeni és sokféle feladat megoldására képes.
A robot építési útmutatói az alábbi honlapon érhetőek el (a „Building Instructions for Robot Educator" rész alatt) : `https://education.lego.com/en-us/support/mindstorms-ev3/building-instructions#robot`. Ennél a résznél elég sok hasonló útmutató van fent, mi a következőket használtuk:

„Color Sensor Down",

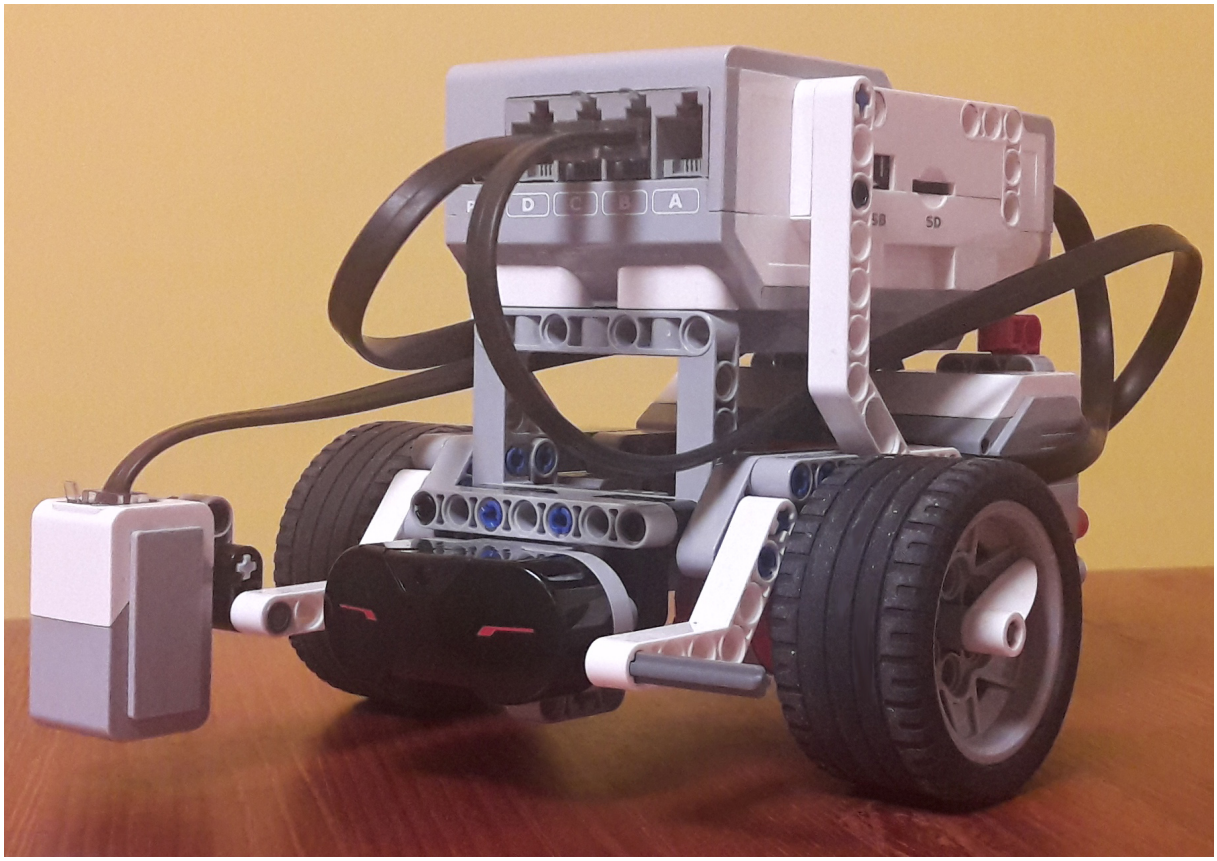
„Driving Base",

„Touch Sensor Driving Base",

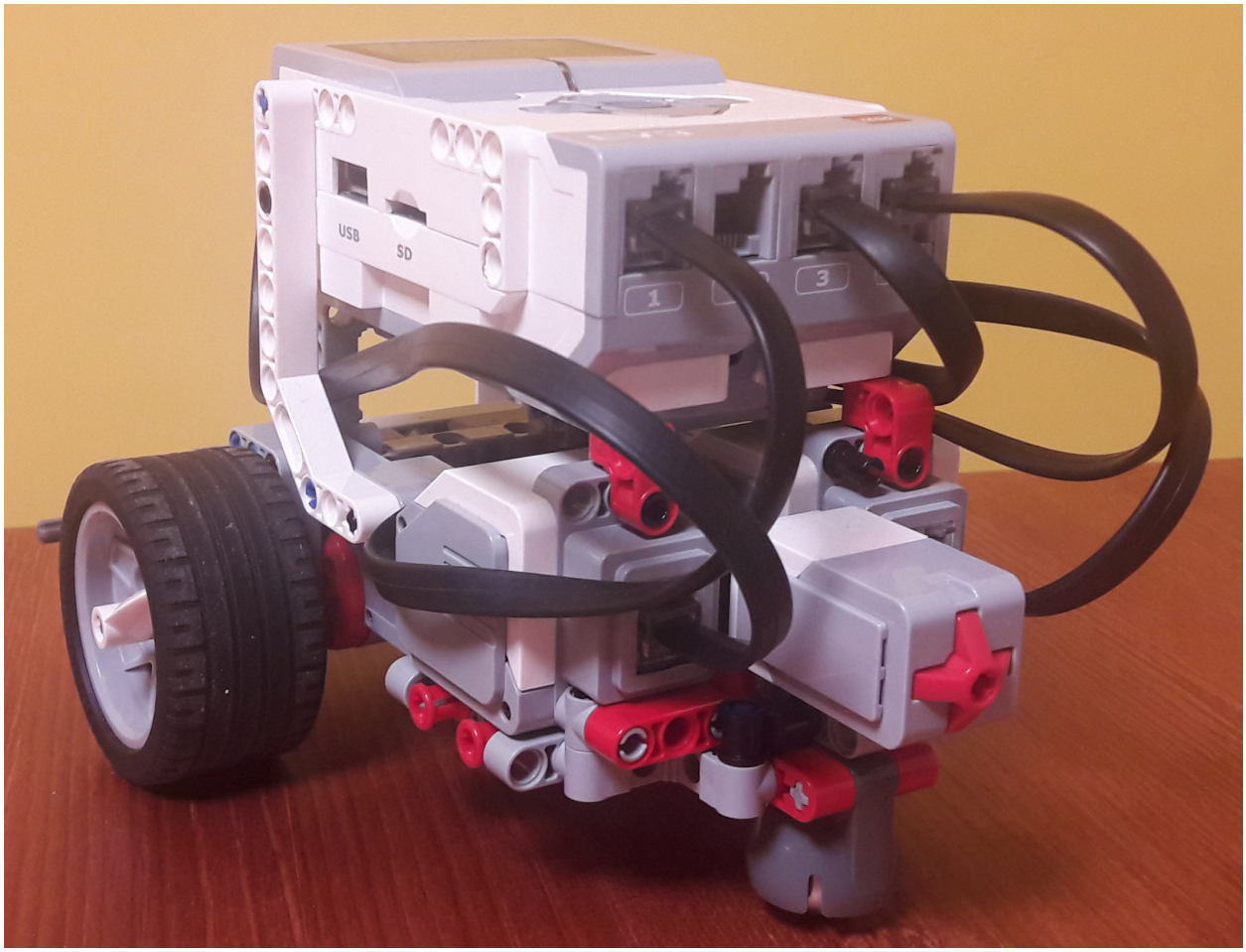
„Ultrasonic Sensor Driving Base".

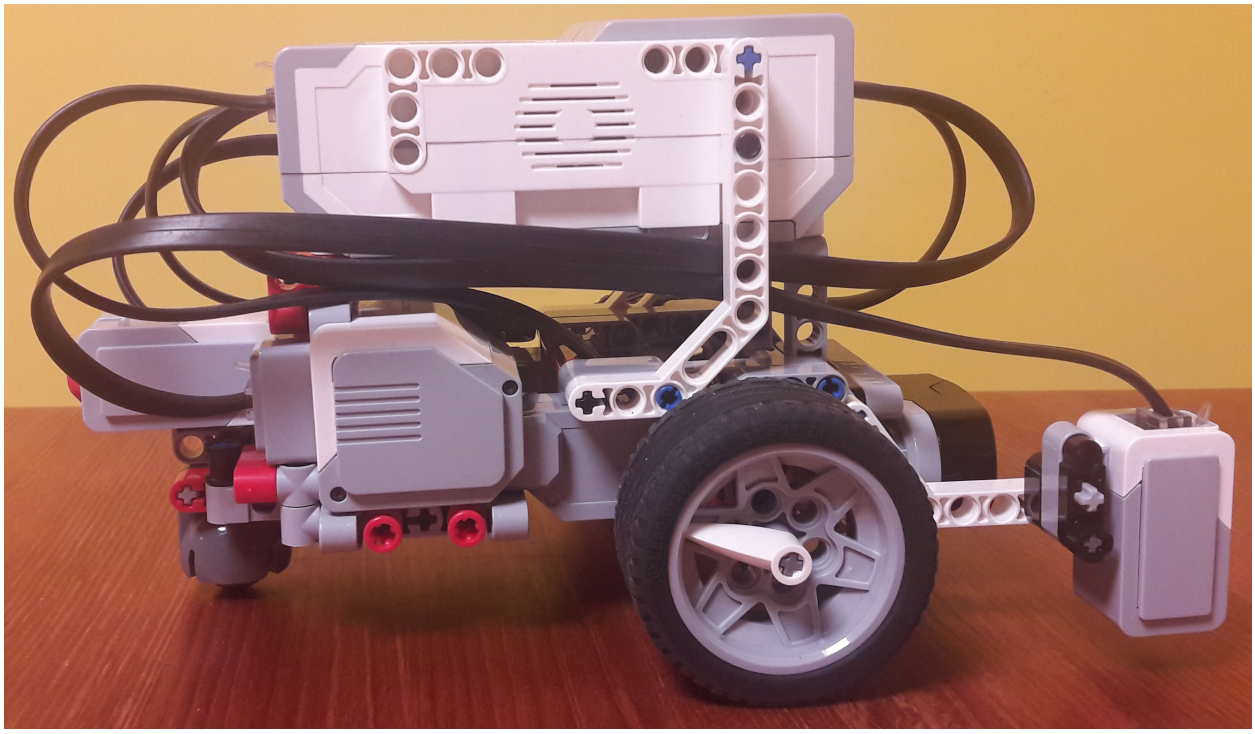
Ha elemekkel használjátok a robotot, akkor érdemes úgy összerakni, hogy könnyen ki lehessen cserélni azokat.
Néhány feladatnál az ütközésérzékelőt hátra kell szerelni, így célszerű már most úgy kialakítani a robot hátulját, hogy könnyen átszerelhető legyen az érzékelő. Sajnos ehhez nem találtunk építési útmutatót, de némi kreativitással könnyen meg lehet oldani a problémát. :) Segítségként mutatok néhány fotót a robotunkról.

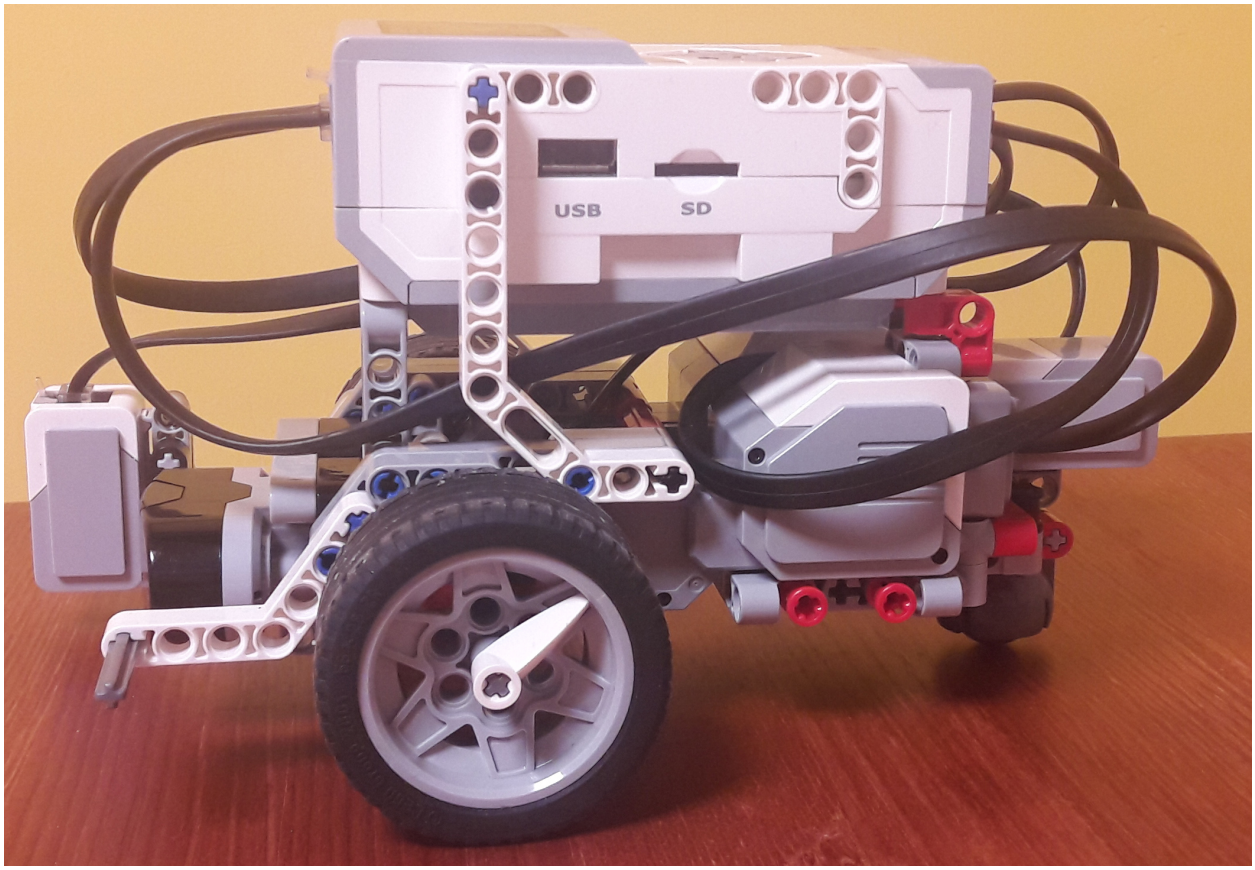
Ezzel természetesen nem azt szeretnénk mondani, hogy más felépítés nem jó, nyugodtan ki lehet próbálni más típusokat is. Nekünk eddig ez bizonyult a legpraktikusabbnak, így ezt szeretnénk megosztani Veletek is.



1. ábra. A robot


USB
SD
1
3




USB
SD

3. Szakköri alkalmak

3.1. 1. alkalom

3.1.1. Csoport megismerése, szabályok kialakítása

Nyári táborokban és szakköri alkalmakon is szoktam ismerkedős játékot játszani. Számomra leginkább a következő vált be.: A játékot mi kezdjük azzal, hogy elmondjuk a keresztnevünket (vagy ahogy szeretnénk, hogy szólítsanak), majd egy jellemző dolgot magunkról.

> *Én általában a hobbit szoktam kérni, mert a gyerekek szünetben, szakkör előtt/után ez alapján könnyebben összebarátkoznak.*

(Pl. X vagyok, szeretek focizni, Y vagyok, imádok olvasni, stb.) A mellettünk ülő megismétli a nevünket és a jellemzőnket (te X vagy, szeretsz ...), majd ő is megmondja a nevét és egy jellemző dolgot magáról. A következő diák már az előző kettőt ismétli sorrendben, majd Ő is „hozzáadja" magát. Így megy végig a játék a csoport összes tagján, majd mi fejezzük be.

> *Itt arra kell figyelni, hogy legalább mi ne hibázzunk. Ehhez jól jön egy kinyomtatott névsor és ugye a kisebb létszám, ami egyébként 10 fő szokott lenni nálunk, maximum. Második órán meg szoktam kérdezni a diákokat, hogy szeretnének-e még egy kört más jellemzővel, és ettől függően játszunk vagy sem. Ilyenkor már a gyerekekre bízom, hogy mi legyen a jellemző, de ha nincs ötletük, akkor én találok ki valamit.*

Ezek után a szabályokat alakítjuk ki.

> *Én a gyerekekkel szoktam összeszedetni a szabályokat, majd kiegészítem őket.*

A szokásos géptermi szabályok itt is érvényesek, de néhányon módosítani szoktam. A szabályok a következők:

- Nem eszünk és iszunk a teremben.

  > *Azonban, ha valaki szomjas, meg szoktam engedni, hogy igyon, de csak kint, a terem előtt (nyitott ajtó mellett).*

- Vigyázunk a gépekre.
- Vigyázunk a robotokra, tesztelni a földön tesztelünk.

  > *Ekkor meg szoktam tippeltetni velük, hogy szerintük mennyibe kerül a robot. Erre azért van szükség, mert tapasztalatom szerint, ha tudják, hogy mennyibe kerül, jobban vigyáznak rá. Ez mondjuk annak is köszönhető, hogy mindig megjegyzem a gyereknek, hogy ha esetleg megsérül a robot, akkor az okozott kárt Nekik, vagyis a szüleiknek kell megtéríteni. Nem a legszebb dolog ez, de az eszközök védelme miatt muszáj.*

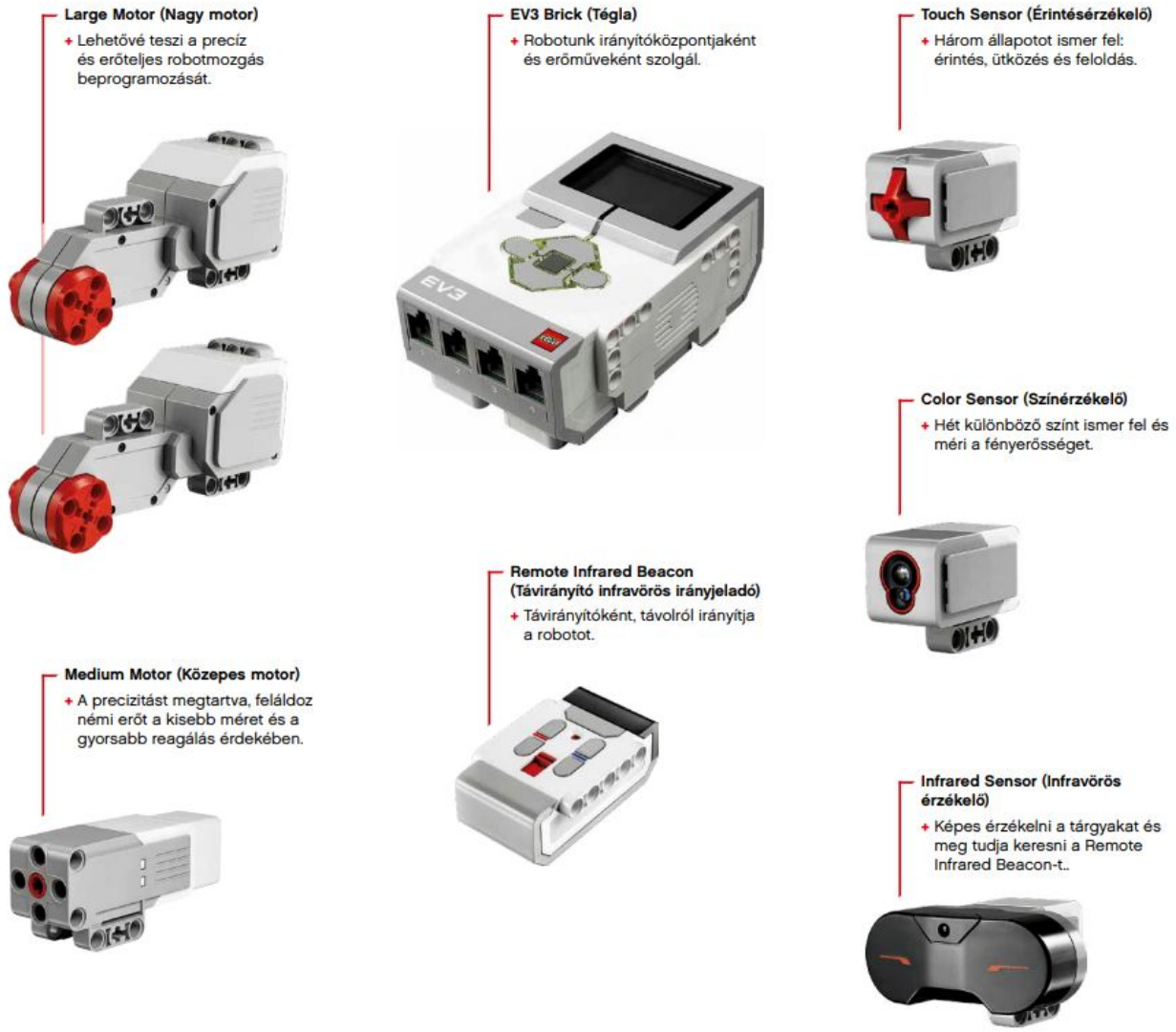
- Egymás számítógépéhez nem nyúlunk, és nem kapcsoljuk ki óra közben. (Sajnos volt rá példa.)

3.1.2. Mi a robot?

Mielőtt nekilátunk a munkának, érdemes beszélgetni arról, hogy tulajdonképpen mi is a robot. Mit tud egy robot? Hol használják őket? Mi a feladatuk? Miért lesznek egyre fontosabbak? Beszéljünk a robotok jövőjéről és az automatizálásról is.

3.1.3. A Lego Mindstorms EV3 bemutatása

A robotok kiosztása után szoktam bemutatni azt, mert így egy kicsit jobban meg tudnak barátkozni vele. Míg én beszélek ők is nézegetik a sajátjukat. Esetleg ha nem egyforma minden robot, akkor az éppen szóba kerülő részét megkeresik a sajátjukon.



2. ábra. A robot részei[3]

Az EV3 „agya”, azaz a Brick (2. ábra), egy Linux alapú mikrovezérlő, ami 64 MB RAM-ot és 16 MB Flash memóriát tartalmaz. Ez azonban, tovább bővíthető 32 GB-ig (a 2.0-s változatban) a beépített micro SDHC kártyahelynek köszönhetően. Az EV3 Brick-et USB kábellel ill. bluetooth-on vagy wifi-n keresztül csatlakoztathatjuk számítógépünkhöz.

[3] `http://bit.ly/ev3_hasznalati_utmutato`

A tégla kapott egy kis képernyőt is, amire ugyan fekete-fehérben, de lehet információkat kiírni. Ez alatt, bal oldalon található egy szürke gomb, ami a programok leállítására, valamint a robot kikapcsolására szolgál. Felmerül a kérdés, hogy mivel kapcsoljuk be? A képernyő alatt lévő gombok közül, a középső gomb kicsit hosszabb megnyomásával kelthetjük életre kis robotunkat.

> *A be- és kikapcsoló gombok „bemutatását" hagyhatjuk későbbre is, mert amint megtudják a diákok, hogy mivel kell bekapcsolni a robotot, elvesztettük őket minimum tíz percre.*

A tégla alján található az akkumulátor vagy az elem foglalatok.
A tégla két (rövidebbik) oldalán portok vannak, az egyik oldalon számmal, míg a másik oldalon betűvel jelölve. A betűvel jelöltekbe csatlakoztatjuk a motorokat, míg a számmal jelöltekbe az érzékelőket.

> *Az érzékelők csatlakoztatását érdemes kétszer leellenőrizni hiba esetén, ugyanis gyakran előfordul, hogy más portba van csatlakoztatva, és más port van a programba írva.*

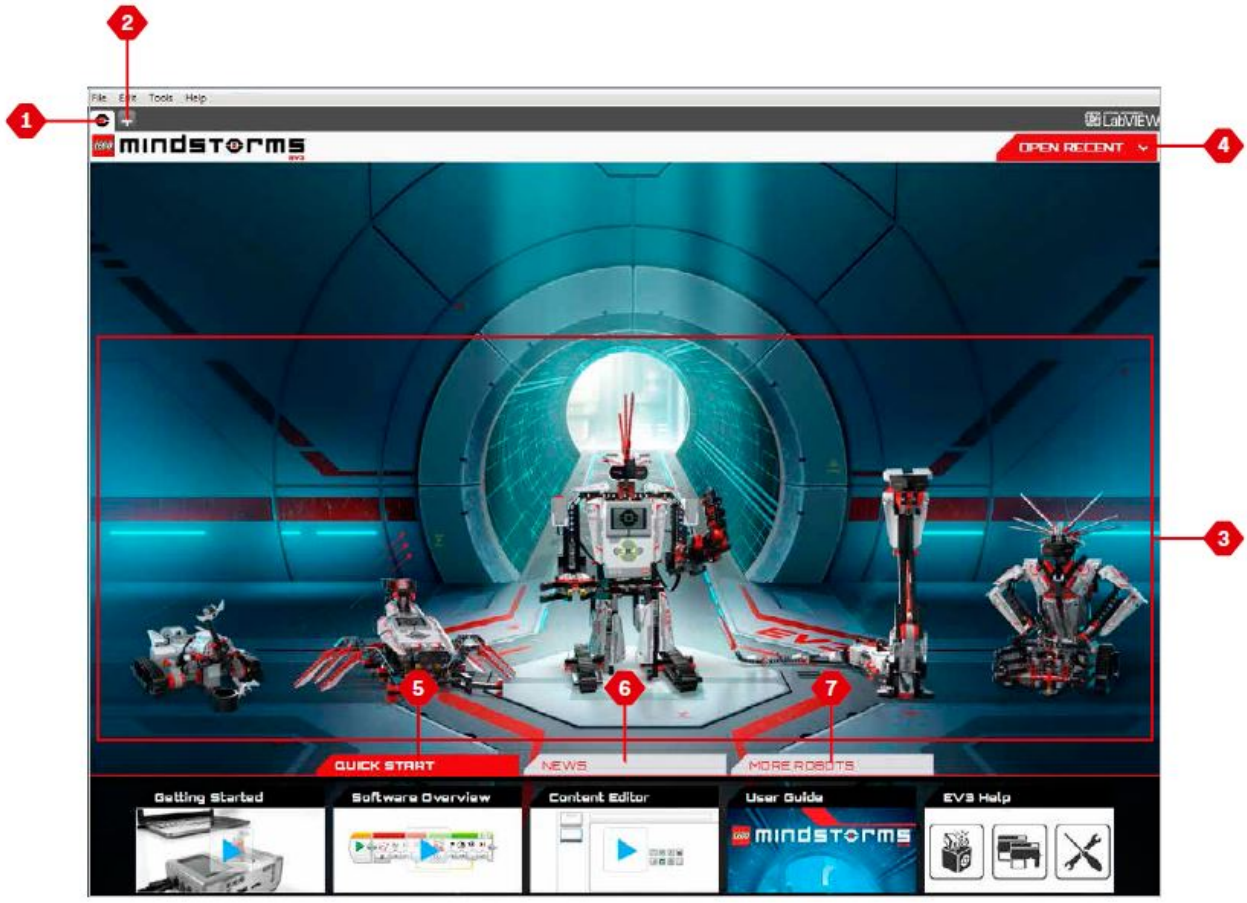
A szettben két nagy motor, és egy közepes motor található, azonban mivel a mi építményünkben csak nagy motorok vannak, így a továbbiakban motor címszó alatt, ezekre fogok utalni. További tartozékok az érzékelők, pontosabban a szín-, az érintés-, és az infravörös érzékelő. Ezek használatáról az egyes szakköri alkalmaknál fogok írni.

A 2. ábrán a *Retail Core Set (31313)* elemei láthatóak. Ebben a dobozban nincs gyro és ultrasonic (ultrahang) szenzor (3. ábra), viszont az *Education Core Set (45544)* már tartalmazza őket. A szakkör során a gyro szenzorra nem, de az ultrasonic szenzorra ki fogunk térni, mert működés szempontjából az utóbbi hasonlít az infrared szenzorra, valamint mindkettő (infrared és ultrasonic) úgy néz ki, mintha a szemei lennének a robotnak (ami részben igaz is).

3. ábra. A gyro és az ultrasonic szenzor

3.1.4. A programozási környezet bemutatása

Igaz, hogy a robotok építése is öröm, de a robotika lényege ezek életre keltése. Ehhez egy ikon-alapú programozói felületet fogunk használni, a LEGO© MINDSTORMS© EV3 Software-t. A programot a következő honlapról lehet letölteni: `www.lego.com/hu-hu/mindstorms/downloads`. Már elérhető tabletre is a szoftver, azonban a gépi verzióhoz képest kevesebbet tud, nem tartalmaz minden blokkot, így a szakkörön a gépi verziót használjuk.

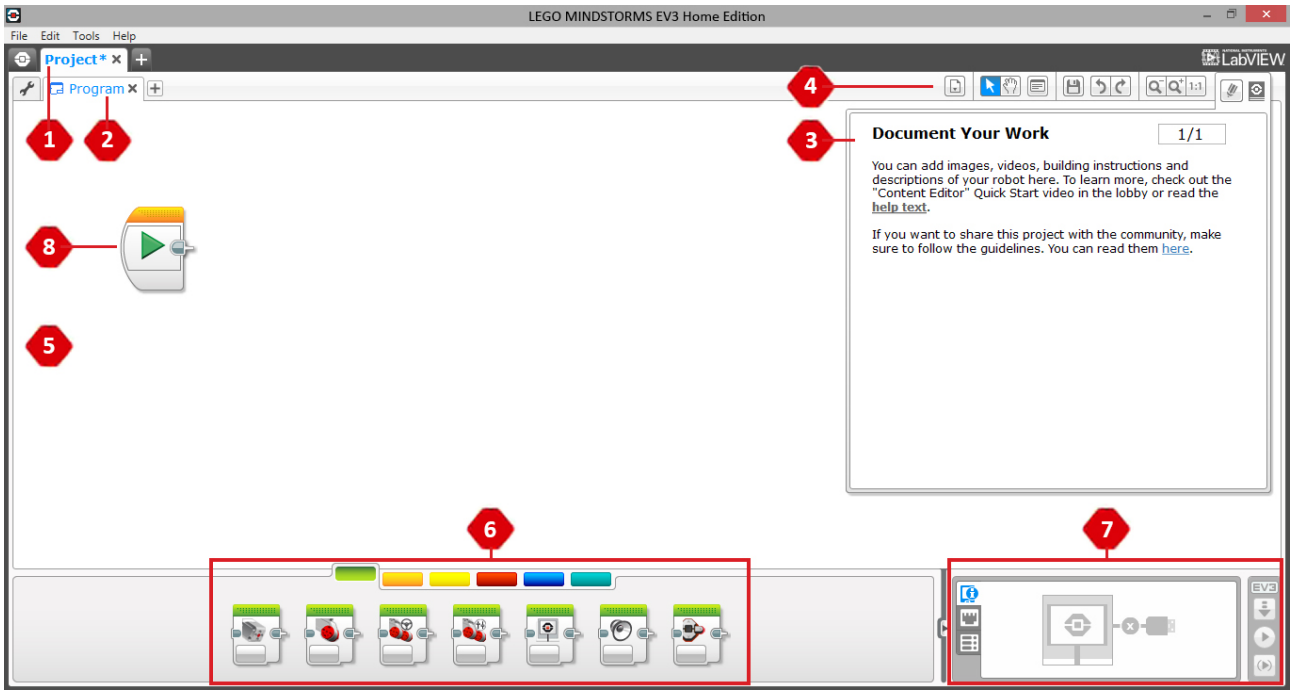


4. ábra. A lobby

A programot megnyitva a lobby-ba, az előszobába jutunk, ez látható a 4. ábrán. A lobby részei a következők:

1. Lobby - Ez a gomb visszavisz a Lobby-ba.
2. Add Project - Új projektet nyithatunk vele.
3. Robot Missions - Ezek a robot küldetések. Egy robotra kattintva az adott robot építési útmutatóját érjük el, ill. további információkat kaphatunk a választott építményről.
4. Open Recent - Legutóbbi programok megnyitása
5. Quick Start - Első lépések: Támogató források, például rövid, bevezető videók, EV3 User Guide (Felhasználói útmutató), és Software Help (Szoftver Súgó).
6. News — Hírek: Rövid történetek és hírek a `www.lego.com/mindstroms` honlapról. Újabb verziókban (1.2.2-től) More Robots.
7. More Robots - További robotok (Hozzáférés további modellek építéséhez és programozásához).Újabb verziókban (1.2.2-től) Robot Remix.

Hogy őszinte legyek, ezt a Lobby részét nem nagyon szoktam használni a szoftvernek, mert elvonja a gyerekek figyelmét a programozástól. Ezért mindig csak nyitunk egy új projektet, és fejest ugrunk a programozásba.



5. ábra. A programozói felület

Az 5. ábrán látható programozói felület részei a következők:

1. Projekt címe
2. Program címe
3. Dokumentáció készítő
4. Gyorsgombok
5. Programozói vászon
6. Program blokkok
7. Hardveroldal (Hardware Page)
8. Program kezdőpontja

A Dokumentáció készítő néha elég zavaró tud lenni, de be lehet zárni, mégpedig a Gyorsgombok közül az utolsóval. (Aminek olyan az ikonja mint a Mindstormsnak.)

A „Hardveroldal” résznél az 5. ábrán az az állapot látható, amikor nincs a géphez csatlakoztatva robot vagy nincs bekapcsolva a robot.
Ha bekapcsoljuk a robotot, akkor mind a három fül, megtelik információval. Az első fül a Tégla információs fül, ami fontos információkat jelenít meg a csatlakoztatott robotról. Ezek a következők: a Tégla neve, töltöttségi szintje, a firmware verzió, a csatlakozás típusa és a memóriasávja. Itt tudunk hozzáférni a Brick belső memóriájához és itt található a Wireless Setup (Vezeték nélküli beállítás) is. A második fül a Port nézet fül, ami a Téglához csatlakoztatott érzékelőkről és motorokról ad információkat. Ha csatlakoztatva van a robot a számítógépünkhöz, akkor az itt megjelenő értékek a valós érzékelt adatok. Az utolsó a Felhasználható Téglák fül, ami azokat a Brick-eket mutatja, amelyekhez éppen lehet csatlakozni.

Wifi és bluetooth kapcsolat esetén figyeljünk arra, hogy mindenki a saját robotjára csatlakozzon, mert ennek akár a robot is láthatja kárát.

Ugyanezen panel jobb oldalán három gomb látható (6. ábra), nevezzük ezeket vezérlőknek. Ezek funkciói a következők:

1. Letöltés - A program letöltése a Brick-re.
2. Letöltés és futtatás - Letöltés után a program azonnal elindul. (A funkció használata bluetooth vagy wifi kapcsolat esetén ajánlott csak.)
3. Letöltés és kiválasztott futtatás - Csak a kijelölt blokkokat tölti le a Téglára és azonnal futtatja azokat.

A következő részt a kivetítőnél magyarázva szoktam elmondani a gyerekeknek. Közben már ők is belépnek a programba és velem együtt nézik a blokkokat.

6. ábra. Hardveroldal vezérlője

Amit még részletezni kell egy kicsit, azok a Program blokkok. A zöld színű az Action Blocks, azaz a Cselekvő blokkok. Itt többnyire a mozgatáshoz kapcsolódó részek találhatók. A narancssárga a Flow Blocks-okat (Folyamat blokkokat), azaz a vezérlési szerkezeteket tartalmazza, míg a citromsárga az érzékelőkkel kapcsolatos blokkokat. A piros a Data Blocks, ami magyarul az adatblokkokat jelenti, de valójában ezek a változókkal kapcsolatos dobozok. A sötétkék az Advanced Blocks, amit a készítők Speciálisként jelöltek meg, és végül a világoskék, ami a My Blocks, a saját blokkjaim.

Mielőtt még nekilátunk a programozásnak, azt meg kell említenem, hogy ha a programot a 6. ábra 2-es számú gombjával indítjuk el (és kábel esetén nem húzzuk ki a kábelt), akkor nyomon követhető a program. Indítás után a blokkok színes felső része, mintha elkezdene pörögni vagy vibrálni. Nos, ekkor a robot, éppen ezt a blokkot hajtja végre.

3.1.5. Mozogjunk!

Na, ennyi elmélet után próbáljuk ki végre a robotokat! Először vegyük rá őket arra, hogy mozogjanak!

Ráhangoló kérdések

1. Mivel tudjuk őket mozgatni?
2. Milyen blokkokkal mozgatjuk a motorokat?
3. Lehet egyszerre két motort mozgatni?
4. Mi szabályozhatja a motor működését?

Válaszok

1. Motorral.
2. Zöldekkel vagy Large Motor, Move Steering, Move Tank blokkokkal.
3. Igen, lehet, az utolsó kettővel az előbbi felsorolásban.
4. (Ha őszinték akarunk lenni elég sok minden, de itt:) idő, elfordulási szög, fordulatok száma.

Három blokk van ami ezeket tudja, és ezek a következők:

7. ábra. Motorblokkok (balról jobbra Large Motor, Move Steering, Move Tank)

Move Steering

Mielőtt továbbmennénk tisztáznunk kell, hogy a Move Steering hogyan működik. Azt már tudjuk, hogy a két motort egyszerre mozgatja, már csak az a kérdés, hogy hogyan. Nos, a válasz nem olyan egyszerű mint gondolnánk. Azt már látjuk, hogy egyetlen sebesség értéket kell megadnunk, aminek az az oka, hogy ezt az értéket a robot elosztja a két motor között. Tulajdonképpen ezt az elosztást határozza meg a Steering paraméter, mert amelyik irányba húzzuk el az adott csúszkát, az a motor lesz gyorsabb, és a robot ennek megfelelően fordul el. (Kattints a programozói felületen az első paraméter értékére és megjelenik a csúszka.) A fordulási szög ettől az értéktől függ.

> *Sajnos ennek a blokknak a működése nem egyszerű, de talán próbálgatással a gyerekek jobban megértik, így célszerű ezt a későbbiekben alaposan letesztelni.*

Motorok mozgatása

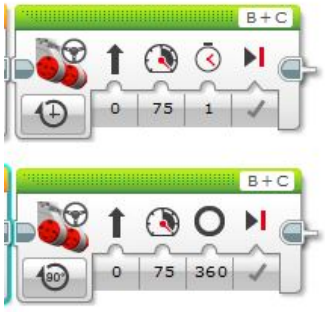
Térjünk vissza a 7. ábrához! Mindhárom opciónál a bal alsó sarokban lévő (mód) menüt lenyitva a következő lehetőségek közül választhatunk:

- Off - Kikapcsolva
- On - Bekapcsolva
- On for Seconds - Bekapcsolva adott másodpercig
- On for Degrees - Bekapcsolva adott tengelyfordulási szögig
- On for Rotations - Bekapcsolva adott tengely fordulatszámig

Az első opció szerintem egyértelmű, a másodiknál csak a sebességet kell megadnunk, ami a Large Motor (nagy motor) és a Move Steering („Kormányvezérelt motor blokk”, becenevén kormányzós mozgás) esetében egyszer szükséges, míg a Move Tank („Sebességvezérelt motor blokk”, becenevén tankos mozgás) esetében kétszer. A motor sebessége egy -100 és +100 közötti szám lehet, amit akár kézzel is beírhatunk, vagy akár a csúszkát húzogatva is beállíthatunk.

Mindig elmondom, hogy annak ellenére, hogy ilyen nagy értéket is beírhatunk, ez az érték inkább -50 és +50 között legyen. Ennek az az oka, hogy ennél nagyobb értéknél nagyon pörög a motor, ami egy falnak ütközésnél nem biztos, hogy jót tesz a robotnak vagy a motornak.

Az On for Seconds opciót választva megjelenik a blokkban egy stopperórával jelzett ikon, ami az idő (másodperc) paraméter (8. ábra felső blokkja). Itt azt állíthatjuk be, hogy hány másodpercig működjön a motor, ami akár tizedes tört is lehet. Ebben az esetben viszont figyeljünk arra, hogy nem tizedes vessző kell, hanem pont. A stopperórával együtt egy másik ikon is megjelenik, ezt „Brake at End"-nek nevezik, ami annyit tesz, hogy hogyan álljon meg a motor. Ha a pipát választjuk, akkor a motor a program végén rögtön megáll, míg ha az x-t, akkor fokozatosan lassul le.



8. ábra. A Move Steering blokk On for Seconds és On for Degress módjai alapbeállításban.

Általában nagy különbség nincs a két mód között, ennek ellenére érdemes kipróbálni. Én legtöbbször az alapbeállítást használom.

A következő lehetőség az On for Degrees (8. ábra alsó blokkja). Ez az az opció, ami rendesen össze tudja zavarni a gyerekeket. Ami megjelenik 360°, az nem a matematikai teljes szög, hanem a tengely fordulási szöge. Ez az érték egyébként lehet nagyobb is, mint 360°, és akár tizedes tört is megadható. A működési elv hasonló az előzőhöz, a megadott tengelyfordulási szögig fordul a motor, utána leáll.

Végül, az On for Rotations, a teljes tengelyfordulások száma következik. Annak ellenére, hogy „teljes" tengelyfordulás, lehet megadni tizedes törteteket is.

Hasznos információk

- A Medium Motor ugyanilyen opciókkal rendelkezik.
- A Move Tank esetén a két motor külön szabályozható, és így finomabban hangolható a mozgása.
- A működési módok közül az utolsó három addig működik, amíg a beállított érték nem teljesül. Ezután áll csak le, és közben más utasítás nem hajtódik végre. Ha mozgás közben szeretnénk más parancsokat is végrehajtatni (pl. szenzorokat figyelni), akkor az On módot kell választanunk. Azonban ebben az esetben, a motor leállításáról külön blokkal kell gondoskodnunk.

3.1.6. Feladatok

1. Készíts programot, amelyben a robot előre halad! Próbáld ki a kormányzós és a tankos blokkokat is több variációban!
2. Készíts programot, amelyben a robot hátra halad! Próbáld ki a kormányzós és a tankos blokkokat is több variációban!
3. Írj programot, amelyben a robot előre megy 1000°-os tengelyfordulásig!
4. Készíts programot, amelyben a robot derékszögben elfordul!

5. Írj programot, amelyet végrehajtva a robot körbe forog (helyben forog)!

6. Írj programot, amelyet végrehajtva a robot leír egy négyzetet, háromszöget, kört!

3.2. 2. alkalom

Ráhangolódás

Az órát kezdhetjük egy, a gyerekek (és felnőttek) által kedvelt „teszttel”, egy Kahoot-tal.

- *Mi a robot teljes neve?*
  - *LEGO robot*
  - *EV3 készlet*
  - *LEGO Mindstorms EV3*
  - *LEGO Mindset EV3*
- *Milyen fajta motorok találhatók a robotunkon jelenleg?*
  - *kicsi és közepes*
  - *közepes és nagy*
  - *közepes*
  - *nagy*
- *Hány érzékelő van a robothoz csatlakoztatva?*
  - *1*
  - *2*
  - *3*
  - *4*
- *A betűvel jelölt portokhoz mit kell csatlakoztatni?*
  - *motorokat*
  - *érzékelőket*
- *Igaz-e, hogy ha a tengelyfordulatot 90-re állítjuk, akkor a robotunk 90°-ot fog fordulni?*
  - *igaz*
  - *hamis*
- *Melyik motorblokkot NEM érdemes használni, ha azt szeretnénk, hogy robotunk egyenesen menjen?*
  - *Kormánymotor*
  - *Tankmotor*
  - *Nagymotor*

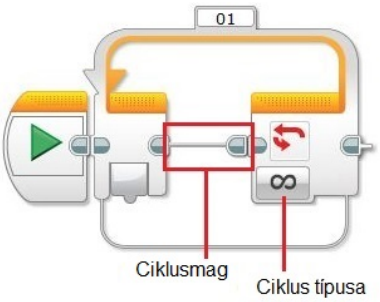
Mit csináltunk eddig?

- Szeretnétek-e megint ismerkedős játékot játszani?
- Mit csináltunk múlt héten, ki emlékszik rá?
- Milyen alakzatokat rajzoltattunk ki a robottal?

A múlt órai feladatok közül kivetíthetjük Nekik néhánynak a megoldását, majd megkérdezhetjük, hogy nem lehetne-e ezt a kódot lerövidíteni, vagy nincs-e benne véletlenül ismétlődés? Nyilván, érdemes valamilyen egyszerűbb kódot választani, például a négyzetét vagy a háromszögét. Ha így nem veszik észre az ismétlődéseket, akkor akár le is rajzolhatjuk nyilakkal, hogy mit csinál a robot. (Pl. a négyzet így nézne ki: ↑ → ↑ → ↑ → ↑ →, ahol a ↑ előre haladást, míg a → jobbra fordulást jelent.)

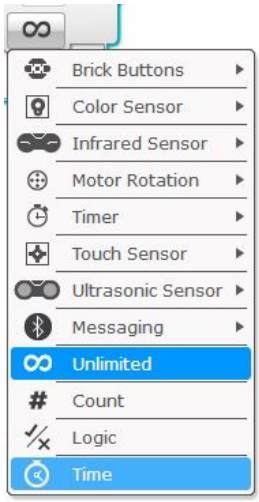
3.2.1. Ciklus bemutatása

Ha a robot egy programrészt többször megismétel, akkor ezt a részt belerakhatjuk egy ciklusba (loop-ba). Fontos, hogy tényleg csak az ismétlendő részt tegyük a ciklus pocakjába (a ciklusmagba), mert különben nem ugyanaz lesz az eredmény.



9. ábra. A ciklus

A végtelen jelre kattintva egy hosszú lista nyílik le, amit a 10. ábra is mutat. Ezek a ciklus fajtái. (A lista eltérő tartalmú és hosszúságú lehet attól függően, hogy milyen blokkok vannak pluszba importálva a gépre. A 10. képen az Ultrasonic Sensor van importálva, a többi alapértelmezetten a program része.)
A bemutatást alulról kezdeném, így a lista utolsó eleme a „Time". Ez azt jelenti, hogy a ciklusmag adott másodpercig fog ismétlődni. Ez előtt van a „Logic", ami a logikai feltételt jelenti, de erről sajnos most nem lesz szó.



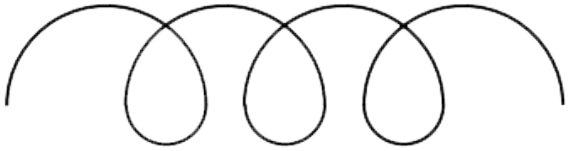
10. ábra. A ciklus típusai

A következő (azaz az előző) a „Count", ami a mi esetünkben darabszámot fog jelenteni, tehát az általunk megadott mennyiségszer fog lefutni a ciklusmag (vagy a blokk pocakja). Ezt megelőzi az „Unlimited", a végtelen, amit szerintem nem kell nagyon magyarázni.
Amik ezek felett helyezkednek el, szinte mind egy szenzortól vagy egy külső hatástól (mozgástól, gomb benyomásától) függnek. A felső blokkból kiemelném még az első lehetőséget, a „Brick Buttons"-t. Ha ezt a funkciót választjuk, akkor a ciklusmag általunk kiválasztott gomb (vagy gombok) megnyomásáig ismétlődik.

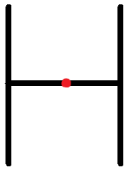
3.2.2. Feladatok

1. Készítsd el a múlt órai négyzet és háromszög programját ciklussal!
2. Írj programot, amelyet végrehajtva a robot leír egy téglalapot, de a programod legyen minél rövidebb!

3. Készíts programot ciklus használatával, amelyet végrehajtva a robot az alábbi alakzatot írja le mozgása során!

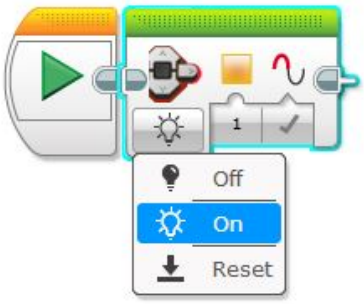


4. A robot mozgása során írjon le egy H betűt a piros pontból indulva!



3.2.3. Színes led használata a roboton

A Tégla a gombjait megvilágító ledet (Brick Status Light) is programozhatjuk. Erre az „Action” fül utolsó blokkját kell használnunk.



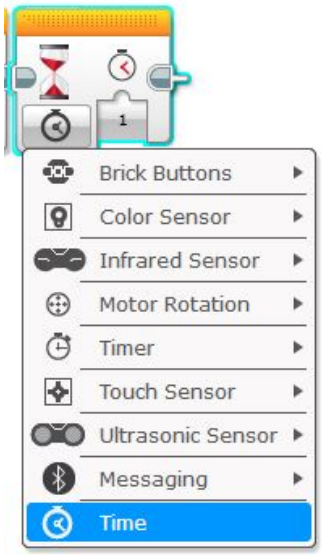
11. ábra. A Brick Status Light blokk módjai

A blokkot a vászonra áthúzva megjelenik egy lámpa ikon, amit ha megnyitunk, három opció közül választhatunk. Ezt láthatjuk a 11. ábrán. Tehát a lehetőségek a következők: Off, On és Reset. Az Off nyilván, kikapcsolja a ledet. Az On bekapcsolja, és ezt választva a 11. ábrán is látható két paraméter jelenik meg. Az első paraméterrel a led színét tudjuk kiválasztani, míg a másodikkal azt tudjuk eldönteni, hogy a led villogjon vagy ne. Az utolsó, Reset, lehetőséggel olyan villogást tudunk produkálni, mint amit egy program futtatásakor szokott a Tégla.
Most, hogy már mindent tudunk a blokkról, próbáljuk is ki a 11. ábrán látható programot! Ne ijedj meg, ha elsőre nem működik. Ez egy olyan blokk, amit „önmagában” nem tudunk használni. Mivel kipróbáltátok a programot, így gondolom észrevettétek, hogy lefut a program, de a Téglán nem történik semmi. Ennek az az oka, hogy nem adtuk meg, hogy meddig fusson a program. Szerintetek, hogyan lehetne megadni azt, hogy meddig villogjon? (Itt akár olyat is meg lehet adni, hogy egy tengelyfordulásig fusson a program, de a fő cél az lenne, hogy az időnél lyukadjunk ki, ugyanis ezzel függ össze a következő parancs, amivel meg fogunk ismerkedni.)

3.2.4. Várj! avagy ismerkedés a várj blokkal

Nos, mint ahogy a neve is utal rá, a blokkal azt tudjuk elérni, hogy a program bizonyos ideig vagy egy szenzor érzékeléséig várakozzon. Ezt nem úgy kell elképzelni, hogy akkor addig nem történik semmi, hanem inkább pont az ellenkezője. A program a várj blokk előtt lévő elemet addig futtatja, amíg le nem telik az idő vagy meg nem kapja a szükséges információt a szenzortól.

A blokkot a narancssárga (Flow Control) parancsok között találjuk meg, és a 12. ábrán látható módon néz ki. Alapértelmezett beállítása az idő, amiben az egyetlen paramétert másodpercben kell megadni. A bal alsó sarokban lévő óra ikonra kattintva a cikluséhoz hasonló ablak nyílik le, amit a 12. ábra is mutat. (Most csak az idővel fogunk foglalkozunk, így a többiről később lesz szó.)



12. ábra. A Wait (várj) blokk és opciói

3.2.5. Feladatok

1. Készíts programot, melyben a ledek sárgán villognak 2 másodpercig!
2. Készíts programot, melyben a ledek pirosan villognak 1 másodpercenként!
3. Készíts programot, melyben a robot vár 2 másodpercet, majd leír egy kört!
4. Készíts programot, melyben a robot pirosan villogtatja a lámpáját miközben előre megy egy tengelyfordulást!
5. Készíts programot, melyben a robot leír egy négyzetet miközben pirosan világít a lámpája!
6. Készíts programot, melyben a led színe 1 másodpercenként változik! (Led villogtatása)
7. Írj programot, amelyet végrehajtva a robot helyben forog, majd várakozik 3 mp-ig, ezután előre megy 1000°-os tengelyfordulásig, majd ismét forog 5 mp-ig!

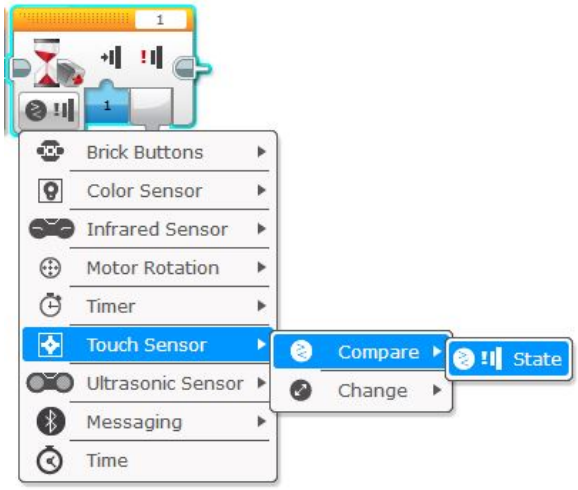
3.3. 3. alkalom

Ráhangolódás

- *Mire jó a ciklus?*
  - *Csak így lehet a robottal megismételtetni a mozgásokat.*
  - *Az ismétlődéseket meg tudjuk adni vele rövidebb formában.*
  - *Mindig mindent végtelenszer meg tud ismételni vele.*
  - *Csak így lehet a motort működtetni.*
- *A számláló ciklus értékét mennyire állítanád, ha a robotodnak egy háromszöget kell kirajzolnia?*
  - *1*
  - *2*
  - *3*
  - *végtelen*
- *Tud kéken világítani a roboton lévő led?*
  - *Igen*
  - *Nem*
- *Ha azt szeretnénk, hogy folyamatosan változzon a roboton a led színe, akkor …*
  - *végtelen ciklusba kell rakni a programunkat.*
  - *végtelenszer le kell írni a programba ugyanazt.*
  - *a program magától megismétli amit egyszer leírtunk.*
  - *a ciklus számlálóját egyre kell állítani.*
- *Milyen színű blokkok között találjuk a szoftverben a homokórát?*
  - *kék*
  - *piros*
  - *zöld*
  - *narancs*
- *Ahhoz, hogy robotunk tolasson, …*
  - *negatív tengelyfordulat értéket kell megadnunk.*
  - *meg kell fordítanunk rajta a motorokat.*
  - *negatív sebességet kell megadnunk.*
  - *a két motor sebességét ellentétes előjellel kell megadnunk.*

3.3.1. Érintésérzékelő használata

A mai órán megismerkedünk egy érzékelővel, mégpedig az érintésérzékelővel. Az érzékelőket a várj blokkal vezetjük be, mert ugyanúgy tudjuk használni a szenzort ezzel, mint a citromsárga blokkal, de itt nem kell foglalkoznunk a paraméterátadással (annak ellenére, hogy megjelenik ez a funkció).



13. ábra. A nyomásérzékelő beállítása

A várj blokkot a 13. ábrán látható módon fogjuk használni a legtöbb esetben, de mielőtt áttérnénk a használatra, kell néhány szót ejtenem arról, hogy miért is ezt használjuk. Amint látható a 13. ábrán, a „Compare" és „Change" lehetőségek közül választhatunk, amik összehasonlítást és változást jelentenek. Ha a Compare-t választjuk, akkor a robot (és a program) addig fog várakozni, amíg a nyomásérzékelő a kiválasztott állapotba nem kerül. Ezzel szemben, Change módban a robot addig várakozik, amíg változás nem áll be az állapotában. Azaz, ha a program kezdetekor kiengedett állapotban volt a szenzor, akkor lenyomásig várakozik, és ha benyomott állapotban volt induláskor, akkor pedig felengedésig várakozik.

A két mód közötti különbséget meg lehet éppen említeni a diákoknak, de nem feltétlenül szükséges, akár később is visszatérhetünk rá. A többi szenzort is ezen a módon fogjuk használni, így érdemes ezt megjegyezni.

Az ütközésérzékelő (Touch szenzor) három módba állítható, ezeket láthatjuk a 14. ábrán, jelentésük pedig a következő:

A 0 típusú a *Released* nevet kapta, ami annyit jelent, hogy az érzékelő kiengedett állapotban van.
Az 1-es a *Pressed*, azaz a benyomott állapot.
A 2-es pedig a *Bumped*, ami a gomb felengedését jelenti.

A működési módok közti különbségeket az 1. és a 2. feladat szemlélteti a legjobban.

14. ábra. A nyomásérzékelő módjai

3.3.2. Feladatok

1. Készíts programot, melyben a robot ledje reset módban villog addig, amíg be nem nyomódik a gombja vagy el nem engedjük a gombot! Ha ez megtörtént, akkor pirosan villogjon 2 másodpercig a lámpája. (Nézd meg, hogy mi történik a kóddal futtatás közben, ha nyomva tartod a gombot, ill. ha csak megnyomod és elengeded!)

2. Készíts programot, melyben a robot ledje reset módban villog addig, amíg nem történik változás az ütközésérzékelő állapotában! Ha megváltozott az állapota, akkor pirosan villogjon 3 másod-

percig a lámpája. (Nézd meg, hogy mi történik a kóddal futtatás közben, ha nyomva tartod a gombot, ill. ha csak megnyomod és elengeded!)

3. Írj programot, amelyet végrehajtva a robot elindul egyenesen, ha benyomódik (Bumped) az érintésszenzor!

4. Készíts programot, melyben a robot 3 másodpercig egyenesen megy az ütközésérzékelő megnyomására!

5. Írj programot, amelyet végrehajtva a robot elindul egyenesen, ha benyomódik (Bumped) az érintésszenzor és egyenesen megy mindaddig, amíg akadálynak nem ütközik! Ekkor álljon meg a robot!

6. Készíts programot, melyben ha benyomódik az ütközésérzékelő, akkor a robot hátramegy két fordulatot! (Menekülő robot)

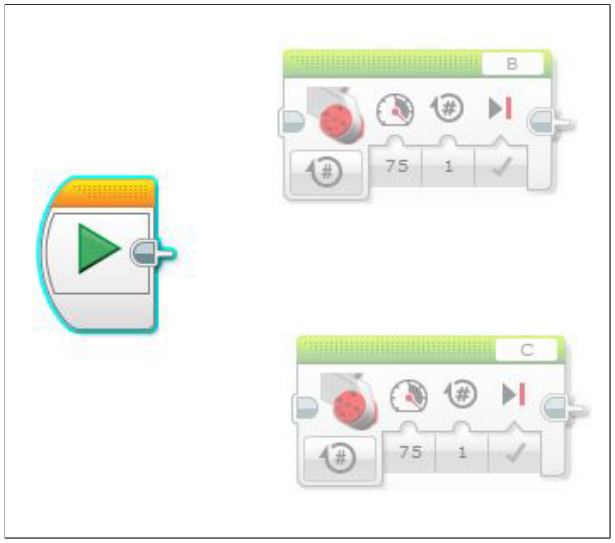
3.3.3. Párhuzamos programok készítése

Nem tudom, hogy figyeltétek-e, de eddig minden programunk úgy készült, hogy egymás mellé pakolgattuk a blokkokat és a robot ilyen sorrendben hajtotta végre őket. Azonban a robotunk ennél sokkal okosabb, egyszerre több dologra is tud figyelni (pl. majd később több szenzorra) és több dolgot tud egyszerre csinálni. Például tud egyszerre menni, villogtatni a lámpáját és figyelni a nyomásérzékelőt. Ha ilyen feladatokat oldunk meg a robotunkkal, akkor párhuzamos programokat készítünk, melyben a feladatokat a program szálainak nevezzük. Ez alapján csoportosíthatjuk a programjainkat, hogy hány szálúak. Az eddigiek egyszálúak voltak, amiket pedig most fogunk elkészíteni többszálúak lesznek.

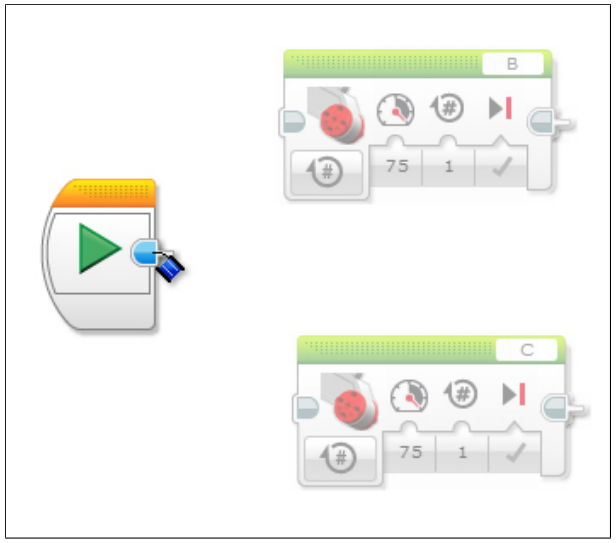
Párhuzamos programok létrehozása a programozó felületen

A párhuzamos programok kialakítása elsőre nem biztos, hogy egyértelmű, ezért készítettem egy kis útmutatót, ami a 15-19. ábrákon látható.

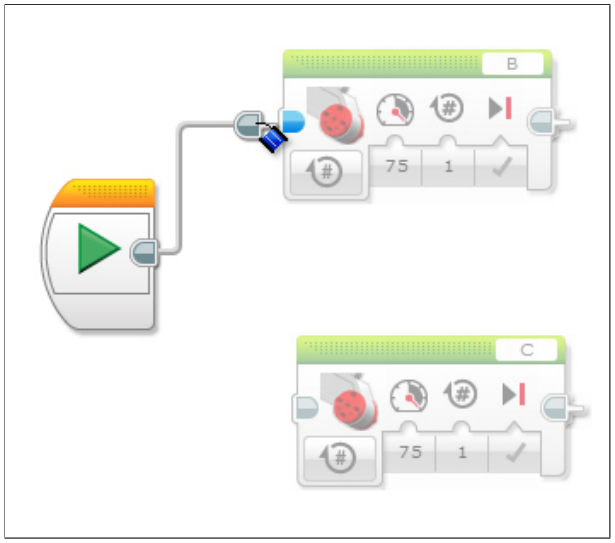
Először elhelyezzük a blokkokat amiket szeretnénk párhuzamosítani (15. ábra). Ezután az indító blokk jobb oldalán lévő szürke félkör szerűség fölé visszük az egerünket, majd ha megjelenik a 16. képen is látható kék alakzat, akkor a bal egérgomb lenyomása közben áthúzzuk az egerünket egy másik blokk ugyanilyen szürke részéhez (16-17. ábrák). Ha a jobb oldali blokkon is kékké válik a szürke félkör, akkor elengedjük az egeret és készen is vagyunk a feladat felével (18. ábra). Ugyanezt el kell végezni a másik blokkal is, hogy teljes legyen a párhuzam. Ha ezzel kész vagyunk, akkor sikeresen összeraktunk egy párhuzamos programot (19. ábra).



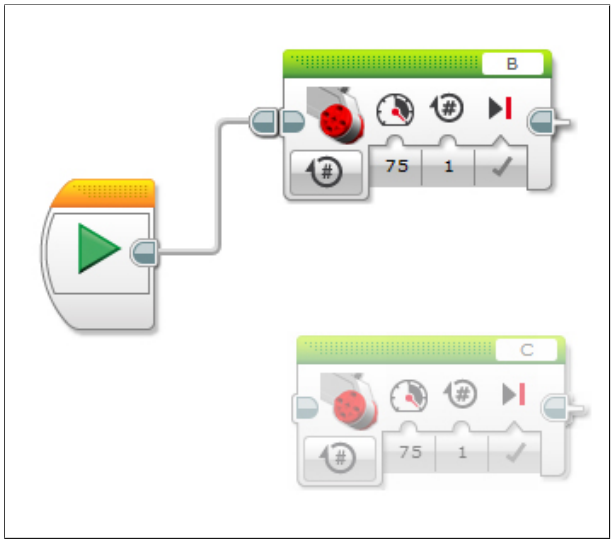
15. ábra



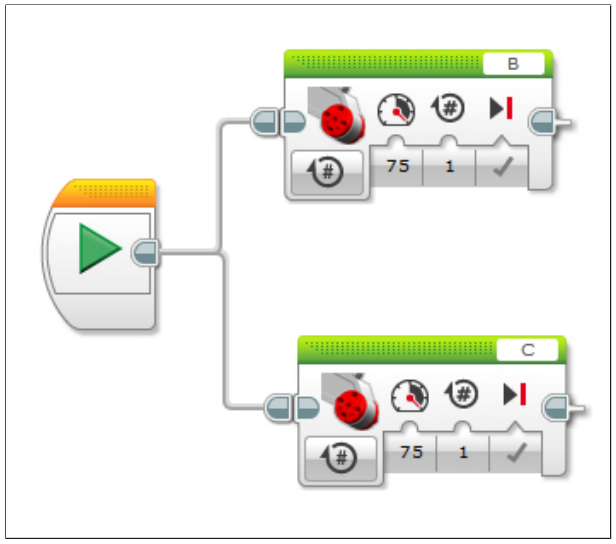
16. ábra



17. ábra



18. ábra



19. ábra

3.3.4. Feladatok

1. Készíts programot, melyben csak a nagy motorokat használva mozgatod a robotot! Elég ha egy tengelyfordulást megy előre a robot.

   Egy kis segítség: 19.ábra

2. Készíts programot, melyben a motorokat külön mozgatva folyamatosan egyenesen megy a robotod!

3. Készíts programot, melyben a robotnak zölden villog a lámpája míg megy előre, de ha nekimegy valaminek akkor megáll!

4. Készíts programot, melyben az ütközésérzékelőt használod úgy, hogy ha 0 értéket észlel, akkor pirosan világít, ha 1-es értéket kap, akkor Reset módban villog, végül ha 2-es értéket észlel, akkor előre megy egy másodpercig!

3.4. 4. alkalom

Ráhangolódás

- *Hány állapota van az érintésérzékelőnek?*
  - *1*
  - *2*
  - *3*
  - *4*

- *Melyik motorbeállítást kell használnunk, hogy robotunk egészen falnak ütközésig menjen?*
  - *on for seconds*
  - *on for rotations*
  - *on for degrees*
  - *on-off*
- *Mi a különbség az érintésérzékelő „Pressed” és „Bumped” állapota között?*
  - *Semmi, ezek lényegében ugyanazt jelentik.*
  - *A „Pressed” a benyomott állapotot jelenti, míg a „Bumped” egy benyomás és kiengedés együtteséből áll.*
  - *A „Pressed” a benyomott állapotot, míg a „Bumped” a kiengedettet jelenti.*
  - *A „Pressed” a kiengedettet állapotot, míg a „Bumped” a benyomottat jelenti.*
- *Párhuzamos programot...*
  - *egy erre való, speciális blokk segítségével tudunk létrehozni.*
  - *nem lehet ebben a szoftverben megvalósítani.*
  - *úgy tudunk létrehozni, hogy két külön indítás blokkot használunk.*
  - *„vezetékek” segítségével tudunk létrehozni.*
- *Egy párhuzamos program úgy működik, hogy...*
  - *robotunk a különböző szálakon lévő utasításokat egyszerre hajtja végre.*
  - *robotunk először azt az ágát hajtja végre a programnak, amely a szoftverben följebb van.*
  - *robotunk önkényesen eldönti milyen sorrendben hajtja végre a különböző szálakon levő utasításokat.*
  - *az a szál fut le előbb, melynek feltétele előbb teljesül, csak aztán a többi.*

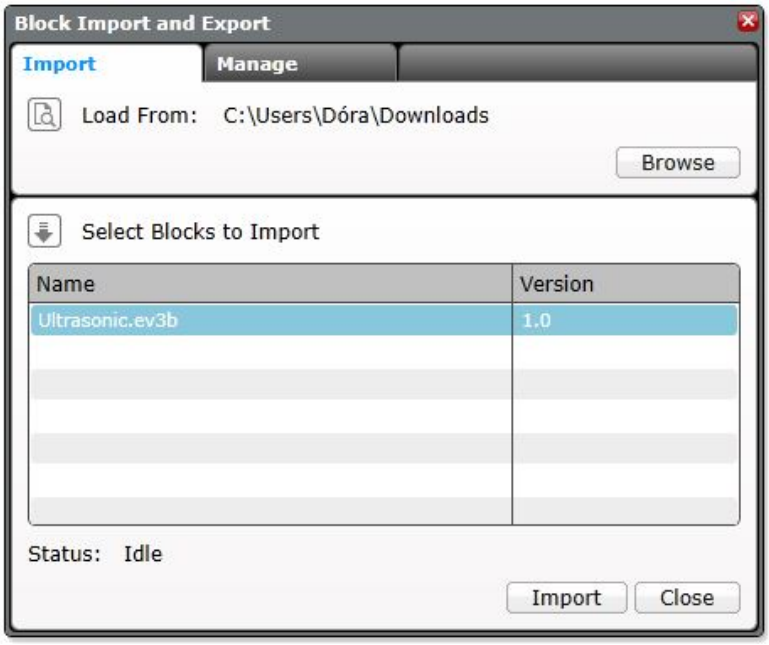
3.4.1. Ultrahangos és infravörös érzékelők használata

Ultrahangos távolságérzékelő

Mi jut eszetekbe az ultrahangról? Mire használják? Kik használják?

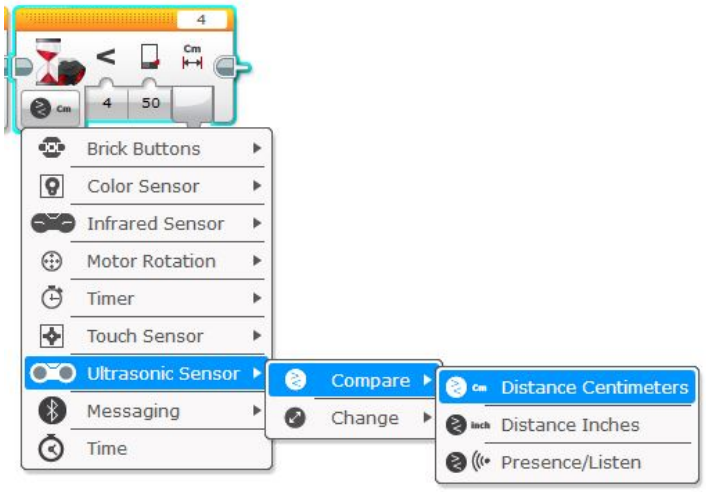
> *A legtöbb gyereknek a denevérek vagy a delfinek jutnak eszébe az ultrahangról, de vannak olyanok is akik tudják, hogy az orvostudomány is használja különféle vizsgálatokra. Itt megemlíthetjük még azt is, hogy néhány önvezető autó is ultrahangot használ a távolságok érzékelésére, ill. a tengeralattjárók is ezzel térképezik fel a környezetüket. Azért fontos ezeket megemlítenünk, mert robotunk érzékelője is a denevéréhez hasonló módon működik.*

Az EV3 ultrahangos távolságérzékelője az előtte lévő tárgyak (vagy fal) távolságát méri meg ultrahang segítségével, az érzékelőhöz viszonyítva. A távolságot körülbelül 5 cm-től 255 cm-ig vagy 100 hüvelykig (inch-ig) tudja megmérni, de ez a mérés pontatlan.
Ezt a szenzort is a várj blokkal használjuk. Behúzzuk a felületre a várj ikont, majd a bal alsó sarokban lévő óra gombra kattintva a lenyíló listából kiválasztjuk az ultrasonic szenzort.
Ha most használjuk először a szenzort, akkor valószínűleg nincs a listában, mert ezt nem tartalmazza a program, külön kell importálni. A blokkot a következő honlapról kell letölteni: `http://www.lego.com/en-us/mindstorms/downloads`, körülbelül az oldal közepén van egy kép a szenzorról, arra kattintva indíthatjuk a letöltést. Ha sikerült letölteni, akkor a szoftver *Tools* → *Block Import* menüjét kell megnyitnunk, majd az előugró ablakon a *Browse* gombot kell megkeresnünk. Erre kattintva ismét feljön egy ablak, amelyben a letöltött blokkot kell megkeresnünk és kiválasztanunk a számítógépünkön. Ha ezt sikeresen teljesítettük, akkor a 20. ábrán látható ablakhoz hasonlót kellene látnunk a saját számítógépünkön.



20. ábra. Blokk importálása

Ezután az *Import* gombra kell kattintanunk, majd a felugró ablakon az *Ok* gombra. Ezt követően figyelmeztet a szoftver, hogy importáltuk a blokkot, de újra kell indítanunk a szoftver ahhoz, hogy a változás látszódjon, azaz megjelenjenek a megfelelő blokkok. Ha újraindítottuk a szoftvert, akkor a várj blokkban is, és a Sensor blokkcsoportban is megjelenik az ultrasonic szenzor.



21. ábra. Az ultrahangérzékelő beállítása

Ezek után megnézhetjük, hogy hogyan is kell kezelni ezt a blokkot. Mint ahogy korábban említettem, minden érzékelőt a várj blokkon belül a Compare módban használunk, így ezt is. (Ez látható a 21. ábrán.) Ebben a módban háromféleképpen állíthatjuk be a szenzort. Az első kettőben a távolságot méri meg, de míg az elsőnél centiméterben, addig a másodiknál hüvelykben (inch-ben). (Mindig gondosan válasszuk ki, hogy melyik kell nekünk, mert a két mértékegység nem egyezik meg.) A harmadik beállítási mód egy olyan mód, amellyel a robot érzékelni tudja a környezetében lévő ultrahang szenzorokat. Ez azt jelenti, hogy ha ezt alkalmazzuk, akkor a robot addig fog várni, amíg nem érzékel egy másik szenzort.

Ha ezeket beállítottuk, akkor nézzük meg mit csinál ez a blokk. Nos, az első paraméter az összehasonlítás típusa nevet kapta, míg a második a küszöbértéket (a harmadik „változóval" a mért értéket tudjuk paraméterként átadni). A működése ugyanazon az elven alapszik mint az ütközésérzékelőé, azaz addig várakozik, amíg a mért érték meg nem felel a várt értéknek. Ez ebben az esetben azt jelenti, hogy a mért érték lehet egyenlő a várttal, de kisebb, nagyobb, stb. is.

A gyerekeknek az első paraméter beállítását úgy szoktam elmagyarázni, hogy vegyék úgy, hogy a bal oldalon lévő homokóra a mért értékünk, és ennek megfelelő műveleti jelet válasszanak a jobb oldali küszöbhöz.

Infravörös érzékelő

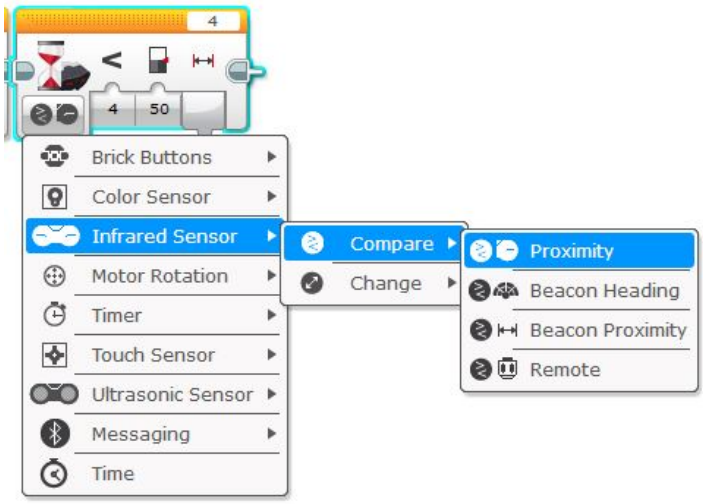
Az infravörös érzékelőt távolságérzékelőként és jel vevőként is lehet használni (az utóbbi esetben szükségünk van az „Infrared beacon"-re, azaz az infravörös távirányítóra), azonban mi most csak távolságérzékelőként fogjuk használni. Ennek ellenére nagy vonalakban megemlítem, hogy mit tud ez a szerkezet. Tehát, az infravörös szenzort három különböző módban lehet használni, ezek a következők:

Proximity (Közelségi mód) - Ez a távolságérzékelő része a szenzornak, ugyanúgy működik mint az ultrahangos távolságérzékelő, de ez körülbelül csak 70 cm-ig tud mérni. Az érték amit visszaad, viszont egy 0-100 közötti szám, ami nem cenimétert jelent. Ha az érték 0, akkor nagyon közel van az előtte lévő tárgy, ha 100, akkor pedig nagyon távol.

Beacon (Irányjeladó mód) - Két fajtája van, az egyik a „Proximity" (ilyenkor a távirányító távolságával dolgozunk), a másik a „Heading" (körülbelül egy félkörben megnézi, hogy előtte vagy valamelyik oldalán van-e a távirányító, és ennek megfelelően ad vissza értéket).

Remote (Távirányító mód) - Ebben a módban a szenzor érzékeli, hogy a távirányítón melyik gombo(ka)t nyomtuk meg.

Amint már korábban is említettem, távolságérzékelőként szeretnénk használni, így a Proximity módot kell választanunk, ill. még előtte a Compare-t (de ennek az okát már korábban tárgyaltuk). Így a blokkunk a 22. ábrán látható módon fog kinézni. Ennek a módnak a használata megegyezik az ultrasonic-éval, így ezt nem írom le újra.



22. ábra. Az infravörös érzékelő beállítása

3.4.2. Feladatok

1. Írj programot, amelyet végrehajtva a robot előre megy mindaddig, amíg a távolságérzékelője 15 cm-nél kisebb távolságot nem mér! Ekkor álljon meg!

2. Készíts programot, melyben a robot egyenesen halad, ha érzékel 10 cm-nél kisebb távolságban valamit, akkor megáll, kikerüli (mint egy kocka alakú dobozt), majd folytatja útját.

3. Készítsd el az előző programot úgy, hogy bármennyi dobozt ki tudjon kerülni!

4. Írj programot, amely a következő feladatot hajtja végre! A robot előre megy addig, amíg nem észlel valamit 10 cm-en belül maga előtt, ekkor elfordul 180°-ot, majd ezt ismételgeti. (Forgolódó program)

5. Írj programot, amelyet végrehajtva a robot addig halad előre, amíg 10 cm-en belül nem érzékel valamilyen akadályt! Ha valamilyen akadályt észlel, akkor elindul hátra, de ha nekitolat valaminek, akkor elindul előre. (Pattogó program)

3.4.3. Színesben szebb a világ! avagy hogyan használjuk a színérzékelőt

A színérzékelőt is a várj blokkal, azon belül compare módban használjuk. Háromféle módját különböztetjük meg az érzékelőnek, melyek a következők:

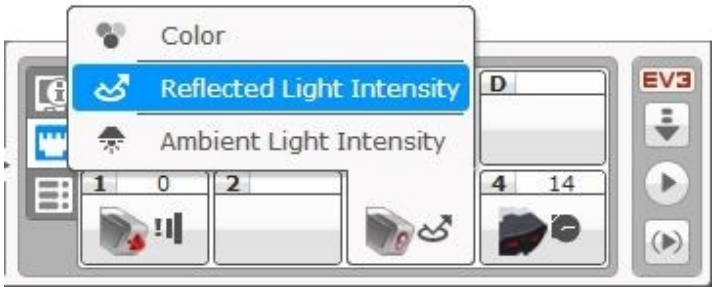
Colour - Ebben a módban hét színt tud felismerni a szenzor, továbbá egy nyolcadikat, az úgynevezett „No Color” (nincs szín) színt.

Reflected light intensity - Ez a visszavert fényerősség mód, ami úgy működik, hogy a szenzor egy piros fénnyel világítja meg a felületet, és az arról visszaverődő fény erősségét méri meg. Az érték egy 0 és 100 közötti egész szám, ahol 0 a sötétebb színt, míg a 100 a világosabb színt jelenti.

Ambient light intensity - Ezt szórt fényerősség módnak nevezik és ha ezt használjuk, akkor a szenzor kéken világít. Ebben a módban az érzékelő nem a felületről visszaverődő fényt méri, hanem a környezet fényét (ami bejut a szenzorba), és ezt egy százalékos értékként adja vissza. A mért érték ennek megfelelően egy 0 és 100 közötti egész szám lesz, ahol a 0 fényerősség nagyon sötétet jelent, a 100 pedig nagyon világosat.

Az utóbbi kettő paraméterezése hasonlít a távolságérzékelőkére, így ezeket nem részletezném. Helyette inkább más hasznos dolgot mutatok ehhez a blokkhoz. A szoftver hardveroldal részénél, ha csatlakoztatva van egy robot, akkor a port nézetben meg tudjuk nézni, hogy milyen értékeket mér a szenzorunk.

Ez akkor lehet hasznos, ha be szeretnénk lőni, hogy az adott felületről milyen értékek jönnek vissza és mit használhatunk a kódban a feladat megoldásához. Itt is tudjuk váltogatni a módokat, mégpedig úgy, hogy rákattintunk az adott érzékelőre és a kódban megjelenő listához hasonló menüből kiválasztjuk a megfelelőt. (23. ábra)



23. ábra. A hardveroldal mód választásakor

(A Color módnak annyi az „érdekessége”, hogy több színt is kijelölhetünk a lenyíló listából, és ekkor a sok szín közül, ha egyet érzékel, akkor lép tovább a program.)

Az összes érzékelő közül, a színérzékelő a legmacerásabb és legkiszámíthatatlanabb. Egyrészt biztosítani kell színeket az órához, másrészt nem mindegy, hogy milyen színeket használunk, mert sajnos nem megfelelően érzékeli őket. Legtöbbször színes szigetelőszalagokat használok, de ezekkel előfordult olyan, hogy a kéket, a zöldet és a feketét feketének vagy éppen kéknek érzékelte. Többször is próbálkoztam a fényviszonyok javításával, de ez sem bizonyult eddig még végleges és jó megoldásnak. Legjobb eredményt nyilván a LEGO© elemekkel lehet elérni, de ez nem mindig oldható meg a feladatok jellege miatt.
Hallottam olyanról, hogy valaki megkereste az érzékelt színek RGB kódját, ezeket a színeket kinyomtatta lapokra és ezzel használta az érzékelőt, ami így csodásan működött. Emiatt én is beleástam magam az internet bugyraiba, és találtam egy megbízhatónak tűnő oldalt[4]*, melyen az alábbi értékek szerepeltek:*

- ***fekete*** *(33,6,15),*
- ***piros*** *(255,49,27),*
- ***kék*** *(47,104,154),*
- ***fehér*** *(255,255,255),*
- ***zöld*** *(34,139,31),*
- ***barna*** *(87,59,29).*
- ***sárga*** *(255,244,48),*

Felmerülhet a kérdés, hogy ezek miért nem egyeznek meg a hivatalos RGB kódokkal. (Pl.: A zöld miért nem (0,255,0)?) Az oldalon részletezik, hogy ezek a színszenzorból kiolvasott értékek (amit egy Python-ban írt kóddal nyertek ki a robotból), így ezek nem feltétlenül egyeznek meg a LEGO© elemek színeivel. Ennek ellenére érdemes kipróbálni, akár más színkódokkal is.
Mindenesetre én kipróbáltam, és azt tapasztaltam, hogy az így készített színes lapokkal, jobban működik a színérzékelő. (Legalábbis nagyobb arányban mutatta a helyes színt, mint eddig.) Úgyhogy, csak ajánlani tudom, próbáljátok ki Ti is! :)

[4] `http://blog.thekitchenscientist.co.uk/2015/02/ev3-python-raw-rgb-values.html`

3.4.4. Feladatok

1. Írj programot, amelyet végrehajtva a robot egyenesen halad előre mindaddig, amíg a fényérzékelője az alapszíntől eltérő színt nem észlel, ekkor álljon meg és tolasson 2 mp-ig! (A feladat megoldása előtt mérjük meg, hogy a különböző színű felületekről milyen intenzitású fényt ver vissza.)

2. Írj programot, amelyet végrehajtva a robot addig forog, amíg sötétebb színt nem érzékel!

3. Írj programot, melyben a robot egy sötét vonalat követ! (Vonalkövetés)

4. Írj programot, amelyet végrehajtva a robot egyenesen megy előre addig, amíg egy sötét csíkot el nem ér. Ekkor balra fordul kb. 90°-ot, majd egyenesen megy, amíg akadálynak nem ütközik!

5. Készíts programot, melyben a robot 20-as sebességgel halad előre, de ha zöld színt észlel, akkor megáll és pirosan felvillantja a ledjét!

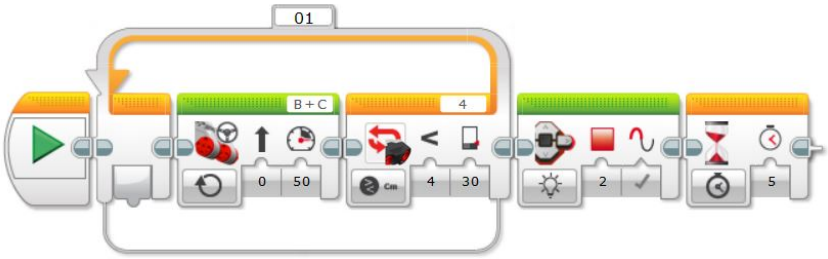
6. Készíts programot, melyben a robot 20-as sebességgel halad előre, ha kék színt érzékel, akkor tesz egy fordulatot a tengelye körül, majd folytatja útját az eredeti irányban.

7. Készíts programot, melyben a robot a következő feladatot oldja meg: Ha sárga színt érzékel a robot, akkor elindul, de ha pirosat, akkor megáll!

3.5. 5. alkalom

Ráhangolódás

- *Melyik állatok tudnak az alábbiak közül ultrahangot kibocsátani?*
  - *denevér, veréb*
  - *denevér, macska*
  - *denevér, tintahal*
  - *denevér, delfin*
- *Az ultrahangos távolságérzékelővel milyen mértékegységben mértünk távolságot?*
  - *centiméter*
  - *inch*
  - *méter*
  - *láb*
- *Milyen relációt használtunk az ultrahangos távolságérzékelőnél?*
  - *egyenlő*
  - *kisebb/nagyobb-egyenlő*
- *Ha a képen látható programot elindítjuk a roboton mikor fog pirosan villogni a led?*
  - *Ha be van nyomva az érintésérzékelő.*
  - *Ha 30 cm-en belül észlel valamit az ultrahangos távolságérzékelő.*
  - *Ha 30 cm-en kívül észlel valamit az ultrahangos távolságérzékelő.*
  - *Ha ki van engedve az érintésérzékelő.*

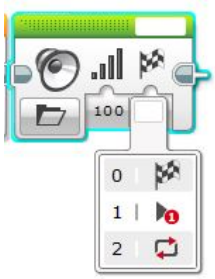


- *A homokórás blokk csak arra jó, hogy...*
  - *várni tudjon a robot pár másodpercig.*
  - *várni tudjon a robot az érintésérzékelő megnyomásáig.*
  - *várni tudjon a robot, amíg nem lát valamit.*
  - *várni tudjon a robot egy beállított esemény bekövetkezéséig.*
- *A színérzékelő csak arra jó, hogy...*
  - *visszavert fény erősségét mérje.*
  - *színeket ismerjen fel.*
  - *visszavert fény és környezetében lévő fény erősségét mérje és színeket ismerjen fel.*
  - *színesen tudjon világítani a robotunk.*

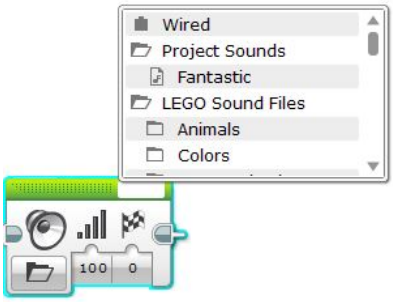
3.5.1. Hangoskodjunk! avagy hogyan játszunk le hangot az EV3-mal

A robot képes hangok lejátszására is, de nem ám olyan ósdi módon, mint a mobilok, ez annál sokkal menőbb! Mégpedig azért, mert négyféle mód közül választhatunk (ami igazándiból három) de ami ennél is izgibb, hogy az egyik a zongora hangok egy csoportja! Nézzük meg, hogyan működik!
Ezt a blokkot az action (zöld) blokkok között találjuk, azon belül az utolsó előtti helyen. A fajtáit a szokásos módon tudjuk megnézni, ezeket láthatjuk a 24. ábrán.
Menjünk sorban! A Stop-nak nincs paramétere, leállítja a zenét. A másik háromnak viszont elég sokféle paramétere van, de szerencsére vannak köztük hasonlóak is. A Play File módban előre rögzített hangok közül tudunk lejátszani egyet. A módot kiválasztva, megjelenik két paraméter. Az elsővel a hangerőt tudjuk szabályozni, míg a másodikkal a lejátszás módját. (25. ábra) Utóbbira kattintva három lehetőség közül lehet választani. A 0-val jelölt azt jelenti, hogy a program addig nem lép tovább a következő blokkra, míg véget nem ér a zene lejátszása. Az egyes számút választva, egyszer lejátssza a hangot, de közben már a következő blokk végrehajtását is elkezdi. Az utolsó a „Repeat" nevet kapta, és az adott hangot addig ismételgeti, amíg az utána lévő blokkok be nem fejeződnek.
Ha már tudjuk, hogy hogyan kell lejátszani a hangot, nem ártana ha keresnénk valamit, amit le tudunk játszani. Szerencsére a szoftverrel kapunk beépített hangokat is. Ezeket a blokk jobb felső sarkában található fehér téglalapra (ahol általában a használt portok szoktak lenni) kattintva tudjuk elérni. (26. ábra) Viszont nemcsak az előre megadottakból tudunk válogatni, a „Wired" opciót használva, mint egy változót tudjuk megadni a lejátszandó hangot (de ebbe most nem megyünk bele).

24. ábra



25. ábra



26. ábra

Az utolsó két mód nagyon hasonló. A „Play Tone” módban egy bizonyos frekvenciájú hangot játszik le a robot. Ezt a frekvenciát kiválaszthatjuk az első paraméter lenyíló ablakából vagy meg is adhatjuk egy 300 és 10000 közti (nem feltétlenül egész) szám formájában. A második paramétere ennek a módnak a hang lejátszásának időtartama, amit másodpercben kell megadni. A másik két paraméter megegyezik a Play File-nál tárgyaltakkal. A „Play Note” az előzőtől csupán az első paraméterében tér el, viszont itt frekvencia helyett egy zongorán tudjuk megadni a lejátszandó hangot.

3.5.2. Feladatok

1. Készíts prgoramot, melyben a *Good job* hangot játszod le úgy, hogy utána 3 másodpercig pirosan villog a robot lámpája! Próbáld ki mindhárom lejátszási módot!

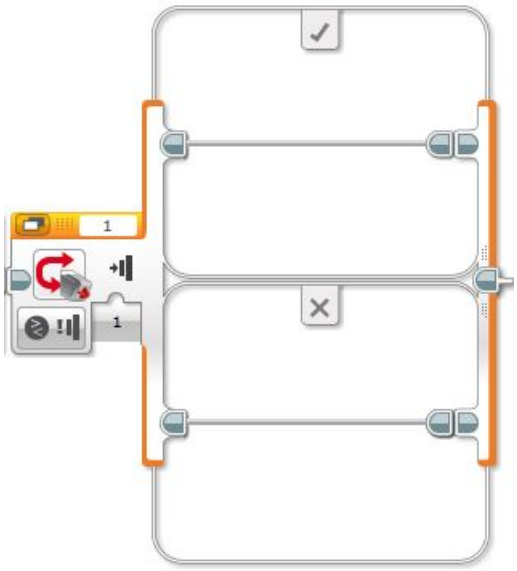
2. Készíts programot, melyben a robot egyenesen halad előre, de ha nekiütközik valaminek, akkor a *No* hangot adja ki 2-es lejátszási módban, és utána 2.3 másodpercig villogtatja pirosan a lámpáját!

3. Írj programot, melyben a robot ha érzékel egy színt, akkor kimondja angolul a nevét! (Hangos színek)

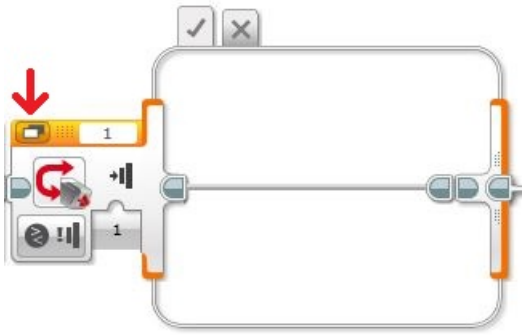
4. Készíts programot, melyben a robot egy vonalon halad, de ha érzékel maga előtt 10 cm-en belül valamilyen tárgyat, akkor dudál egyet a C frekvencián, majd folytatja útját!

3.5.3. Elágazás megismerése

Mielőtt megtanulnánk használni az elágazásokat, érdemes (bevezető feladatként) megoldani a Hangos színek fantázianevű feladatot. Ezt eddig többszálú feladatként oldottuk meg, de igazándiból semmi nem indokolja a párhuzamos programozást, mert nem használjuk ki a módszer adta lehetőségeket. A feladatot egyszerűbben és átláthatóbban is meg toldjuk oldani elágazás (Switch) segítségével. Ezt a blokkot a Flow Control (narancssárga) csoportban találjuk és alapállapotban a 27. ábrán látható módon néz ki. A nézetet viszont meg tudjuk változtatni a 28. ábrán láthatóvá, ha a bal felső sarokban lévő „Switch to Flat View” gombra kattintunk. Ezt az ábrán piros nyíl jelöli.

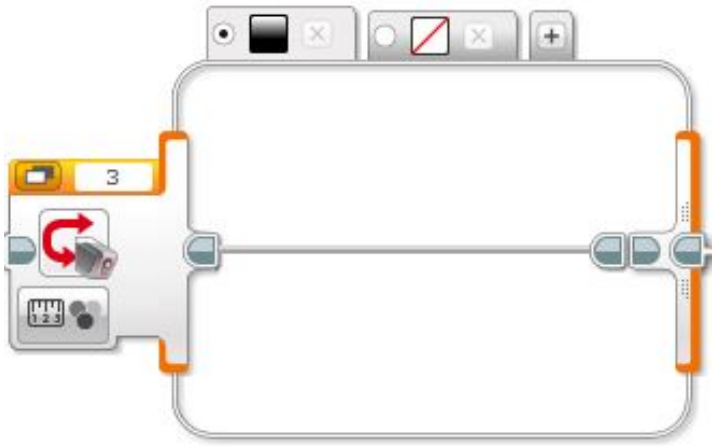


27. ábra. Elágazás egyik fajtája



28. ábra. Elágazás másik fajtája

Ennél a blokknál is a bal alsó sarokban lévő gombbal tudjuk megváltoztatni a módot és itt is ugyanazokkal a lehetőségekkel találkozunk, mint amik a várj és a ciklus blokknál előfordultak. Viszont, ha egy szinttel beljebb megyünk, akkor már némelyik opciónál felbukkan egy „Measure" mód a Compare-n kívül. Nos, ha a Compare-t választjuk, akkor egy egyszerű, igaz és hamis ágból álló elágazásunk van, viszont ha a Measure-t, akkor egy többágú elágazással (switch-case-zel) állunk szemben. (29. ábra) Utóbbit akkor használjuk, ha például minden színre más „reakciót" szeretnénk kiváltani. Ebben a módban be kell állítanunk egy „Default Case" esetet, ami azt jelenti, hogy ha a feltételek közül nem talál olyat, ami megfelelne az adott értéknek, akkor ezt fogja végrehajtani. Ezt a 29. ábra tetején látható menü megfelelő rádiógombjára kattintva tehetjük meg. (A másik elágazás nézetben is hasonló helyen találjuk a rádiógombot.)



29. ábra. Többágú elágazás (switch-case) színszenzorral

3.5.4. Feladatok

1. Készítsd el a Hangos színek c. feladatot a switch blokk használatával!
2. Készítsd el a Vonalkövető feladatot az elágazás használatával!
3. Írj programot, melyben a robot a következő feltételek alapján cselekszik:

   Ha piros színt érzékel, akkor pirosan villogtatja a ledjét 2 másodpercig,
   ha zöldet, akkor reset módban villog,
   ha kéket, akkor egyhelyben megfordul,
   ha sárgát, akkor angolul kimondja, h sárga,
   ha fehéret, akkor egy zongora hangot játszik le,
   ha pedig feketét észlel, akkor azt mondja, hogy *Bravo*.

4. Készíts programot, melyben egyenesen halad előre a robot és ha piros/zöld/kék csíkot érzékel, akkor kimondja a szín nevét angolul, de ha feketét érzékel akkor leállítja a motort!

3.6. 6. alkalom

Ráhangolódás

- *Melyik módot kell választanunk a hang-blokkban, ha zongora hangokat szeretnénk robotunkon megszólaltatni?*

  - *STOP*
  - *Play File*
  - *Play Tone*
  - *Play Note*

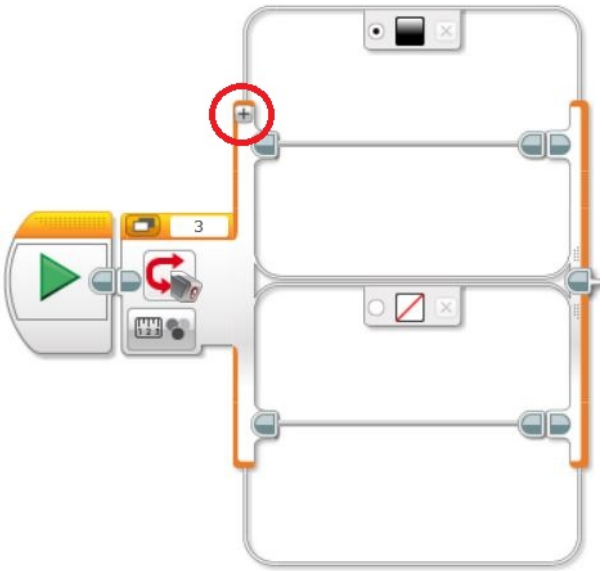
- *Elágazás szerkezet használatakor a pipával jelölt rész...*
  (Segítségként berakhatjuk az elágazás szerkezet képét)
  - *mindenképp lefut.*
  - *csak akkor fut le, ha a megadott feltétel hamis.*
  - *csak akkor fut le, ha a megadott feltétel igaz.*
  - *az X-el jelölttel párhuzamosan fut le.*

- *Mi a különbség az elágazás szerkezet és a többszálú program között?*
  - *Semmi*
  - *Elágazásnál egyszerre csak az egyik ágba megy bele a program.*
  - *Többszálú programnál csak az egyik ágba megy bele a program.*
  - *Az elágazásnak csak két ága lehet.*

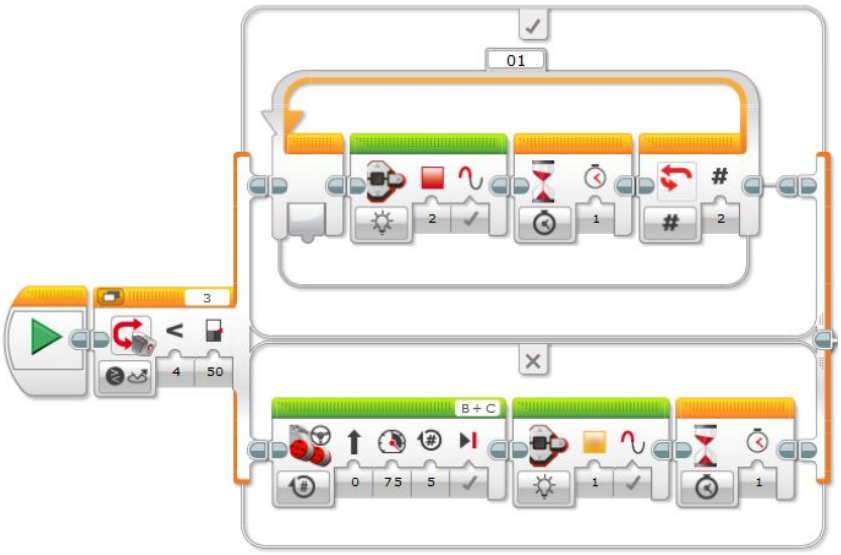
- *Ha színérzékelő esetén elágazást használunk, akkor az alapértelmezett ágba mi kerül?*
  - *Amit akkor csinál a robot, ha feketét észlel.*
  - *Amit akkor csinál a robot, ha zöldet észlel.*
  - *Amit akkor csinál, amikor egyik feltétel sem teljesül.*
  - *Amit akkor csinál a robot, ha fehéret észlel.*

- *Igaz-e, hogy elágazás használatával maximum két színt tudunk egyszerre vizsgálni?*
  - *Igaz*
  - *Hamis*

- *A képen piros körrel jelölt + jel arra szolgál, hogy...*
  - *más érzékelő értékét is figyelembe tudjuk venni(színen kívül).*
  - *össze tudjuk „csukni” a szerkezetet, hogy átláthatóbb legyen.*
  - *plusz feltételt adhassunk hozzá, új színt vehessünk fel.*
  - *az alapállapotot ki tudjuk jelölni.*



- *Az alábbi program esetén mikor villog pirosan a led?*
  - *Ha a robot világos (fehér) színt észlel.*
  - *Ha a robot sötét (fekete) színt észlel.*
  - *Ha a robotnak be van nyomva az érintésérzékelője.*
  - *Ha az ultrahangos távolságérzékelő 50 cm-en belül érzékel valamit.*

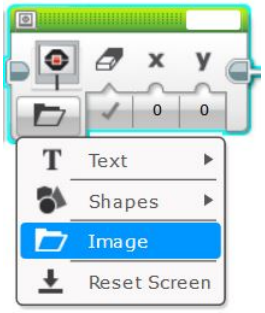


3.6.1. Rajzolás a tégla képernyőjére

A tégla felső részén található kijelzővel már sokat ismerkedtünk a korábbi alkalmak során. Itt találunk számos beállítási lehetőséget, el tudtuk indítani a már robotra töltött programjainkat és információkat is le tudtunk olvasni róla például a töltöttségi állapotot. A téglán lévő kijelzőt azonban még másra is tudjuk használni, írhatunk, rajzolhatunk rá ezzel kiegészítve programunkat.

A zöld színű blokkok között találjuk a szoftverben a kijelző blokkot (Display), ez lesz segítségünkre a kijelző programozásakor.

A mód választó gombra kattintva több lehetőség közül is választhatunk, melyekről részletesen később lesz szó. Először ismerkedjünk meg a blokk elemeivel. A bal felső sarkában található a kijelző előnézeti gomb, melynek segítségével ellenőrizhetjük még a robotunkon való futtatás előtt, hogy mi fog megjelenni a kijelzőn. Ezt majd látni fogjuk, hogy nagyon hasznos lesz feladataink megoldása során, például, amikor koordináták segítségével szeretnénk meghatározni egy alakzat elhelyezkedését a kijelzőn. De ne szaladjunk még ennyire előre! A jobb felső sarokban most egy üres fehér téglalap látható. Ez valójában egy szövegbeviteli mező. Azonban ez a mező nem áll rendelkezésünkre minden módban (nem is lenne rá szükség) csak a Text és az Image választásoknál látható.
Most pedig ismerkedjünk meg részletesen a választható módokkal!



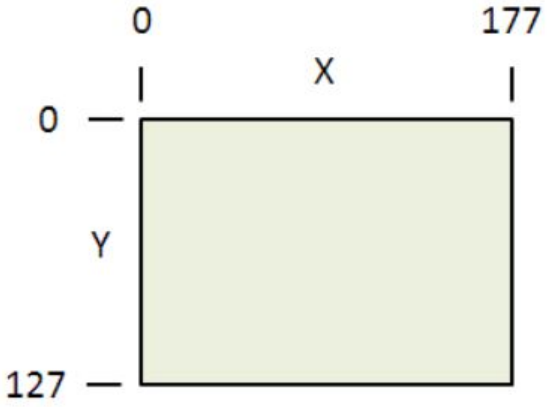
30. ábra. Kijelző blokk (Display)

31. ábra. Szöveg kiírása a képernyőre

Text

Az első mód választásával lehetőségünk van szöveget írni a robot képernyőjére. Fontos tudnunk, hogy ékezetes karaktereket nem használhatunk szövegek kiírása során!
Szöveg megjelenítésére két lehetőségünk is van, ahogy ezt a 31. ábra is mutatja. Nézzük először az elsőt, vagyis amikor a **Pixels**-t választjuk. Ekkor a robot képernyőjét pixelekre „bontjuk”, így bárhova írhatunk a robot képernyőjén. Ahhoz, hogy pontosan meg tudjuk határozni hova szeretnénk elhelyezni a szöveget, szükségünk lesz egy koordináta-rendszerre, melynek segítségével a kijelző minden pixelét egy (x,y) párnak feleltethetjük meg egyértelműen. Nézzük, hogy is néz ki ez a koordináta-rendszer.

Vegyük figyelembe, hogy a gyerekek ismerik-e már a koordináta-rendszert, előkerült-e már matematikai tanulmányaik során! Ha nem, akkor tapasztalataim szerint fontos rá több időt szánni, mert nagy akadály lehet a feladatok megoldása során, ha ezt a részt nem értették meg jól. Én fel szoktam rajzolni a robot képernyőjét a táblára, illetve a koordináta-rendszert és először csak szóban kérdezgetek tőlük a táblánál pontokat, hogy az vajon hova fog kerülni a roboton.



32. ábra. A kijelző koordináta-rendszere

A 32. ábrán látható, hogy a robot kijelzőjét hogyan koordinátázzuk. Kicsit más, mint a matematikában megszokott koordináta-rendszer.

> *Ezért még ha diákjaink tanulták is geometriából fontos, hogy részletesen beszéljünk róla, nehogy belezavarodjanak épp e miatt. A könnyebb megértés érdekében hivatkozhatunk az x-koordinátára mint oszlopszámra, az y-ra pedig mint sorszámra.*

A tengelyek megegyeznek a Descartes-féle koordináta-rendszer tengelyeivel, azonban a számozásban észrevehetünk némi furcsaságot. Az x koordináta 0-tól 177-ig fut balról jobbra növekedve. Azonban az y máshogy viselkedik, fentről (vagyis a robotunk képernyőjének bal felső sarkából) lefele nőnek az y értékek. A kijelző bal felső sarkában 0, a bal alsóban pedig 127 az y paraméter értéke.

> *Fontos megjegyeznünk ezen a ponton, hogy attól függetlenül, hogy az x képernyőn látható maximum értéke 177, y-é pedig 127, ettől még nagyobb szám megadására is lehetőségünk van a programozás során! A diákok erre rá is szoktak néha kérdezni, ha nem akkor is sokszor belefutnak feladatmegoldás közben. Ezt akár egy feladaton keresztül érdemes is szemléltetni.*

Most, hogy megismerkedtünk a kijelző koordináta-rendszerével nézzük meg milyen beállítási lehetőségeink vannak a blokkon!
Az első paraméter a **képernyő törlése**, melyet egy kis radír ikon jelöl. Hogyha ezt igazra állítjuk, akkor a szöveg kiírása előtt törölni fogja a teljes képernyő tartalmát, így eltűnik például a robot bekapcsolásakor alapértelmezetten megjelenő menü (ami a program futása után újra látható lesz), illetve ami előtte rajta volt, legyen ez akár rajz vagy bármilyen szöveg. Tehát így egy tiszta képernyőt kapunk melyre rákerül az általunk kiírni kívánt szöveg. Ellenben, ha ezt az értéket hamisra állítjuk nem fog törlődni a szöveg kiírása előtt a menü sem, így az nem lesz jól olvasható!

> *Érdemes tehát az ilyen jellegű feladatoknál legalább az első blokkon törölni a képernyőt, hogy a menü eltűnjön és utána szabadon írhassunk, rajzolhassunk robotunk képernyőjére.*

33. ábra. Elhelyezés a képernyőn

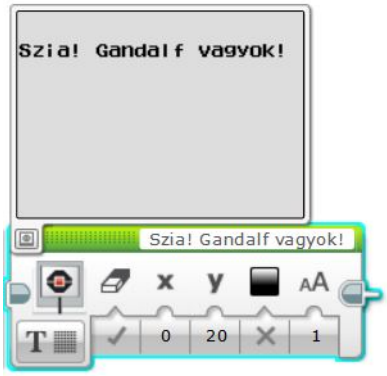
A második, illetve a harmadik beállítási lehetőség az **x és y koordináták** beállítása, mellyel megadjuk honnan kezdődjön a szövegünk. Ahogy a 33. ábrán is látható a paraméter beállításához használhatjuk

a megjelenő csúszkát (segítségképpen az x koordináta esetén ez vízszintesen jelenik meg, míg az y esetén függőlegesen, ahogy a koordináta-rendszerben is).

Különös ezen csúszkákkal kapcsolatban, hogy míg a bal felső sarka a képernyőnknek a (0,0) pont, az x csúszkájának minimuma 0 helyett -177, y-é pedig -127. Érdemes erre felhívni diákjaink figyelmét, nehogy úgy gondolják ezek a minimum értékek jelentik a bal felső sarkot!

Érdemes figyelni a kijelző előnézeti képét minden beállítás közben. Ahogy állítjuk a csúszkák segítségével a koordinátákat, úgy változik a megjelenített kép is. Az x és y értékek beállításához nem muszáj a csúszkákat használnunk, pontos érték beállításakor könnyebb kézzel beírni a kívánt számokat. A következő blokk-paraméter a **betűszín** beállítása. Fekete és fehér közül választhatunk. Fehér választása esetén megjelenik a kiírt szöveg mögött egy fekete háttér az olvashatóság érdekében (ld. 33. ábrán). A blokkon az utolsó beállítási lehetőség a **karakterméretre és stílusra** vonatkozik. Itt 3 opció közül választhatunk. A normál(0) típussal írt szöveg látható a 34. ábrán, a normál méretű, de félkövéren kiemelt(1) a 35. ábrán,a nagy(2) betűmérettel írt pedig a 36. ábrán látható.

34. ábra

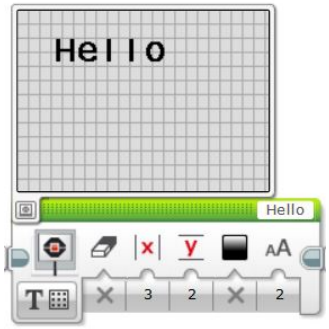


35. ábra

36. ábra

Ahogy a 36. ábrán is látszik, figyelni kell a szöveg hosszára! Ugyanaz a szöveg normál betűmérettel kifért a kijelzőre, míg nagy betűmérettel már a fele lemaradt róla, nem látható! Fontos, hogy erről a hibalehetőségről beszéljünk, a gyerekeket meg lehet kérdezni milyen megoldási ötleteket tudnak mondani! Ebben a helyzetben a legjobb megoldás, ha két külön blokkba írjuk, ami különböző sorba fog kerülni és a koordináták segítségével megadjuk hol kezdődjön a következő sor. De hogy tudjuk kiszámolni pontosan hány pixellel kellene ez lejjebb kezdenünk? Ez elég macerás lenne. Épp ennek a megkönnyítésére fogjuk megnézni a következő szövegkiírási lehetőséget!

Most ismerkedjünk meg a szövegmegjelenítés másik módjával is! Ez a **Grid**, vagyis amikor a kijelzőnk nem pixelekre, hanem egy téglalaprácsra van felosztva. Ebben már sokkal könnyebben fogunk tudni mozogni, különböző sorokba elhelyezni szövegeket. A kijelző 21 oszlopra és 11 sorra van felosztva. Minden oszlop 8 pixel széles és minden sor 10 pixel magas. Ahogy a 37. ábrán is látható a blokk bemeneti paraméterei megegyeznek a pixeles blokkéval. Csak az előnézeti kép különbözik a rács miatt, illetve az x és y értékek jelentenek mást, mint az előző esetben. Itt x és y paraméterek a sorokat és oszlopokat jelentik. (0,0) a bal felső sarok és ahogy



37. ábra. Téglalaprácson nagy betűk

növeljük 1-el valamelyik értéket, úgy fog a szöveg jobbra, illetve lefelé lépni egy sort/oszlopot.

> *Úgy vettem észre, hogy a diákok ezt sokkal könnyebben és gyorsabban meg tudják érteni, mint a pixelekkel való számolgatást, így érdemes a csoporthoz igazítani, hogy melyik lehetőséget mutatjuk meg előbb, illetve, hogy a feladatokat melyikkel oldjuk meg.*

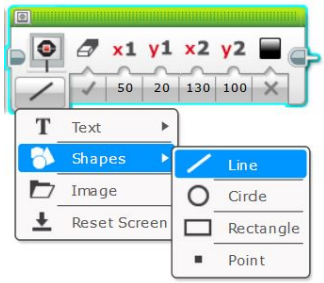
A 37. ábrán látható, hogy a nagybetűvel írt szöveg karakterei 2 sor magasak és minden karakter 2 oszlopot foglal el, míg a normál betűmérettel írt szöveg csak 1 sort és egy oszlopot foglal el (ld.: 38. ábra). Tehát ahhoz, hogy az általunk a képernyőre írt szövegek ne kerüljenek fedésbe érdemes figyelnünk ezekre a mértékekre.

38. ábra. Téglalaprácson normál betűk

Alakzatok

Nem csak szövegek kiírására van azonban lehetőségünk! Alakzatokat is megjeleníthetünk robotunk képernyőjén. Egyenes, kör, téglalap és pont rajzolásával ötletes ábrákat készíthetünk.

> *A gyerekek itt szabadon engedhetik fantáziájukat, a 4 alakzatból „építkezhetnek". Nyugodtan szánhatunk arra időt, hogy maguktól alkossanak egy képet a robot kijelzőjére, hiszen így fogják a leginkább elsajátítani ezek működését!*



39. ábra. Rajzolható alakzatok

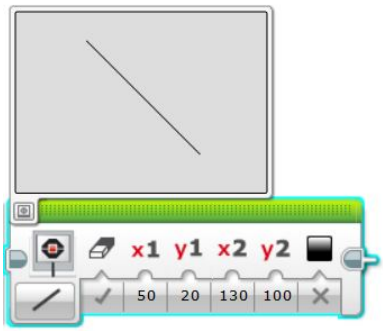
Nézzük meg külön-külön az egyes alakzatok paramétereit. Mindegyiknél az első beállítási lehetőség a képernyőtörlés. Ez ugyanazt jelenti, mint az előző blokkok esetén. Az utolsó paraméter pedig a szín beállítása (fekete/fehér) ami az alakzat körvonalára és kitöltésére egyaránt vonatkozik. A többi paraméter azonban a különböző alakzatok esetén más és más lesz, ezért érdemes velük részletesen megismerkednünk.

A 40. ábrán látható egy **vonal** rajza. Itt a blokk második és harmadik paramétere koordináta (x1,y1), mely egy pontot határoz meg, (x2, y2) pedig egy másik pontot a kijelzőn. A kijelzőn megjelenő alakzat tehát egy (x1,y1) és (x2,y2) koordináták közti egyenes szakasz lesz.

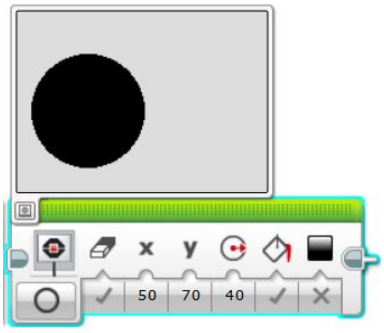
A 41. ábrán egy **kör** képét láthatjuk. Ebben az esetben az x és y koordináták a kör középpontját határozzák meg. A negyedik beállítási lehetőség a kör sugara, melyet pixelben mérve kell megadnunk. A következő bemeneti értékkel határozhatjuk meg, hogy ki akarjuk-e tölteni a kört, vagy csak egy körvonalat szeretnénk megjeleníteni robotunk képernyőjén.

A 42. ábrán egy **téglalapot** látunk. Az x, y koordinátákkal ezen alakzat esetén a bal felső sarok pozícióját tudjuk megadni. A következő két érték pedig arra szolgál, hogy pixelben megadjuk a téglalapunk szélességét, illetve magasságát. Az előző alakzathoz hasonlóan itt is meg tudjuk határozni, hogy csak határoló vonalat rajzoljunk vagy kitöltött alakzatot.

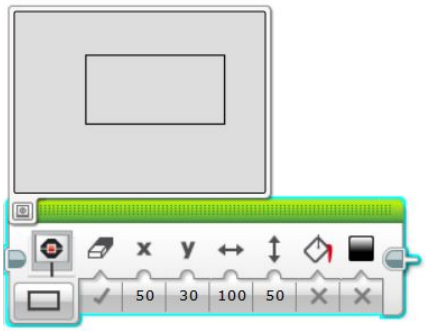
A negyedik alakzat, melyet ezen blokk segítségével rajzolni tudunk a **pont**. Erről azért nem szerepel kép, mert az előnézetben alig látszik az a kis pixel, melyet a robotunk képernyője megjelenít. Pont esetén csak az x és y koordinátákat kell meghatároznunk, azt a pixelt, melyet fehérre vagy feketére szeretnénk színezni.



40. ábra



41. ábra



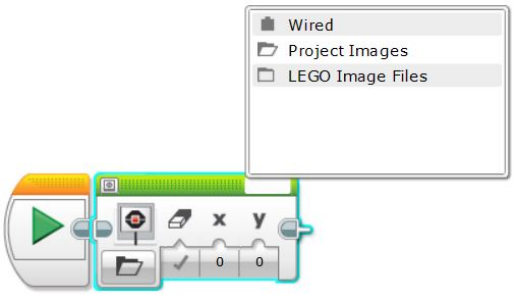
42. ábra

Ahogy láttuk, minden alakzat esetében színnek választhatjuk a fehéret is fekete helyett. Azonban ha csak a színt fehérre változtatjuk nem fogjuk látni alakzatunkat a képernyőn. A fehérrel rajzolás csak úgy működik, hogyha az adott blokk képernyő törlését hamisra állítjuk és előtte egy feketével kitöltött alakzatot rajzolunk ugyanerre a helyre.

Például egy egész képernyőt kitöltő fekete téglalapra tudunk rajzolni fehér vonalat, kört, pontokat.

Kép megjelenítése fájlból

A következő képernyőhöz kötődő funkció a fájlból való megjelenítés. A jobb felső sarokban található fehér téglalapra kattintva tudjuk kiválasztani a megjelenítendő képet. A „LEGO Image Files” kategóriában olyan képfájlok közül válogathatunk, melyek már letöltéskor a szoftver részét képezik. A listában a másik fájl neve „Project Images”. Ez azokat a fájlokat tartalmazza, melyeket már korábban használtunk a projektünkben, így biztosítva az újbóli gyors elérést.



43. ábra. Kép fájlból

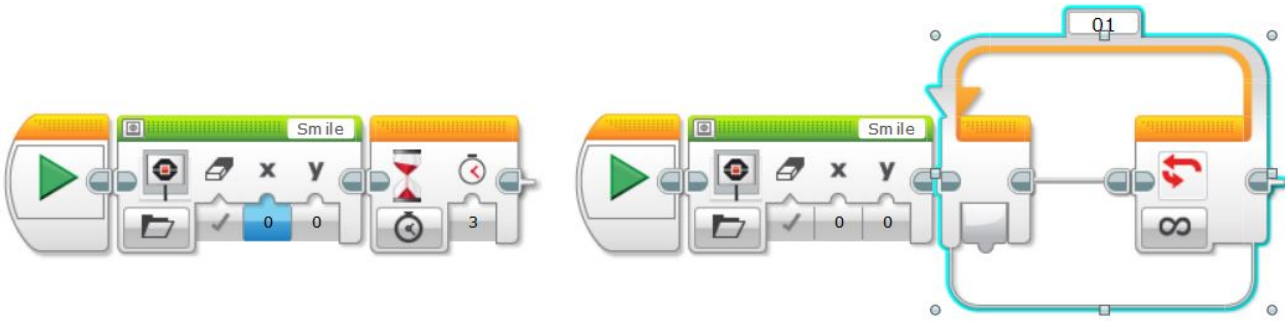
Képernyő törlés

Az utolsó mód, melyről még nem esett szó a Reset Screen, vagyis a képernyő törlés. A neve félrevezető lehet, mivel ez a blokk nem arra szolgál, hogy a képernyőről letöröljük az eddig megjelenített ábrákat, szövegeket és újat rajzoljunk rá! Akkor használjuk ezt a blokkot, hogyha megjelenítettünk valamit a robotunk képernyőjén és még a programunk futása során vissza szeretnék állítani a képernyőt arra, mely alapból egy program futása során megjelenne.

44. ábra. Reset Screen

Ahhoz, hogy robotunk képernyőjén lássuk is a képet, amit meg szeretnénk jeleníteni programunkat a 45. ábrán látható módokhoz hasonlóan kell megírnunk. Ha nem rakunk várakozást vagy ciklust a képernyő blokk mögé, akkor a program működési elvéből adódóan nem fogjuk látni a megjeleníteni

kívánt képet a roboton. Ez abból következik, hogy a program futása végén letörli ami a képernyőn volt és vissza áll az eredeti „menü nézetbe”. Ha nem rakunk valamilyen késleltetést a programunkba akkor ez a teljes képernyőtörlés olyan gyorsan megtörténik, hogy észre sem vesszük a képet, amit meg szerettünk volna jeleníteni!



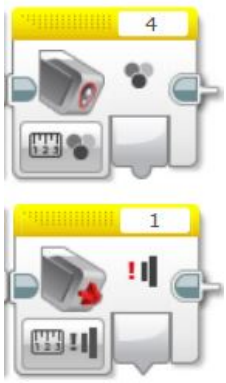
45. ábra. Képernyőn kép megjelenítése

3.6.2. Feladatok

1. Írd ki saját becenevedet a robot képernyőjére!
2. Írd ki a saját nevedet a robot képernyőjére úgy, hogy vezetékneved a bal felső sarokban legyen, keresztneved pedig az alatt! Használj félkövér betűtípust!
3. Jeleníts meg robotod képernyőjén egy tetszőleges képet fájlból 3 másodpercig!
4. Írj programot, melyben a roboton lévő led pirosan világít, amíg be van nyomva az érintésérzékelő és zölden, ha ki van engedve! Világítás mellett mosolyogjon, amíg be van nyomva, szomorú arcot mutasson, ha ki van engedve!
5. Rajzolj egy házikót alakzatok segítségével a robot képernyőjére! A részletek rád vannak bízva!
6. Készíts programot, melyben a robotod a következőt hajtja végre: amikor elindítod a programot a képernyőjén 3-tól visszaszámol 1-ig 1 mp-enként, majd kiírja, hogy START a képernyő közepére és motorjait beindítva leír egy tetszőleges alakzatot (pl.: négyzetet, S-alakot).
7. Készíts programot, mely során robotod lassan egyenesen előre halad a pályán (fehér nagy lap/felület fekete, piros alakzatokkal). Ha fekete színt lát írja ki a képernyőre, hogy „FEKETE”, ha pirosat, akkor azt, hogy „PIROS”, ha fehéret, akkor pedig azt, hogy „FEHER”!

3.6.3. Érzékelő blokkok használata

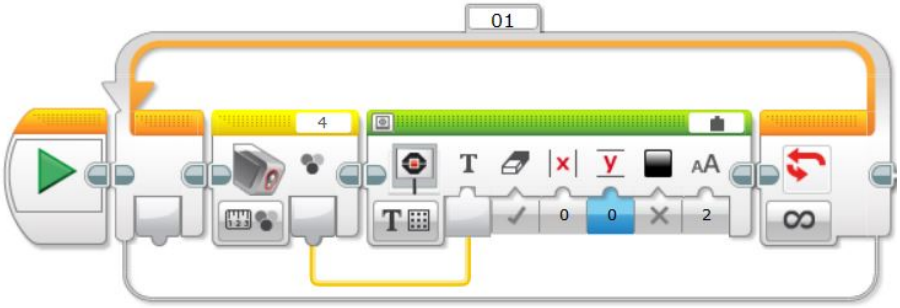
Az érzékelő (citromsárga) blokkok között találjuk azokat a blokkokat, melyek segítségével le tudjuk kérdezni egyes érzékelők értékét.



46. ábra. Érzékelő blokkok

Felmerül persze a kérdés, hogy de erre nekünk mi szükségünk is van... Ezen értékek felhasználásával azonban nagyon sok érdekes feladatot tudunk megvalósítani! De hogyan is tudunk hozzáférni az érzékelők értékeihez? Erre szolgálnak például a 46. ábrán látható blokkok. A felső segítségével a színérzékelő értékét, míg az alsóval az érintésérzékelő értékét tudjuk lekérdezni. Mit is kezdjünk ezekkel az értékekkel? Például ki tudjuk írni a robot képernyőjére. Ehhez egy újabb beállítással kell megismerkednünk a kijelző blokkon! Használjuk a szöveg kiírására alkalmas Grid módot. A jobb felső sarokban szöveg beírása helyett kattintsunk a „Wired" lehetőségre. Ekkor egy új beállítási lehetőség jelenik meg a blokk paraméterei között. Ehhez tudjuk csatlakoztatni a most megismert érzékelő blokkot a 47. ábrán látható módon.

Fontos, hogy ciklusba tegyük a programunkat, hogyha folyamatosan szeretnénk vizsgálni egy érzékelő értékének változását! Ha ezt nem tennénk meg akkor a program indításakor beolvasott érték jelenne meg a képernyőn, mely nem frissülne a soha az újbóli vizsgálat hiányában!



47. ábra. Színérzékelő értékének megjelenítése a robot képernyőjén

3.6.4. Feladatok

Valószínűleg ezen a szakköri alkalmon már nem marad sok idő ezekre a feladatokra. Amit nem sikerül megvalósítani azt a következő alkalommal folytatjuk!

1. Készíts olyan programot, melynek eredményeképp a robotod akkora sugarú körlapot rajzol a kijelző közepére, amilyen távol észlel magától valamit távolságérzékelőjével.

2. Fejleszd tovább a már korábban megvalósított radar programodat! Miközben a robot körbe-körbe pörög folyamatosan írja ki a képernyőre az általa mért távolság értékét!

3. Valósítsd meg a következő programot: a robot haladjon egyenesen, a sebességét az ultrahangos távolságérzékelő értéke határozza meg!

   Nagyon kicsi távolság esetén (kisebb mint 2 cm) azt tapasztaljuk, hogy az ultrahangos távolságérzékelő a maximum értékét, 255 cm-t mutat. Ezt a jelenséget az ultrahangos távolságérzékelő működési elvével magyarázhatjuk. Az egyik „szemén" ultrahangot bocsát ki, a másik pedig a visszaverődő hangot észleli. A távolságot abból számolja, hogy mennyi idő telt el a kiküldés és az észlelés között. Amikor az akadály nagyon közel van, akkor a kibocsátott ultrahang nem tud visszaverődve eljutni a másik „szembe". Mivel nem jön vissza a kibocsátott hang, a robot nem képes ezt megkülönböztetni attól az esettől, amikor az akadály olyan messze van, hogy már nem verődik vissza érzékelhető erősségben.

4. *Robotod írja ki mindhárom tanult érzékelő értékét egymás alá a következő elrendezésben:*
   érzékelő neve: érték

3.7. 7. alkalom

Ráhangolódás

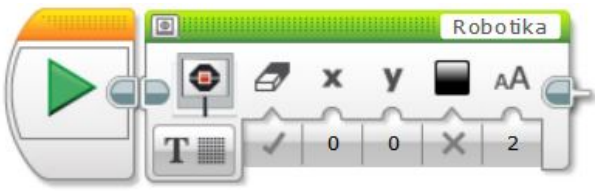
- *A kijelző blokk „Reset Screen” módja arra szolgál, hogy letöröljük robotunk képernyőjéről a robot menüjét és egy tiszta képernyőt kapjunk.*
  - *Hamis* — *Igaz*

- *Ha szöveg kiírásánál a szöveg túl hosszú és nem fér ki a robot képernyőjére, akkor...*
  - *a szöveg összetömörítve, kisebb betűkkel jelenik meg a képernyőn, hogy mégis elférjen.*
  - *nem jelenik meg semmi a képernyőn.*
  - *a szövegből annyi látszik a képernyőn, amennyi kifért.*
  - *a program automatikus elválasztást alkalmaz a szövegen, így jelenítve meg az egészet.*

- *Mi a különbség a szövegkiírás két módja a „Grid” és a „Pixels” között?*
  - *A megjelenítésben semmi, csak a blokk-paramétereik és az előnézeti képek között van különbség.*
  - *„Grid” esetén a kijelzőn téglalaprács látszik, így jelenik meg a szöveg.*
  - *„Pixels” esetén a kijelzőn négyzetrács látszik, így jelenik meg a szöveg.*
  - *A „Grid” móddal nem szöveget, hanem alakzatot tudunk megjeleníteni.*

- *Milyen színű blokkok között találjuk a képernyőre írás/rajzolás(Display) blokkot?*
  - *piros*
  - *narancssárga*
  - *citromsárga*
  - *zöld*

- *Mi történik, ha az alábbi programot szeretnénk futtatni a roboton?*
  - *A program azonnal leáll, látszólag semmi nem történik.*
  - *Kiírja a képernyőre, hogy „Robotika”.*
  - *Villogni kezd a képernyőn a „Robotika” felirat.*
  - *Hibás a program, ezért el sem indul.*



- *Ha az x=0, y=0, koordinátára írunk valamit, akkor hova kerül a szöveg a kijelzőn?*
  - *Bal felülre*
  - *Jobb felülre*
  - *Bal alulra*
  - *Jobb alulra*
- *Ki tudjuk írni az érzékelők értékét a képernyőre?*
  - *Igen*
  - *Nem*

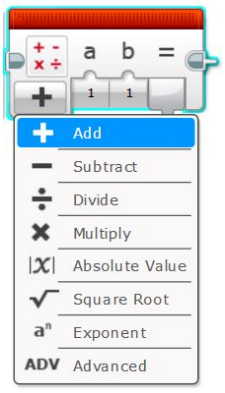
3.7.1. Érzékelők értékét felhasználó feladatok

Az előző alkalomról maradt feladatok megvalósítása.

3.7.2. Matematikai műveletek

Az adatműveletek (piros) blokkok között találjuk a matematikai műveleteket megvalósító blokkokat. Ezek közül most csak eggyel fogunk megismerkedni, melyre szükségünk lesz későbbi feladataink megoldása során!

A 48. ábrán látható blokk segítségével számos matematikai műveltet tudunk elvégezni. Ezek közül most az összeadás, kivonás és szorzás műveletekkel fogunk megismerkedni.



48. ábra. Matematikai műveletek

Érdeklődéstől függően lehet több opciót is megtanítani, kipróbálni!

Hogyha kiválasztjuk az Add, vagyis összeadás módot, akkor látjuk, hogy két paraméter beállítására van lehetőségünk. Az *a* és *b* számok értékeit meghatározva megkapjuk azok összegét. De mi történik az összeggel? Hogy tudjuk elérni azt? Ahogy előző alkalommal az érzékelő blokkok esetén, itt is össze kell kötnünk valamivel a rendelkezésünkre álló értéket! Például ki tudjuk írni robotunk képernyőjére, ahogy ezt már korábban is tettük!

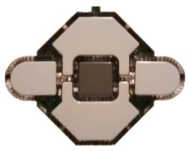
3.7.3. Feladatok

1. Próbáld ki tetszőleges a és b értékekkel az összeadás, kivonás, szorzás műveleteket! Írd ki eredményed a robot képernyőjére!

3.7.4. A tégla gombjainak használata

Eddig a téglán lévő gombokat csak a menüben való navigáláshoz, illetve a robot bekapcsolásához és program elindításához használtuk. Azonban ezek a gombok másra is alkalmasak, programozhatjuk is őket! Ezzel a lehetőséggel fogunk az alábbiakban közelebbről megismerkedni.

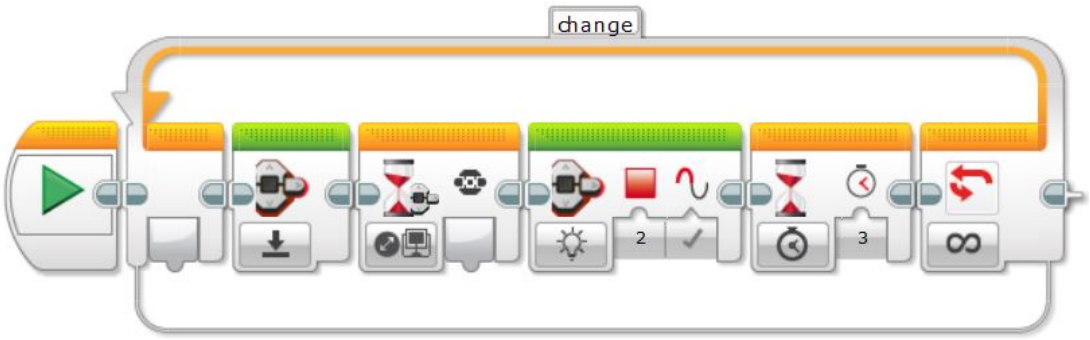
A tégla tetején található 5 gombot (bal, jobb, középső, alsó, felső) érzékelőként is felfoghatjuk! Közvetlenül a képernyő alatt található gomb, mely többek között a roboton futó programok leállítására szolgál nem tartozik ezen „érzékelő/programozható” gombok közé! Ennek kitüntetett feladata van! Bármely, a 49. ábrán látható gomb megnyomásához feltételt köthetünk programunkban, vagy akár felhasználhatjuk a gomb értékét feladataink kreatív megoldásához! Működése leginkább az érintésérzékelőhöz hasonlít, hiszen ezen gomboknak is három állapotát különböztetjük meg, csakúgy mint az előbb említett érzékelő esetén!



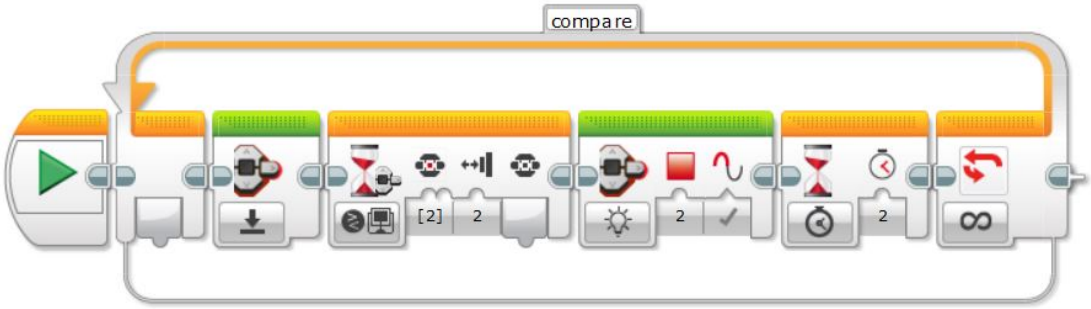
49. ábra. A tégla programozható gombjai

Nagyon fontos, hogy a gyerekek megértsék ezt a három állapotot, hogy jól meg tudják különböztetni azokat! Véleményem szerint nem baj, hogyha ennek szemléltetésére ennél a résznél is szánunk időt! A érintésérzékelő már régen volt (főleg, ha heti egy szakköri alkalom van), így nem árt az ismétlés!

A három állapot közötti különbség szemléltetésére szolgálnak az 50. és 51. ábrán látható kódok. Ezek segítségével jobban meg lehet érteni a különbséget.



50. ábra. „Pressed” és „Bumped” módok különbségének szemléltetése



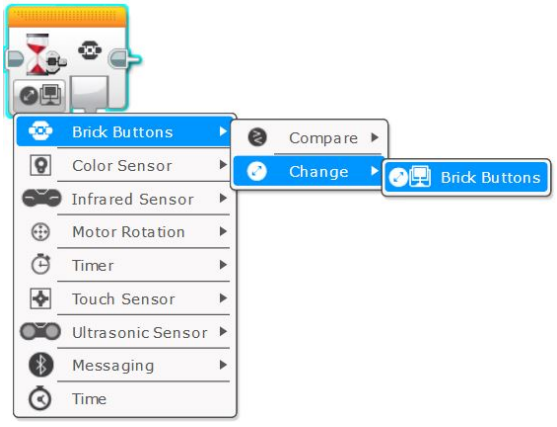
51. ábra. Hogy működik a „Bumped” állapot

Ezt a programot fel lehet használni olyan módon, hogy kivetítjük a gyerekeknek, ők pedig megírják saját gépeiken, majd elmondjuk, hogyan érdemes a robotokon tesztelni. A másik lehetőség, hogy a tanári gép képernyője kivetítve látszik, és a gyerekek az éppen készülő programot másolják le. Ekkor programozás közben meg is beszélhetjük miért így írjuk a programot. Ezt követően mindenki kipróbálja a roboton működés közben is!

A 50. ábrán lévő kódot a következő módon érdemes tesztelni. Amikor a program először várakozik egy gomb megnyomására, akkor benyomjuk az egyik (tetszőleges programozható) gombot a tégla tetején és nyomva tartjuk egészen a következő „ellenőrzésig". Ekkor elengedjük az eddig nyomva tartott gombot. Harmadik vizsgálatkor csak nyomjunk meg egy gombot, tehát megnyomás után közvetlenül engedjük is el. Így rögtön látszik a blokk működési elve, illetve a három különböző gomb-állapot.
Most nézzük meg hogy érdemes a 51. ábrán lévő kódot tesztelnünk. Első vizsgálatkor hagyjuk a tégla egy tetszőleges programozható gombját sok ideig benyomva, csak később engedjük el. Második vizsgálatkor a benyomás után közvetlen engedjük el. Ezt egy roboton kipróbálva már látszik is, hogy mit jelent „egy bumped"!

Gombok használata a várj blokk segítségével

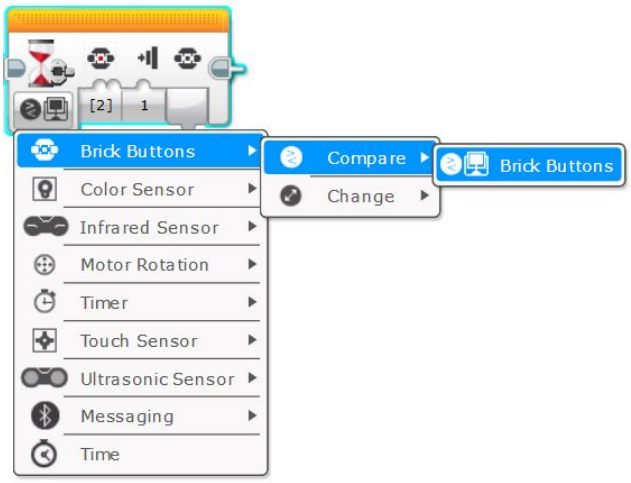
A szoftverben több, a gombok programozásához szükséges blokk is rendelkezésünkre áll. Ahogy a korábban tanult érzékelők esetén is láttuk nem csak a szenzor-blokkok között találunk érzékelőhöz kötődő funkciót, hanem például a várj-blokk esetén is. Így van ez a most megismert programozható gombokkal is! Nézzük meg milyen lehetőségeink vannak a gombokkal kapcsolatban, hogyha a szoftver várj-blokkját használjuk!



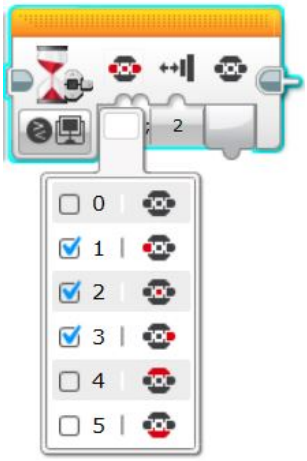
52. ábra

Először nézzük meg mire szolgál a várakozás blokkon belüli „Brick Buttons-Change" mód. Ahogy az 52. ábrán látható ezt a módot választva a blokknak egyetlen kimeneti paramétere lesz, rajta semmit nem tudunk beállítani. Ez a blokk arra szolgál, hogy programunk addig várakozzon, amíg egy tetszőleges programozható gombjának állapota megváltozik. Ha a programunk indításakor egy gomb sem volt benyomva, akkor bármelyiket megnyomjuk az öt közül, programunk tovább megy és elvégzi a várakozás blokk utáni részt. Ha viszont a program futtatásának kezdete előtt már be volt nyomva a tégla egy gombja, akkor a várakozás blokk utáni rész a lenyomva tartott gomb felengedésével le fog futni, hiszen ekkor a gomb állapota megváltozott. Tehát ez a mód a tégla gombjainak változását vizsgálja. Nem különbözteti meg az öt gombot egymástól és a három gombállapotot sem.
Abban az esetben, ha a várj blokk „Brick Buttons-Compare" módját választjuk már van lehetőségünk bemeneti paraméterek megadására, ahogy ezt az 53. ábra is mutatja. Itt mi dönthetjük el, hogy programunk melyik gomb(ok) állapotának megváltozására menjen tovább és azt is, hogy milyen állapotváltozásra reagáljon. Tehát például ha a bal, középső és jobb gombok megnyomására szeretnénk várakozni és az a feltételünk, hogy benyomásra még ne menjen tovább a programunk, csak akkor ha az adott gombot ki is engedtük, akkor az első paraméternél kiválasztjuk az 1, 2, 3 opciókat. A második

paraméternél pedig a 2-es opciót, mely a „Bumped” állapotot jelöli.



53. ábra. Compare mód



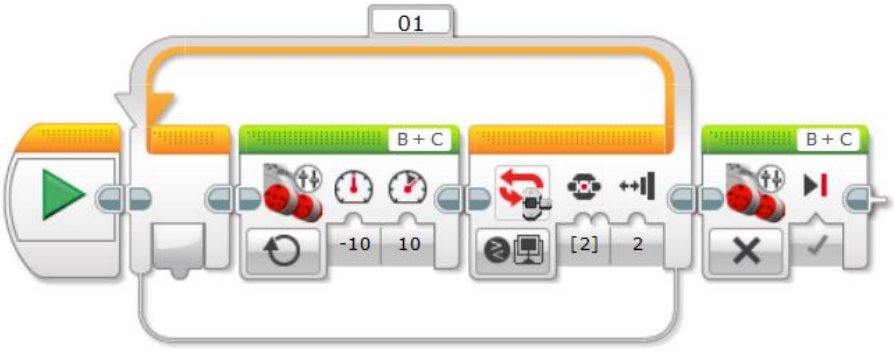
54. ábra. Measure mód

A tégla gombjainak megfeleltetett számokat nem is kell megjegyeznünk, hiszen a blokk első paraméterére kattintva látjuk, hogy melyik gomb melyik számnak van megfeleltetve.

Gombok a ciklusfeltételben és elágazás szerkezetben

A narancssárga blokkok (vezérlési szerkezetek) között azonban nem csak a várj-blokk esetén van lehetőségünk a gombokra feltételt szabni, hanem például a ciklus és az elágazás szerkezeteknél is. Vizsgáljuk meg ezeket az opciókat.

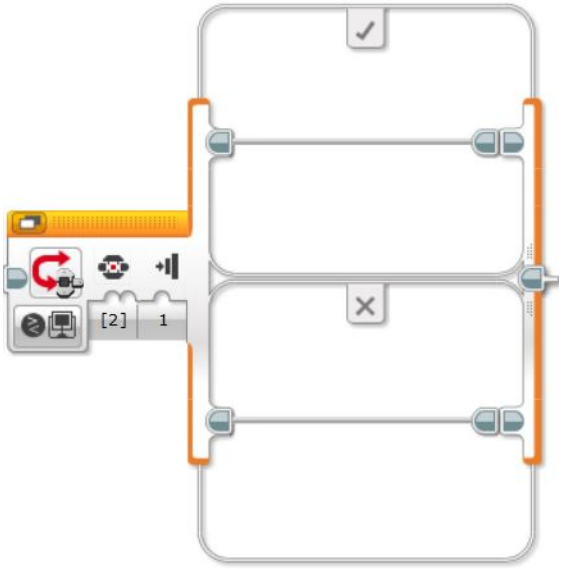
Hogyha ciklusunk kilépési feltételének adjuk a gombok állapotának megváltozását, akkor azt az 55. ábrán látható módon tehetjük meg. Ekkor kiválaszthatjuk melyik gombra és annak milyen állapotára szeretnénk kilépni a ciklusból.



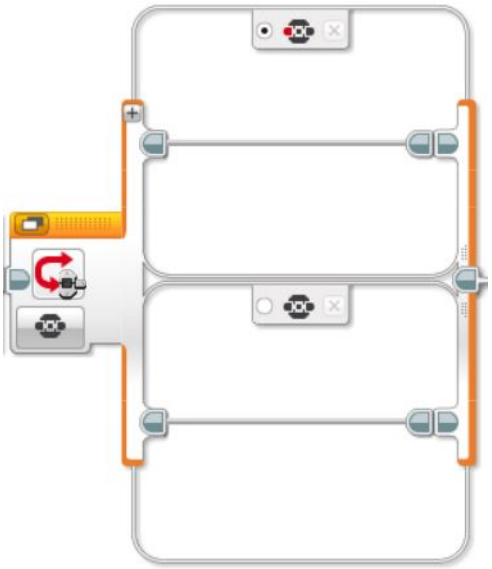
55. ábra. Gombnyomás, mint kilépési feltétel

Az elágazás szerkezetben kétféle módon is használhatjuk a gombokat feltétel megadására. Az egyik lehetőségünk az 56. ábrán látható. Ehhez a módválasztóban a „Brick Buttons-Compare” lehetőséget kell kiválasztanunk. Ez a szerkezet arra szolgál, hogy feltételt tudjunk szabni arra, hogy robotunk mit tegyen például egy benyomott vagy kiengedett gomb esetén.

Ha a módválasztóban a másik opciót választjuk, vagyis a „Brick Buttons-Measure” lehetőséget, akkor az 57. ábrán látható szerkezetet kapjuk. Ez nagyon hasznos abban az esetben, ha azt szeretnénk, hogy robotunk különböző gombok megnyomására különbözően reagáljon.



56. ábra. Compare mód



57. ábra. Measure mód

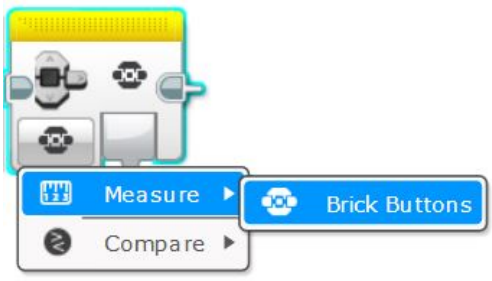
Gombok használata az érzékelő blokk segítségével

Tekintsük át milyen lehetőségeink vannak, hogyha az érzékelő blokkok közül választjuk ki a gombokat vezérlő blokkokat.

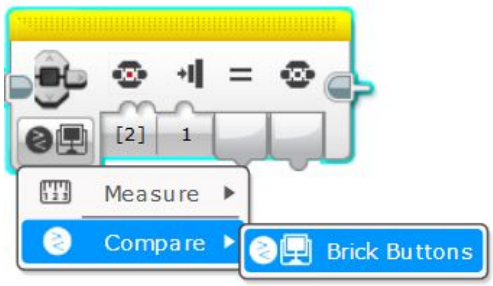
Az 58. ábrán látható Measure módot választva megkapjuk szám-kimenetként a téglán éppen benyomott gomb azonosítóját (vagyis a neki megfeleltetett számot).

Ezt az értéket „vezetéken" beköthetjük olyan blokk paraméterébe, mely számot vár.

Az 59. ábrán látható Compare módot választva pedig egy logikai és egy szám értéket kapunk. A szám a logikai értéktől függ. 0, hogyha a logikai érték hamis, 1, hogyha igaz. A logikai kimenet értéke pedig attól függ, hogy a bemeneti paramétereken általunk megfogalmazott feltétel igaz vagy hamis a gombok éppeni állapotára. Tehát a 59. ábrán látható blokk értéke pontosan akkor igaz, hogyha a középső gomb be van nyomva a téglán. Ha kiengedjük ez az érték hamisra változik. Fontos megjegyeznünk, hogy több gomb kiválasztása esetén a feltételek között „ÉS" szerepel, tehát a logikai érték csak akkor ad igazat, ha az adott feltétel az összes kiválasztott gombra teljesül.



58. ábra



59. ábra

3.7.5. Feladatok

1. Írj programot, melynek hatására robotod a középső gomb megnyomásakor kirajzolja a képernyőre az EV3 ikont. A bal gomb megnyomására balra mutató nyilat rajzol és így tovább, minden gomb megnyomására egy felé mutató nyilat jelenít meg!

2. Készíts programot, melyben robotod a középső gomb megnyomására kiad egy hangot röviden (ez jelzi, hogy elkezdett futni a program)! Ezek után úgy tudod mozgatni a robotot, hogyha megnyomod a felfelé gombot, robotod előre megy 2 mp-ig, a lefelé gomb hatására pedig tolat 2 mp-ig. A balra gomb lenyomására balra fordul 90°-ot, a jobb gombra pedig jobbra ugyanennyit.

3. Fejleszd tovább a Radar programot(alap feladatot, ne a már továbbfejlesztettet!), melyet már az előző alkalmak egyikén elkészítettél! Építsd programodba azt a funkciót, melynek hatására a tégla tetején lévő bármely gombot megnyomva robotod képernyőjén egy Stop tábla rajza jelenik meg, majd 1 másodperc múlva kilép a programból!

4. Készíts olyan programot, melynek során a tégla tetején lévő fölső gombot nyomva tartva a led narancssárgán, míg az alsó gombot nyomva tartva pirosan világít!
   Ha sikerült megoldanod a feladatot, akkor keress más jó megoldást is! Programodat próbáld a gombok értékének felhasználásával megvalósítani!

5. Valósítsd meg, hogy robotod képernyőjén egyre növekvő sugarú körök jelenjenek meg, ahogy a tégla gombjait sorban megnyomod! A körök a következő gombnyomás sorrendre növekedjenek: bal, középső, jobb, felső, alsó!
   Gondoskodj róla, hogy robot a program futása alatt ne világítson és a körök megjelenésekor semmi más ne legyen a képernyőn!

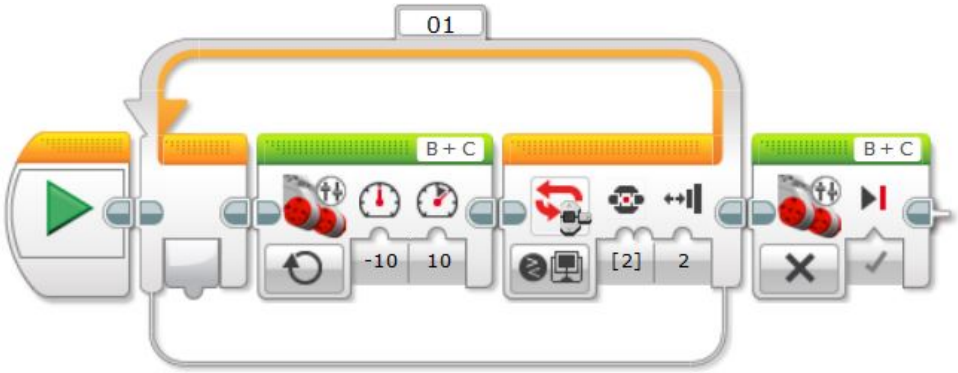
3.8. 8. alkalom

Ráhangolódás

- *A Stop-program blokk mit csinál?*
  - *Az adott szálat leállítja a programban.*
  - *Kikapcsolja a robotot.*
  - *Kiírja a képernyőre, hogy „STOP”.*
  - *Megállítja a teljes program futását.*
- *A tégla programozható gombjainak száma...*
  - *3*
  - *4*
  - *5*
  - *6*
- *A képen látható blokk hatására...*
  - *programunk várakozik arra, hogy minden gomb el legyen engedve a roboton.*
  - *programunk várakozik arra, hogy az öt közül valamelyik gombot benyomjuk.*
  - *programunk várakozik arra, hogy az öt közül valamelyik gombot felengedjük.*
  - *programunk várakozik arra, hogy az öt közül valamelyik gombnak megváltozzon az állapota.*

- *A tégla gombjainak két állapotát különböztetjük meg, a „Pressed” (benyomva) és „Released” (kiengedve) állapotokat.*
  - *Igaz*
  - *Hamis*

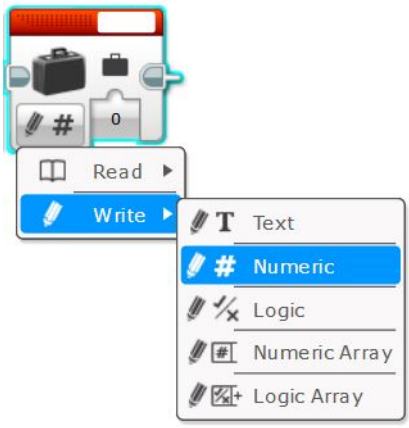
- *Ha elágazást használunk ahhoz, hogy a gombokhoz különböző eseményeket kapcsoljunk, akkor maximum ??? ágat tudunk létrehozni.*
  - *4*
  - *5*
  - *6*
  - *7*
- *Az alábbi program hatására robotunk addig forog körbe saját tengelye körül, amíg...*
  - *ki nem engedjük a középső gombot.*
  - *meg nem nyomunk a roboton egy tetszőleges gombot.*
  - *be nem nyomjuk a középső gombot.*
  - *amíg nem észleli robotunk a középső gomb benyomásának, majd kiengedésének együttesét.*



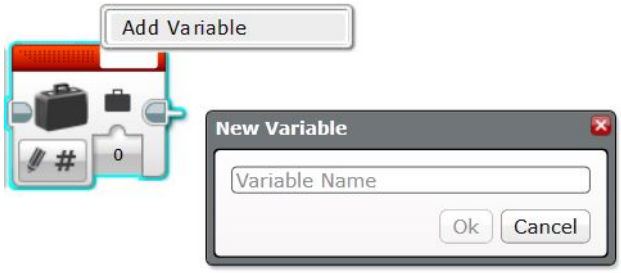
3.8.1. Változók használata

A szoftverben az adatművelet (piros) blokkok között a sorban elsőként találjuk a változók kezeléséhez szükséges blokkot. A változó egy olyan hely az okos tégla memóriájában, ahol adatokat tud tárolni. Akkor érdemes egy változót létrehozni, ha olyan értéket szeretnénk benne tárolni, melyhez később hozzá szeretnénk férni.
Minden változónak van egy típusa és egy neve. A név már egyértelműen azonosítja a változót, mivel különböző típusú, de megegyező nevű változók létrehozására nincs lehetőségünk, ezt a szoftver nem engedi!

Érdemes rövid és beszédes nevet adni változóinknak, hogy később könnyen felidézhessük mit is tároltunk benne.



60. ábra. Változó lehetséges típusai



61. ábra. Változó létrehozása

Öt különböző típus közül választhatunk, amikor létrehozunk egy változót. Most részletesen csak a szám típusúval fogunk megismerkedni, de működési elvét tekintve a többi típus sem más! Változót a blokk jobb felső sarkában lévő fehér téglalpra kattintva és ott az „Add Variable" lehetőséget választva tudunk létrehozni. Az így létrehozott változó látszani fog a projektünkön belül az összes programban, de értékét nem tudjuk máshol felhasználni.

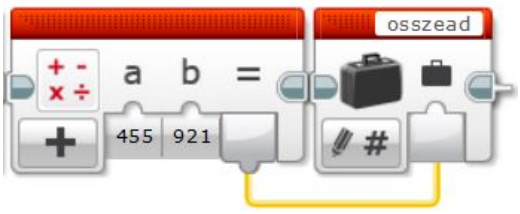
> *Tehát ha például összeszámoltuk egy program segítségével, hogy a futása alatt hányszor nyomtuk be az érintésérzékelőt, és ezt az értéket eltároltuk egy változóban, akkor ez az érték pontosan addig elérhető, amíg a program futása le nem áll! Utána már nem tárolja az előzőleg megkapott értéket.*

Egy változó értéke megváltozhat a program futása során. Ekkor az előző érték felülíródik és az új érték lesz a változónkban. De nézzük meg hogy is tudunk új értéket adni, vagy éppen hozzáférni változónk értékéhez.

Értékadás

Értékadáshoz a változó módválasztó gombjára kattintva a „Write" lehetőséget kell kiválasztanunk. Ahogy a 60. ábrán is látható ebben a módban egy bementi paraméter áll rendelkezésünkre. Itt értéket tudunk adni változónknak (jelen esetben egy számot). Kezdetben minden szám típusú változó értéke nulla.

Tehát, hogyha 0 kezdőértékkel szeretnénk változónkat használni, akkor nem szükséges a változó létrehozásakor értékét 0-val felülírni! (Ha mégis kiírjuk programunk talán más számára átláthatóbb lesz.) Értéket tudunk közvetlenül adni olyan módon, hogy begépelünk a 60. ábrán látható 0 helyére egy számot. A másik lehetőségünk értékadásra, hogy egy szám típusú értéket bekötünk bemeneti paraméterként. Ekkor változónk azt az értéket fogja megkapni, melyet „vezetéken" bekötöttünk. A 62. ábrán például egy összeadás művelet végeredményét adtuk értékül „osszead" nevű, szám típusú változónknak.



62. ábra

Változó értékének kiolvasása

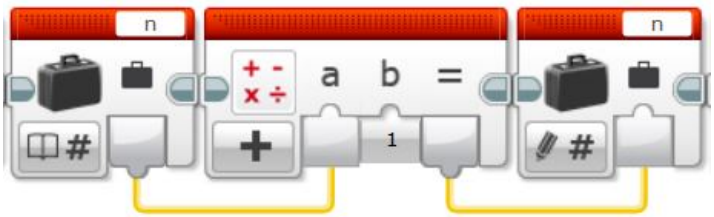
Nagyon fontos művelet változónk értékének kiolvasása is, hiszen azért tároltuk el az értéket, mert később is szükségünk lesz rá. Felhasználhatjuk például arra, hogy kiírjuk a robot képernyőjére. Azonban, ahogy már korábban is láttuk ehhez önmagában nem lenne szükség változó használatára, mivel a képernyőre kiírás közvetlenül is megtörténhet.

63. ábra

Változónk értékének kiolvasása akkor nyer értelmet, amikor értéke a program futása közben valaminek hatására megváltozik. Ha az így nyert, új értéket szeretnénk kiíratni a képernyőre, akkor már valóban szükségünk van változó bevezetésére. Az érték felhasználásához a változó módválasztó gombjára kattintva a „Read" lehetőséget kell kiválasztanunk.

A blokk jobb felső sarkára kattintva ki kell választanunk, hogy melyik változó értékéhez szeretnénk hozzáférni. A 63. ábrán látható, hogy bemeneti paraméter nem áll rendelkezésünkre. Az egyetlen kimeneti paraméter a változónk értékét adja vissza. Ezt „vezetéken" hozzáköthetjük bármilyen blokk szám típusú beviteli paraméteréhez, így felhasználva változónk aktuális értékét.

Gyakran előfordul feladatok megoldása során, hogy változónk értékét akarjuk növelni valamilyen feltétel teljesülésekor. Ekkor a 64. ábrán látható szerkezetet szokás alkalmazni. Itt az n változó értékéhez hozzáadunk egyet, majd az így kapott értékkel felülírjuk változónk értékét. Ha legközelebb elkérjük n értékét, akkor az előzőhöz képest, már eggyel nagyobb számot fogunk kapni.



64. ábra.

A gyerekeknek tapasztalataim szerint néha nehézséget okoz a változók működésének megértése. Valóban sokkal elvontabb problémával állunk szemben az eddigieknél! Fontosnak tartom azonban, hogy megpróbáljuk életközeli példával közelebb hozni ezt a részt hozzájuk, hiszen a változók megértése fontos komolyabb programozási nyelvekben is. Tehát ez a későbbiekben is javukra válhat. Nekem bevált az a módszer, ha valódi iskolatáskával szemléltetem a helyzetet. Elnevezzük az iskolatáskát, és csak egyféle dolgot pakolunk bele. Lehet ez alma, tankönyv... Kezdetben egyet sem teszünk bele. Majd megnézzük mennyi alma van a táskánkban. (Kiolvassuk a változó értékét) Beleteszünk két almát, amit le kell adminisztrálnunk, vagyis fel kell jegyeznünk, hogy mostmár két alma van a táskánkban! (Felülírjuk a változó értékét.)
Ezzel a szemlélettel sokkal könnyebb lesz megérteniük. Ha így sem megy, akkor alma és tankönyv helyett próbálkozzunk csokival, úgy már biztos menni fog!

3.8.2. Feladatok

1. Írj számsorozatokat robotod képernyőjére! A számok az érintésérzékelő megnyomásával változzanak a következő módon:

   Gondoskodj arról, hogy egy megnyomásra tényleg csak egyszer végezze el robotod az adott műveltet!

   (a) érzékelő megnyomására + 1

   (b) érzékelő megnyomására * 2

   (c) csak páratlan számok

2. Írj programot, mely kiírja a robot képernyőjére, hogy hányszor nyomtuk meg a tégla középső gombját!

3. Írj programot, mely megszámolja, hogy hány zöld vonal van a pályán és kiírja a képernyőre a következőképpen:
   ZOLD: ...db

4. Írj programot, mely megszámolja, hogy hány piros, fekete, zöld vonal van a pályán és kiírja a képernyőre a következőképpen:
   PIROS: ...db
   FEKETE: ...db
   ZOLD: ...db

5. Készíts egy robot mérőszalagot! Mérje fel a robot, hogy előtte és mögötte az éppeni helyzetéhez viszonyítva pontosan mennyi a távolságösszeg. Vagyis mekkora a távolság az előtte és a mögötte lévő két akadály között!

Hívjuk fel a diákok figyelmét arra, hogy a következő alkalommal már nem fogunk új ismeretet elsajátítani, hanem az eddig tanultak ismétlése következik! Próbálják meg következő alkalomra összegyűjteni kérdéseiket, melyek felmerültek az eddig tanultakkal kapcsolatban!

3.9. 9. alkalom

Ráhangolódás

- *Milyen új blokkot használtunk ahhoz, hogy megszámoljuk például az érintésérzékelő megnyomásait?*
  - *Konstans blokkot*
  - *Változó blokkot*
  - *Ütköző blokkot*
  - *Motor blokkot*
- *Mi a jele a változó blokknak?*
  - *Tálca*
  - *Kosár*
  - *Vödör*
  - *Táska*
- *A változó blokk arra jó, hogy...*
  - *robotunk folyamatosan tudja változtatni sebességét.*
  - *robotunk folyamatosan tudja változtatni a téglán lévő led színét.*
  - *eltároljunk segítségével egy értéket, melyre később még szükségünk lesz.*
  - *eltároljunk segítségével blokkokat, hogy máskor könnyebben elő tudjuk állítani őket.*
- *Le kell törölni a program elején a robot képernyőjét mielőtt kiíratjuk a változó értékét?*
  - *Nem*
  - *Igen*
- *Egy változóban eltárolt érték a program futása során...*
  - *megváltozhat, az új értékek mindig felülírják a régi értékeit.*
  - *nem változhat, a program futása végén is ugyanaz lesz az értéke.*
  - *megváltozhat, de az előző értékei sem vesznek el, egy tömbben tárolódnak el sorban.*
  - *megváltozhat, de az előző értékei sem vesznek el, azok mind egy-egy változóban lesznek tárolva.*
- *Változóban csak számot tudunk tárolni.*
  - *Hamis*
  - *Igaz*

3.9.1. Gyerekek kérdéseinek megválaszolása

Bíztassuk a gyerekeket, hogy most kérdezzenek meg minden olyan részt az eddig tanultakkal kapcsolatban, ami nem volt teljesen érthető! Ha vannak kérdések, akkor azt az egész csoporttal beszéljük meg, ne csak négyszemközt a kérdező diákkal, hiszen másnak is javára válik az ismétlés! Hogyha nem akad több kérdés, akkor kezdjünk bele a Kahoot tesztbe!

3.9.2. EV3 akadálypálya

A diákok párokban fognak dolgozni a következő feladaton. Egy robot akadálypályát kell teljesíteniük kis robotjaikkal a szakköri alkalom végéig!

A párok fontos, hogy egymást segítve, együttműködve tudjanak dolgozni! Mindkettejüknek programozni kell, de nem muszáj ugyanazon feladatnak közösen megalkotniuk a megoldását, fel is oszthatják maguk között a feladatokat!

A pálya épülhet bármilyen nagy méretű kartonból, kemény egyenes alapból! Kötelező elemek a start és a cél, de az egyéb elemek szabadon elhelyezhetőek a pályán (*arra figyeljünk, hogy a robot mindenhol elférjen, ahol feladatot kell végrehajtania*)!
Ismertessük az akadálypálya szabályait!

- A pályán folyamatosan kipróbálhatjátok készülő programjaitokat, de fontos, hogy mindenkinek lehetősége legyen a kipróbálásra (ne foglalja be senki huzamosabb időre) és ne lépjetek a pályára!
- A robotokat a START feliratú mezőről kell indítani és a CÉL feliratú mezőbe kell megérkezniük végül az összes feladat teljesítése után!
- A feladatokat robototok tetszőleges sorrendben hajthatja végre!
- Ne egy hatalmas programot alkossatok, hanem feladatrészenként készítsetek új programot egy projekten belül! A programokat beszédesen nevezzétek el, hogy a kipróbálás és a végső megoldás során könnyen megtaláljátok az éppen szükséges állományt.
- Két külön gépen dolgozhattok! Végül mindkettőtök gépéről töltsétek a robotra az elkészült programokat!
- A feladatok teljesítése között robototokat motorok segítségével mozgassátok a következő teljesítendő akadályig! Fontos, hogy a pályát a robot felemelése és áthelyezése nélkül kell teljesíteni!
- Két akadály között újabb program elindításához hagyjátok robototokat a pályán, helyben indítsátok el rajta a következő programot!

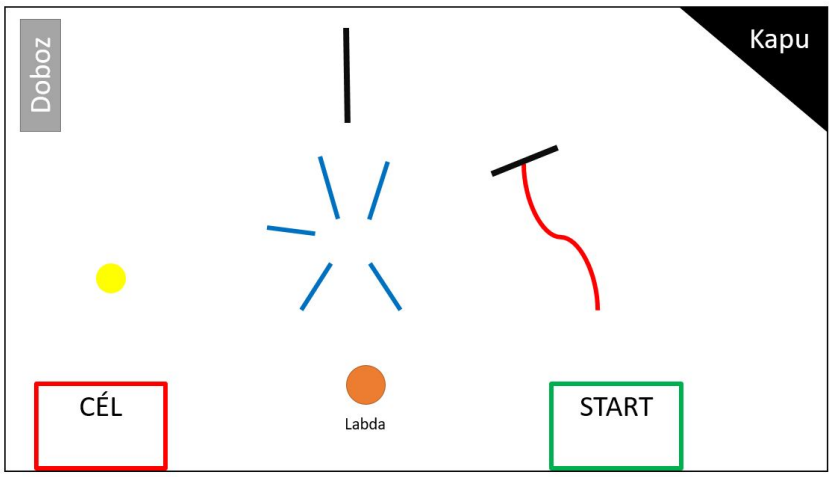
Ha szeret versenyezni a csoportunk, akkor ebből az akadálypályából hirdethetünk „versenyt” is és akkor egy pontozási útmutató meghatározásával tudjuk értékelni az elkészült megoldásokat! Ekkor járhat pontlevonás arra, ha egy akadályt nem tudott teljesíteni a robot, vagy csak úgy tudott továbbmenni, hogy ahhoz át kellett helyezni a pályán belül máshová!

Fontos, hogy a szakkör végén maradjon idő arra, hogy minden páros bemutassa munkáját, kis robotja tudja-e teljesíteni az akadályokat!

A következőkben leírt feladatok opcionálisak! Fontos, hogy a csoport tudásához mérten határozzuk meg a feladatokat! Ha túl kevés feladatot adunk, akkor hamar unatkozni kezdenek. Ha túl sok a feladat, akkor viszont megvan a veszélye, hogy nem lesz sikerélményük, mivel nem végeznek a feladatokkal a szakköri alkalom végéig!

Feladatok

- Az akadálypályát robotod a Start feliratú mezőről kezdje, a zöld vonal mögül! Innen a középső gomb megnyomására induljon!
- Robotod menjen végig a piros hullámvonalon!
- Számolja meg az összes kék vonalat a pályán és számukat írja a képernyőre: „KEK: db”
- Annyiszor csinálja meg a következő feladatot, ahány kék vonal van a pályán:
  Ingázás: Doboz és fekete vonal között ingázzon fordulás nélkül oda-vissza! Az egyik irányban a téglán lévő led pirosan, a másik irányban zölden világítson folyamatosan!
- Juttassa a pályán található labdát a kapuba! Miközben ezt a feladatot végzi képernyőjén jelenjen meg valamilyen ábra (például vicces szemek)!
- Ha sárga színt észlel, rajzoljon ki egy virágot úgy, hogy ez a sárga rész legyen a virág közepe!
- Érjen be a robot a célba, melyet piros szín jelöl. Menjen át a vonalon, ne lógjon be a robot a pályára! Írja a képernyőre: TELJESITETTEM! Adjon ki valamilyen hangot, mely jelzi örül a sikernek!



65. ábra. Az akadálypálya egy lehetséges felépítése

A pályán az elmozdítható akadályok helyét jelöljük például szigetelő szalagból készített X-el, hogy ha egy robot a feladat elvégzése során eltolja/felborítja azt, akkor ugyanoda tudjuk visszatenni. Ez azért nagyon fontos, mert a robotok a feladatok elvégzése között motormozgásokat használnak. Ha az akadály nem az eredeti helyére kerül vissza, akkor a robot nem feltétlen fognak helyesen odatalálni a következő akadályig.

3.10. 10. alkalom

3.10.1. Saját projekt kidolgozása - terv

A diákok feladata ezen és az ezt követő szakköri alkalmon egy saját projekt megvalósítása lesz. Ehhez először is párokat kell alkotniuk. A terv elkészítése, kitalálása, az építés és programozás is közös feladat, melyet akár fel is oszthatnak egymás között.

Fontos, hogy a diákok meg tudjanak egyezni a projektjük témájával kapcsolatban, hogy közösen, kreativitásukat használva alkossanak egy tervet. Ha nincs ötlete valamelyik párosnak először bíztassuk őket. Csak akkor álljunk elő segítőkész ötletünkkel számukra, ha látjuk különben nagyon lemaradnak a többiektől!

Megvalósítható projekt ötletek:

1. Hercegnő és béka esete a mesében - daru építése és programozása
2. Három kismalac a mesében - házak készítése, propeller a robotra, ami elfújja a házat
3. Színszkennelő robot

3.10.2. Építés, átépítés és programozás

A diákok ebben a részben saját kezet kapnak. A tervük alapján elkezdik a robot átépítését, ha kell segédtárgyak létrehozását. Minden párosnak rendelkezésére áll egy Lego Mindstorms EV3 teljes készlet, így szabadon építkezhetnek.

Tanárként fontos arra figyelni, hogy a diákok ne merüljenek el annyira a legozásban, hogy ne maradjon idejük a robot programozására. Az építés is számos készséget fejleszt, azonban szakkörünk célja a robotok programozásának elsajátítása.

A programozás során felhasználhatják az összes, korábbi alkalmakon megírt programjukat, ötletet meríthetnek azokból.

A diákok kérdéseire mi is válaszolhatunk, azonban ha olyat kérdeznek, melyet egy korábbi alkalommal már tanultunk, akkor jobban tesszük, ha csak rávezetjük őket a megoldásra, vagy eláruljuk, hogy melyik szakkörön volt hasonlóról szó. Ezzel mélyebb tudásra tehetnek szert, mintha rögtön a megoldással állnánk elő. Fontos, hogy a diákok gondolkozzanak a megvalósításon, ötleteljenek és a párok egymás között próbálják megvalósítani ötleteiket. Tanárként ne segítsünk azonnal ebben a helyzetben, hagyjuk a diákokat, hogy maguktól jöjjenek rá a megvalósítás mikéntjére! Így sokkal jobban magukénak fogják érezni az elkészült projektet is!

3.11. 11. alkalom

3.11.1. Saját projekt befejezése

A diákoknak még 90 perc áll rendelkezésükre, hogy befejezzék az előző alkalommal elkezdett páros munkájukat.

> *Figyelmeztessük őket arra, hogy projektjüket gyakran mentsék, mert egy esetleges programösszeomlás esetén sok munkájuk elveszhet.*

Fontos, hogy ezen az alkalmon befejezzék munkájukat, mert következő alkalommal már a bemutatáson lesz a sor. Biztatssuk a párosokat arra, hogy ne vállalják túl magukat a feladatokkal, hanem inkább legyen egy szép, kerek megoldásuk a kitalált problémára.

> *A diákokat figyelmeztessük a szakköri alkalom végén, hogy következő alkalommal már nem programozással fogunk kezdeni, hanem mindenki kap majd 10-20 percet munkája átgondolásához. Utána be kell mutatniuk elkészült projektjüket.*

3.12. 12. alkalom

3.12.1. Projekt dokumentálása

A szakkör elején mindenki kap még időt projektjének részletes felidézésére, végiggondolására. Ebben a 10-20 percben (a csoport létszámától függ mennyit szánhatunk rá) alkalma nyílik a párnak arra,hogy megbeszéljék hogy osztják fel egymás között a prezentálással kapcsolatos feladatokat. Bemutatójukhoz akár további anyagokat is készíthetnek (pl.: mese illusztrálása netes képekkel) de persze, csak a megadott időkorláton belül.

3.12.2. Prezentálás

Elérkezett az idő, hogy minden páros bemutassa társainak projektjét, melyen több alkalmon keresztül, hosszasan dolgozott.

> *Praktikus ilyenkor a többi diákkal, a hallgatósággal kikapcsoltatni a számítógépek képernyőjét, vagy lecsukatni a laptop tetejét. Ezért azért szükséges, hogy még a csábítás se legyen meg arra, hogy ne társaik bemutatójára figyeljenek.*

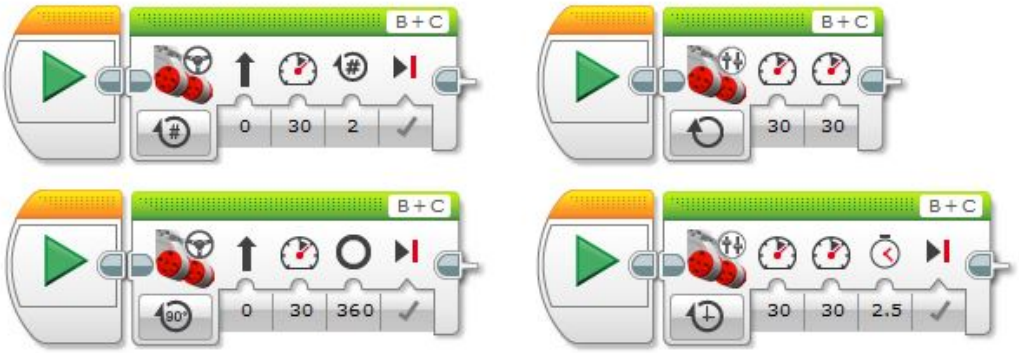
Ha mindenki bemutatta projektjét és még marad egy kis idő, akkor beszélgetéssel fejezzük be a szakkört. Érdeklődhetünk arról, kinek hogy tetszett a szakkör, mi volt a legjobb, mi nem tetszett...Hogyha részletes visszajelzést szeretnénk kapni tanárként, akkor hasznos kitöltetni utolsó alkalom végén a gyerekekkel egy online kérdőívet, melyben kérdéseken keresztül fejthetik ki véleményüket.

3.13. Megoldások

3.13.1. 1.alkalom

1. *Készíts programot, amelyben a robot előre halad! Próbáld ki a kormányzós és a tankos blokkokat is több variációban!*

   Ez csak néhány a sok megoldás közül:



2. *Készíts programot, amelyben a robot hátra halad! Próbáld ki a kormányzós és a tankos blokkokat is több variációban!*

   És ez is csak néhány a sok megoldás közül:

3. *Írj programot, amelyben a robot előre megy 1000°-os tengelyfordulásig!*

4. *Készíts programot, amelyben a robot derékszögben elfordul!*

   Forgásoknál nagyon sokat számít, hogy milyen talajon tesztelünk, ugyanis ez a kód linóleum padlón nagyon szép derékszöget ad. Viszont parkettán tesztelve már nem mindig ilyen pontos, van, hogy picit túlfordul.

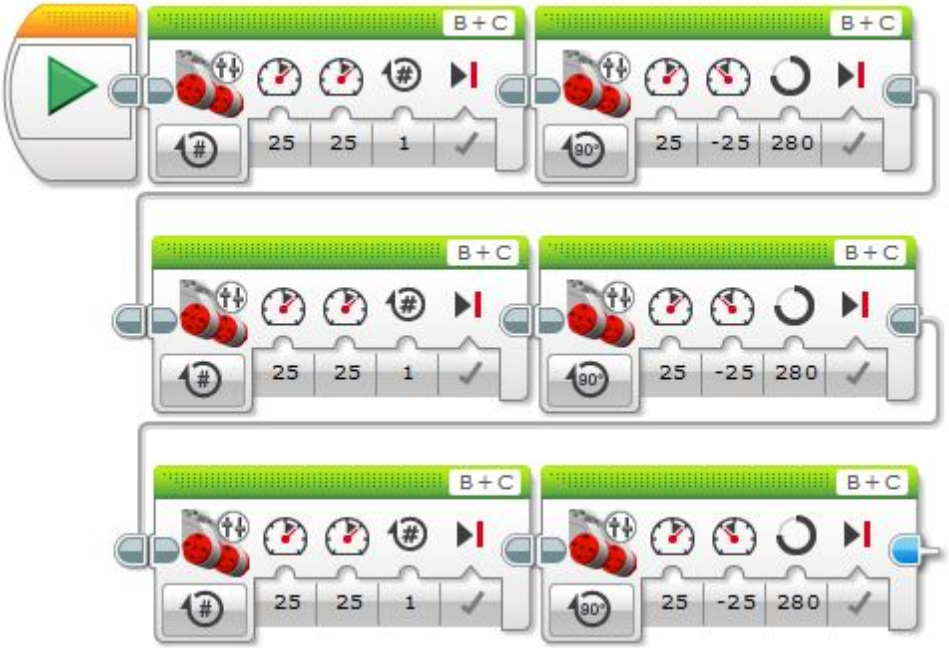
5. *Írj programot, amelyet végrehajtva a robot körbe forog (helyben forog)!*

   Ez csak két megoldás, ezen kívül lehetnek még jó megoldások.

6. *Írj programot, amelyet végrehajtva a robot leír egy négyzetet, háromszöget, kört!*

   Itt is sokat számít a talaj, ezek linóleumon lettek tesztelve, ott szépen működtek.

66. ábra. Négyzet



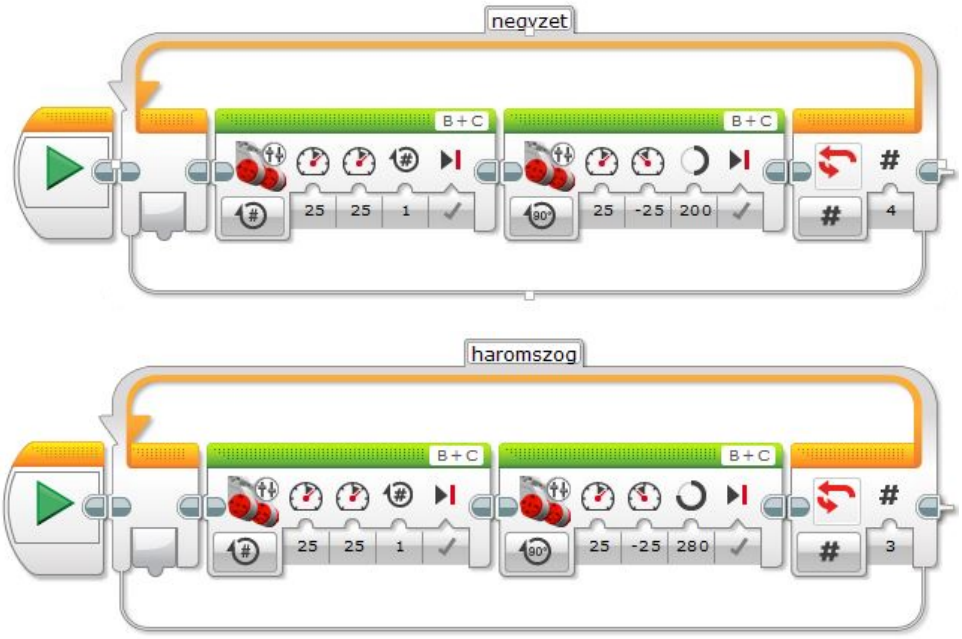
67. ábra. Háromszög

68. ábra. Kör

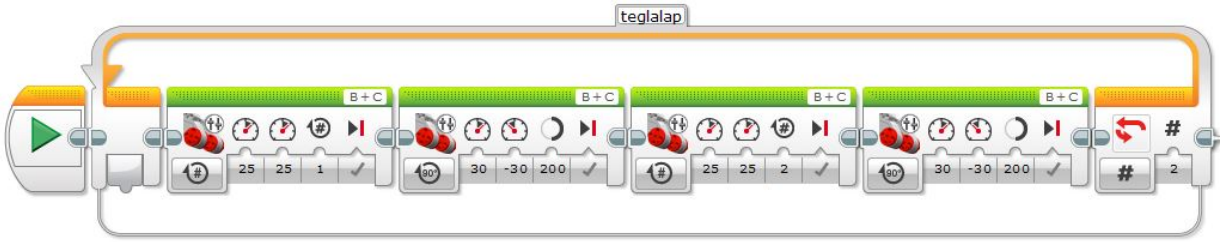
3.13.2. 2.alkalom

Ciklus

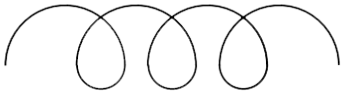
1. *Készítsd el a múlt órai négyzet és háromszög programját ciklussal!*



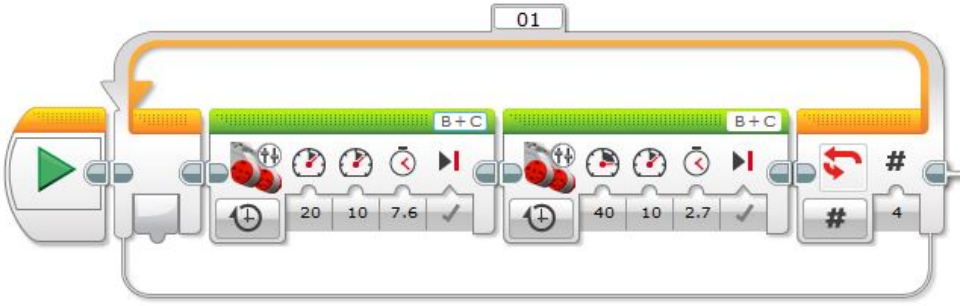
2. *Írj programot, amelyet végrehajtva a robot leír egy téglalapot, de a programod legyen minél rövidebb!*



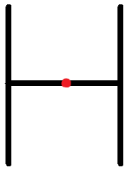
3. *Készíts programot, amelyet végrehajtva a robot az alábbi alakzatot írja le mozgása során, de használj ciklust!*

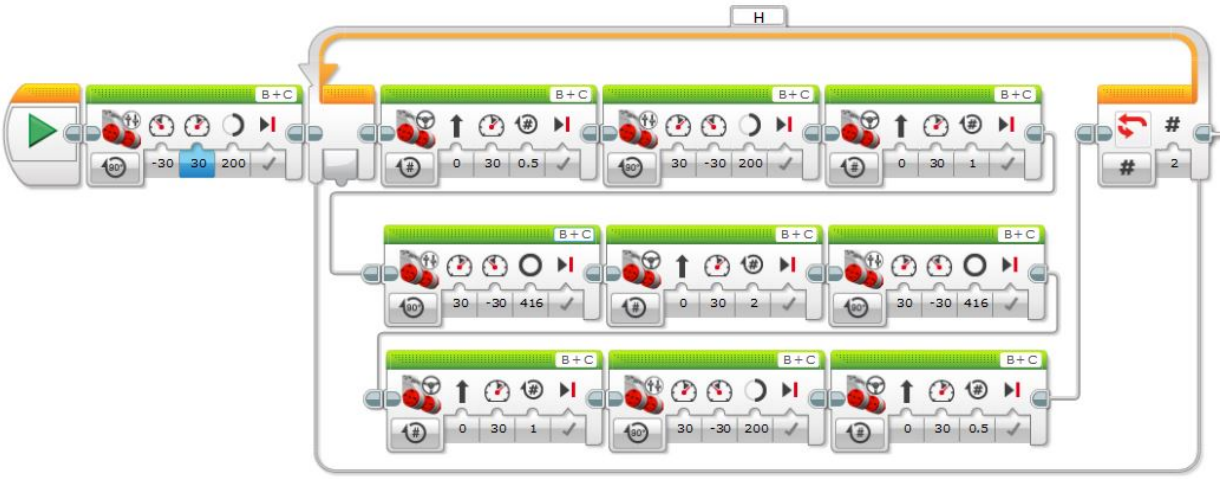


Sajnos a félkört nem lehet úgy megcsinálni, hogy pontosan abba a vonalba érkezzen ahonnan elindult, így mindig picit feljebb is halad, nemcsak oldalra.



4. *A robot mozgása során írjon le egy H betűt úgy, hogy a piros pontból indul!*





Színes ledek és várj blokk használata

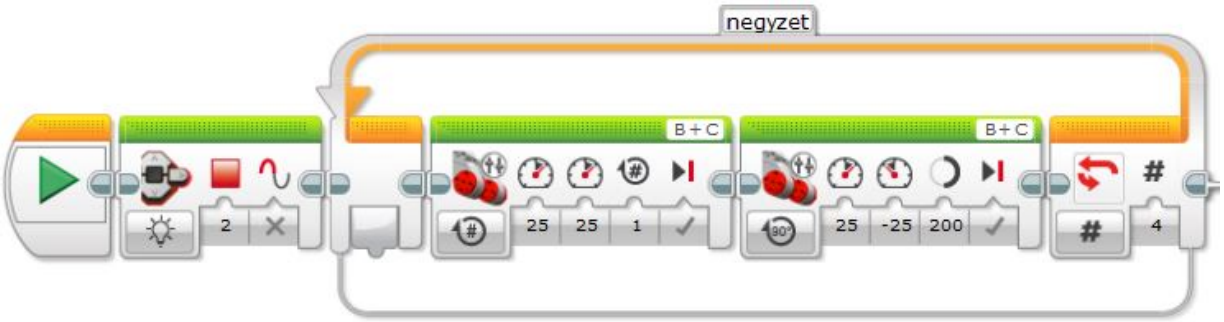
1. *Készíts programot, melyben a ledek sárgán villognak 2 másodpercig!*

2. *Készíts programot, melyben a ledek pirosan villognak 1 másodpercenként!*

3. *Készíts programot, melyben a robot vár 2 másodpercet, majd leír egy kört!*

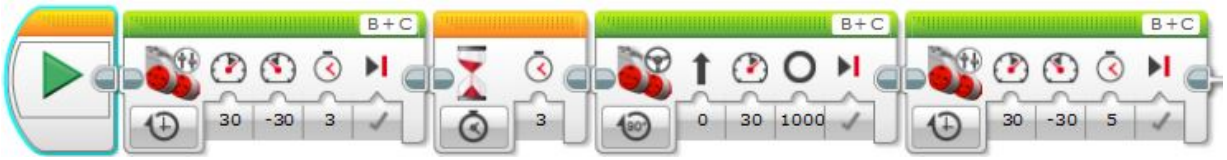
4. *Készíts programot, melyben a robot pirosan villogtatja a lámpáját miközben előre megy egy tengelyfordulást!*

5. *Készíts programot, melyben a robot leír egy négyzetet miközben pirosan világít a lámpája!*



6. *Készíts programot, melyben a led színe 1 másodpercenként változik! (Led villogtatása)*

7. *Írj programot, amelyet végrehajtva a robot helyben forog, majd várakozik 3 mp-ig, ezután előre megy 1000°-os tengelyfordulásig, majd ismét forog 5 mp-ig!*

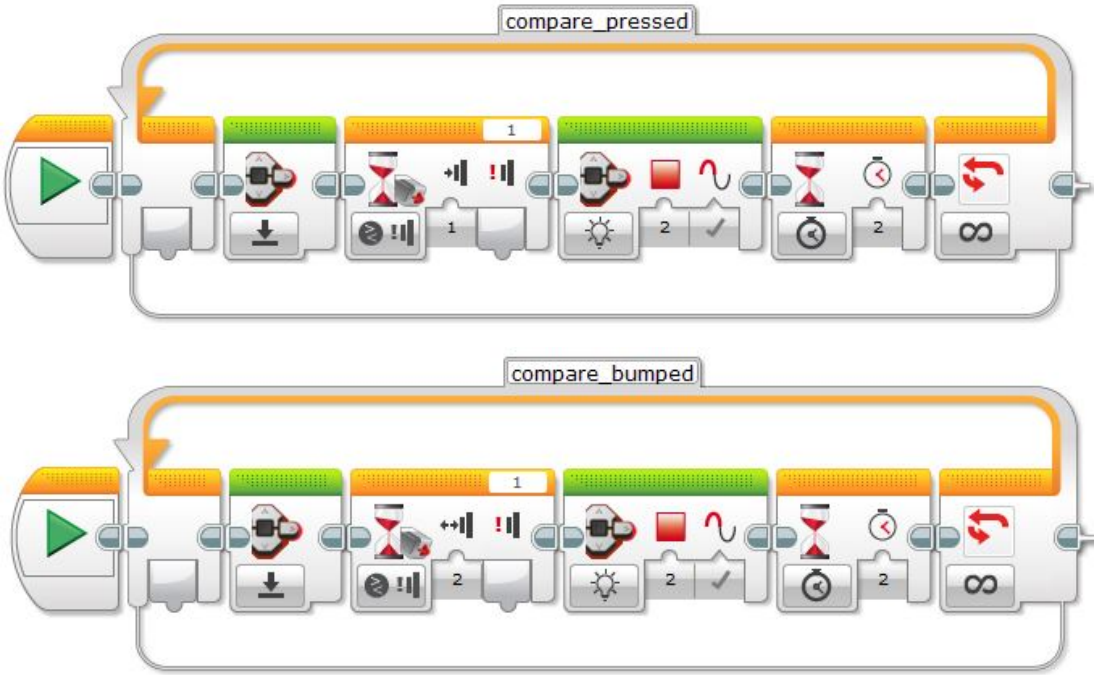


3.13.3. 3.alkalom

Nyomásérzékelő használata

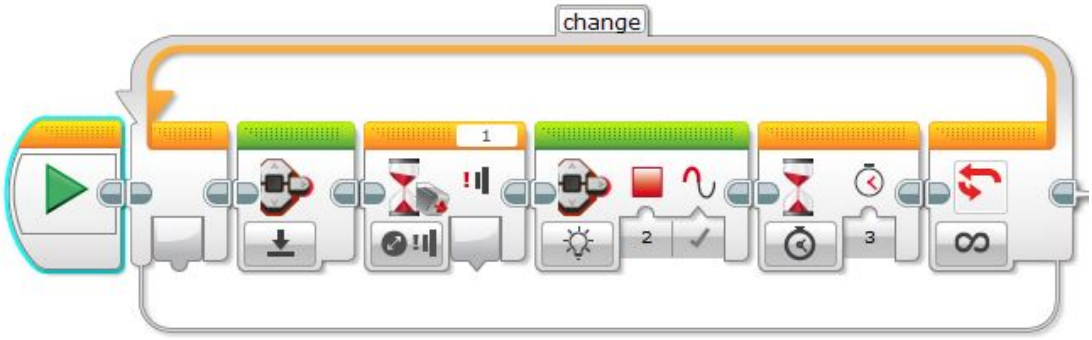
1. *Készíts programot, melyben a robot ledje reset módban villog addig, amíg be nem nyomódik a gombja vagy el nem engedjük a gombot! Ha ez megtörtént, akkor pirosan villogjon 2 másodpercig a lámpája. (Nézd meg, hogy mi történik a kóddal futtatás közben, ha nyomva tartod a gombot, ill. ha csak megnyomod és elengeded!)*

   Azt kell észrevenni, hogy míg ha Pressed módot választunk, akkor addig fog pirosan villogni a led míg nyomva tartjuk az érzékelőt (plusz még 2 másodpercig), különben Reset módban villog. Ha a Bumped módot választjuk, akkor akármeddig is tartjuk nyomva az érzékelőt, mindig az elengedése után villog pirosan 2 másodpercig, addig pedig Reset módban villog.



2. *Készíts programot, melyben a robot ledje reset módban villog addig, amíg nem történik változás az ütközésérzékelő állapotában! Ha megváltozott az állapota, akkor pirosan villogjon 3 másodpercig a lámpája. (Nézd meg, hogy mi történik a kóddal futtatás közben, ha nyomva tartod a gombot, ill. ha csak megnyomod és elengeded!)*

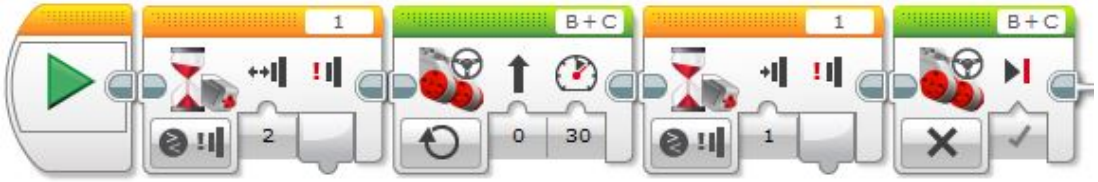
   A robot bármilyen változásnál 3 másodpercig fog villogni, előtte pedig Reset módban teszi ezt.



3. *Írj programot, amelyet végrehajtva a robot elindul egyenesen, ha benyomódik (Bumped) az érintésszenzor!*

4. *Készíts programot, melyben a robot 3 másodpercig egyenesen megy az ütközésérzékelő megnyomására!*

5. *Írj programot, amelyet végrehajtva a robot elindul egyenesen, ha benyomódik (Bumped) az érintésszenzor és egyenesen megy mindaddig, amíg akadálynak nem ütközik! Ekkor álljon meg a robot!*



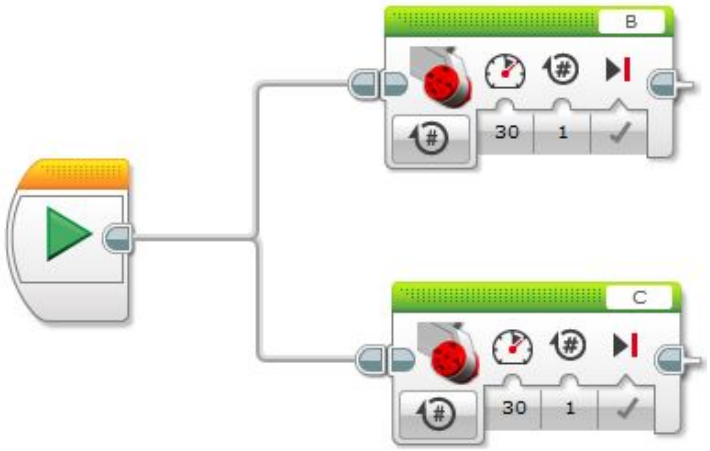
6. *Készíts programot, melyben ha benyomódik az ütközésérzékelő, akkor a robot megy hátra két fordulatot! (Menekülő robot)*

Párhuzamos programok készítése

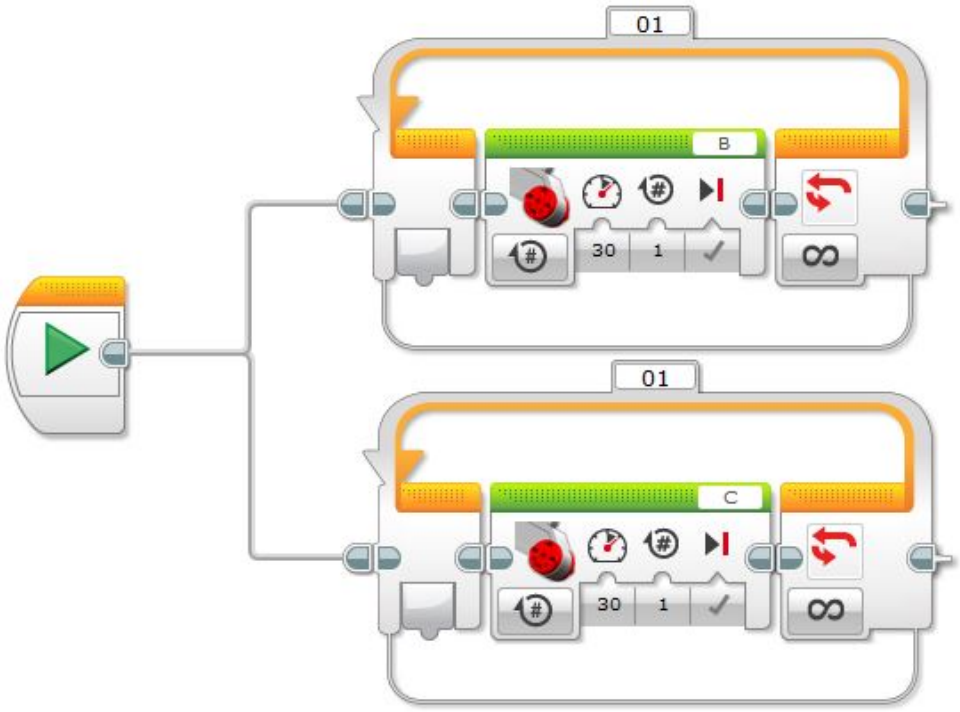
1. *Készíts programot, melyben csak a nagy motorokat használva mozgatod a robotot! Elég ha egy tengelyfordulást megy előre a robot.*

> *Egy kis segítség:*

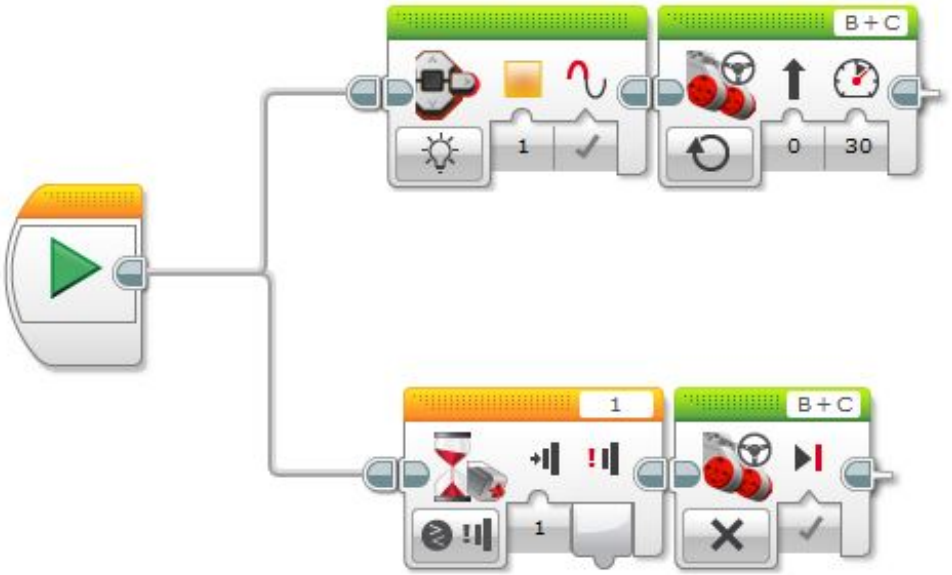
19.ábra



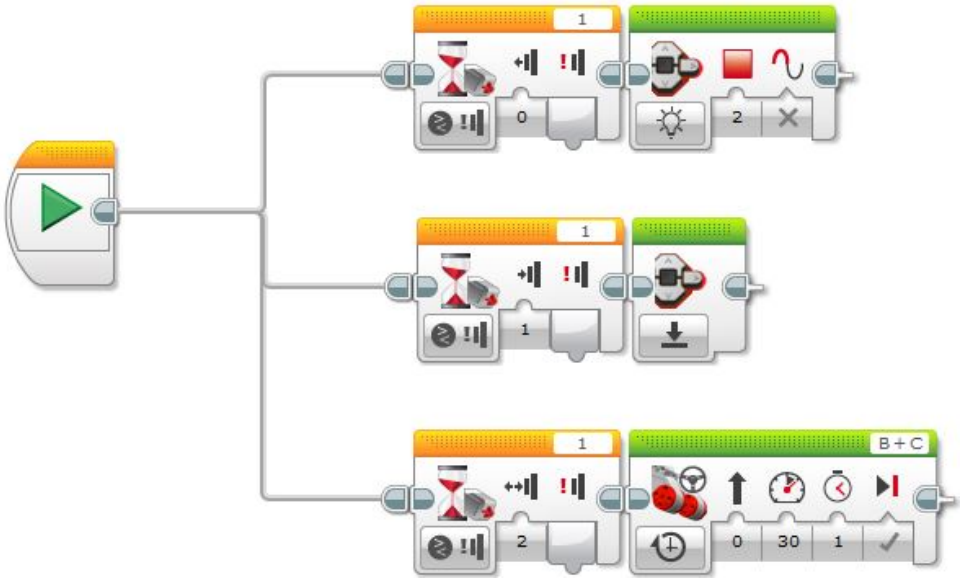
2. *Készíts programot, melyben a motorokat külön mozgatva folyamatosan egyenesen megy a robotod!*



3. *Készíts programot, melyben a robotnak sárgán villog a lámpája míg megy előre, de ha nekimegy valaminek akkor megáll!*



4. *Készíts programot, melyben az ütközésérzékelőt használod úgy, hogy ha 0 értéket észlel, akkor pirosan világít, ha 1-es értéket kap, akkor Reset módban villog, végül ha Bumped ütközést észlel, akkor megy előre egy másodpercig!*



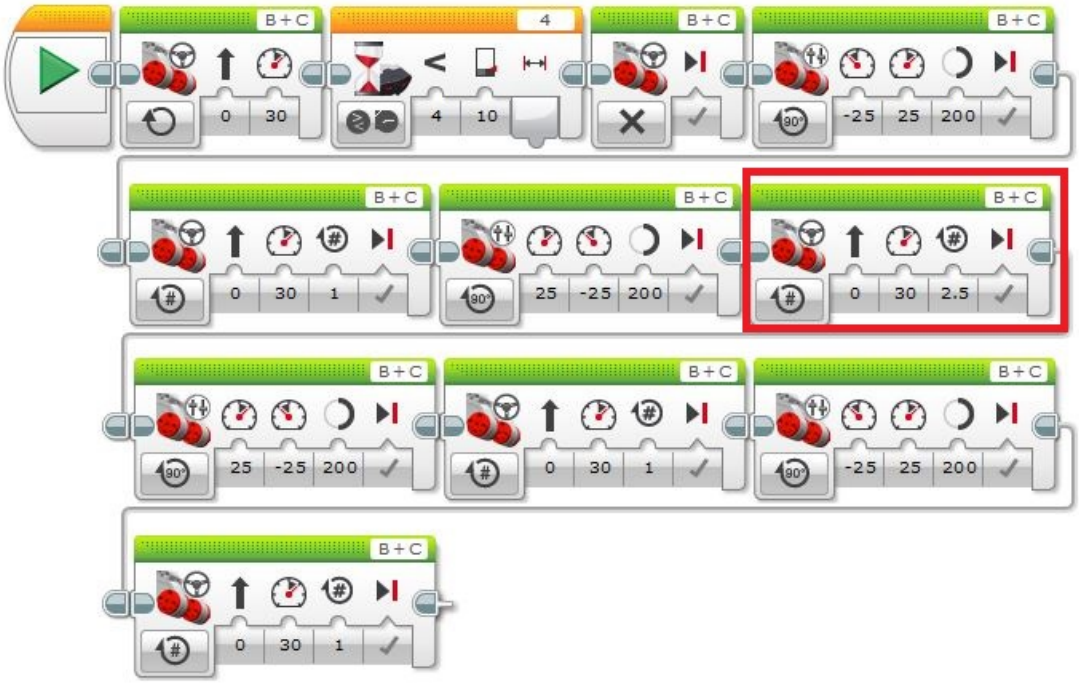
3.13.4. 4.alkalom

Távolságérzékelő(k) használata

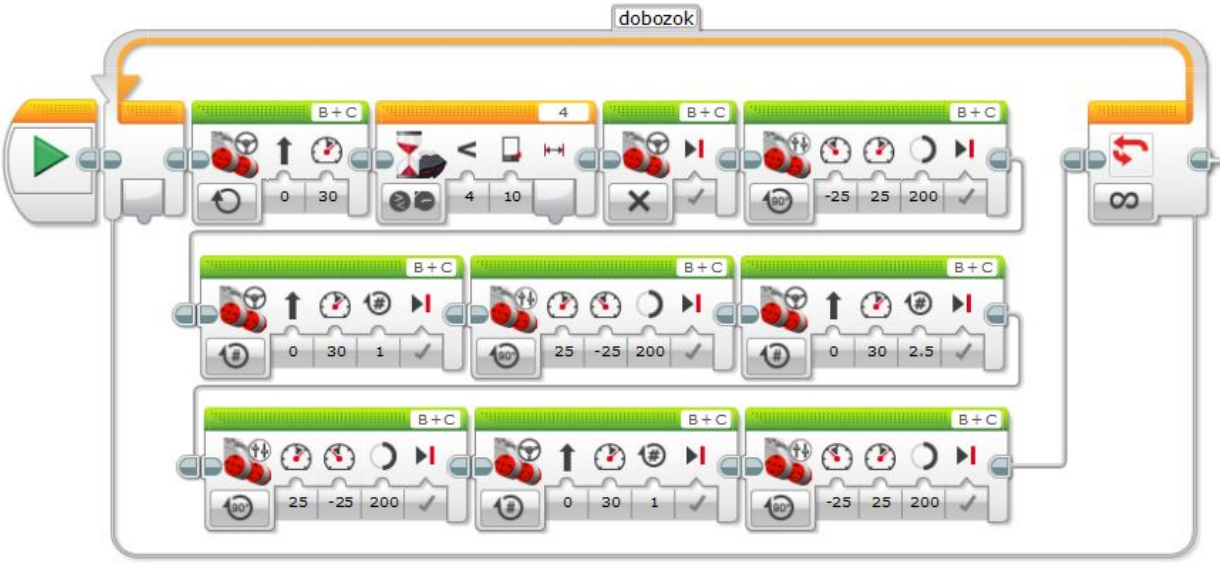
1. *Írj programot, amelyet végrehajtva a robot előre megy mindaddig, amíg a távolságérzékelője 15 cm-nél kisebb távolságot nem mér! Ekkor álljon meg!*

2. *Készíts programot, melyben a robot egyenesen halad, ha érzékel 10 cm-nél kisebb távolságban valamit, akkor megáll, kikerüli (mint egy kocka alakú dobozt), majd folytatja útját.*

   A piros négyzetben lévő blokk harmadik paraméterének (Rotations) értéke attól függ, hogy mekkora tárgyat teszünk a robot elé. Úgy kellene megkerülni a testet, hogy nem ér hozzá, nem dönti fel. Ez lehet akár egy kisebb papírdoboz, tolltartó, papírtörlőnek a hengere vagy épp egy táska (ami éppen akad).



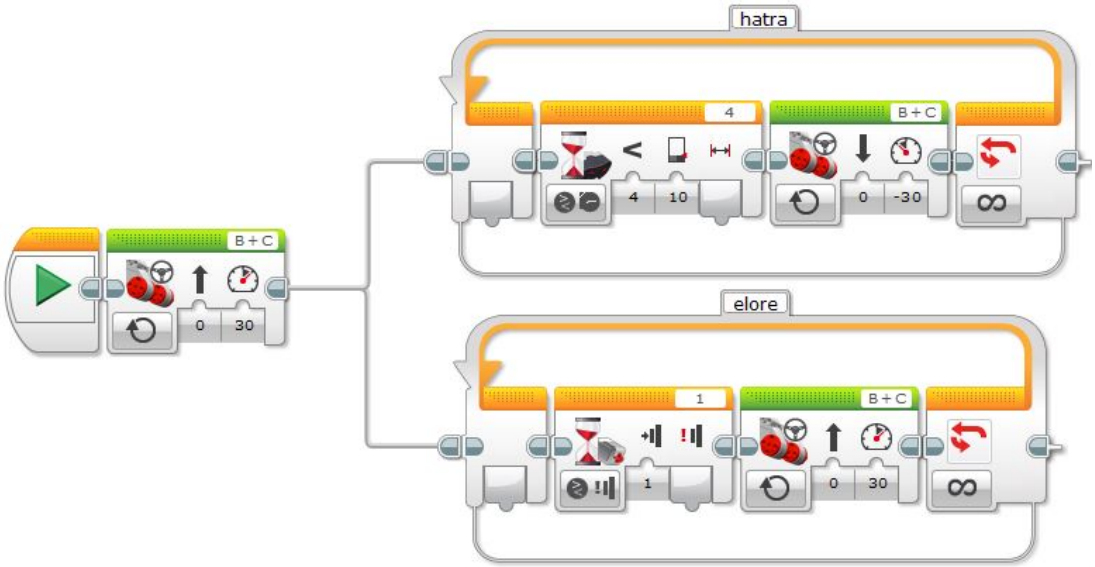
3. *Készítsd el az előző programot úgy, hogy végtelen sok dobozt ki tudjon kerülni!*



4. *Írj programot, amely a következő feladatot hajtja végre! A robot előre megy addig, amíg nem észlel valamit 10 cm-en belül maga előtt, ekkor elfordul 180°-ot, majd ezt ismételgeti. (Forgolódó program)*

5. *Írj programot, amelyet végrehajtva a robot addig halad előre, amíg 10 cm-en belül nem érzékel valamilyen akadályt! Ha valamilyen akadályt észlel, akkor elindul hátra, de ha nekitolat valaminek, akkor elindul előre. (Pattogó program)*

   Ennél a feladatnál át kell szerelni a nyomásérzékelőt a robot hátuljára. Ha nem tesszük ezt meg, akkor is működni fog a program, csak épp nem lesz pattogós.

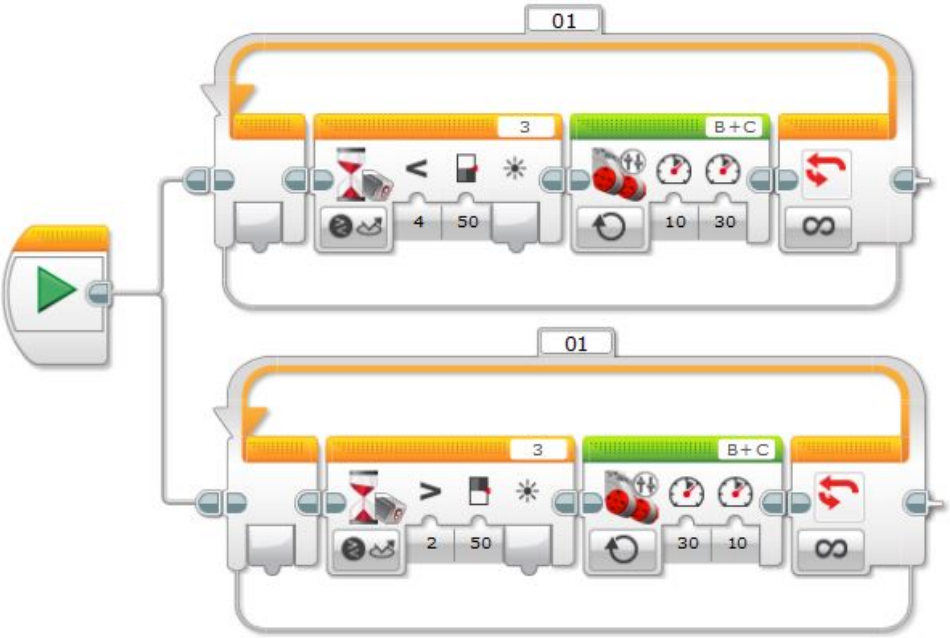


Színérzékelő használata

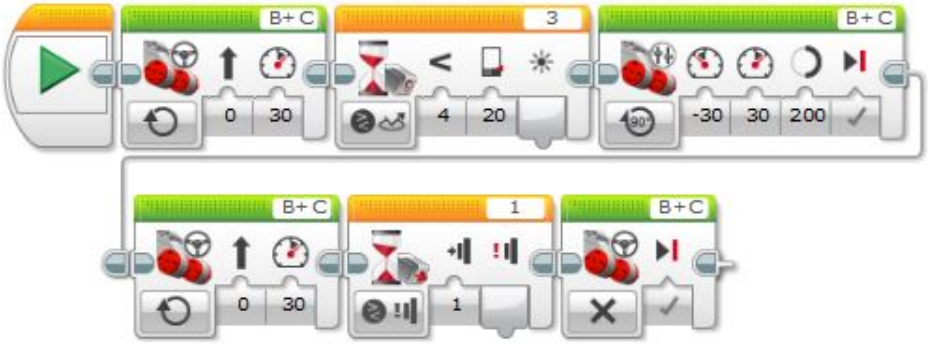
1. *Írj programot, amelyet végrehajtva a robot egyenesen halad előre mindaddig, amíg a fényérzékelője az alapszíntől eltérő színt nem észlel, ekkor álljon meg és tolasson 2 mp-ig! (A feladat megoldása előtt mérjük meg, hogy a különböző színű felületekről milyen intenzitású fényt ver vissza.)*

2. *Írj programot, amelyet végrehajtva a robot addig forog, amíg sötétebb színt nem érzékel!*

3. *Írj programot, melyben a robot egy sötét vonalat követ! (Vonalkövetés)*



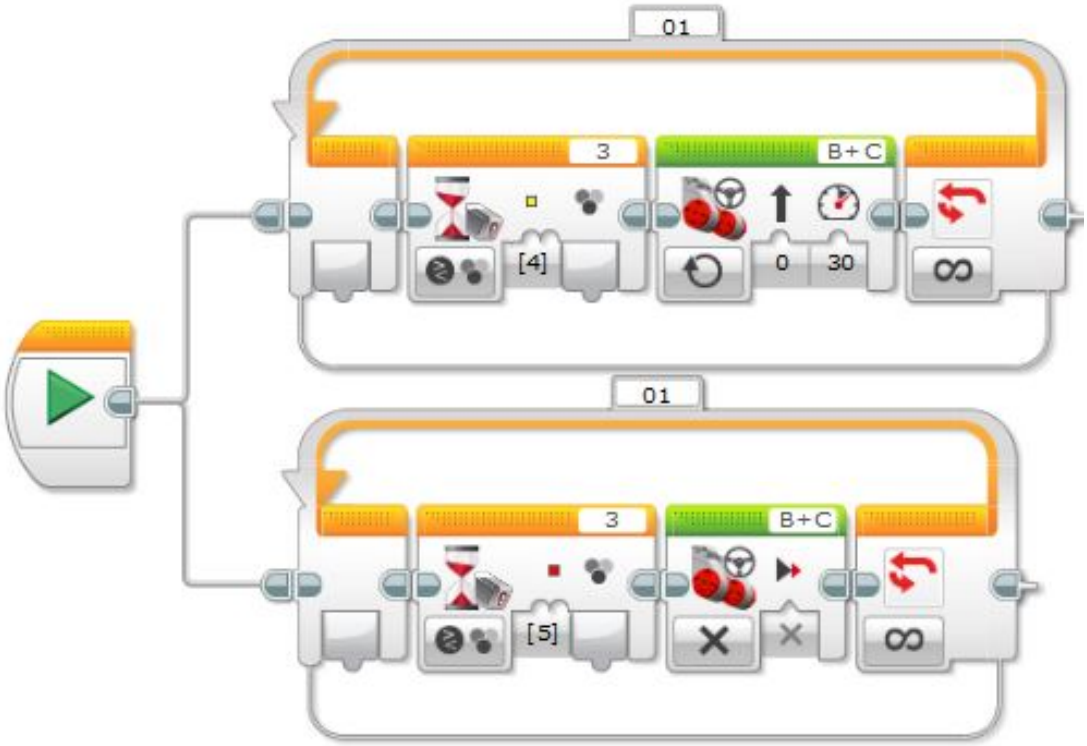
4. *Írj programot, amelyet végrehajtva a robot egyenesen megy előre addig, amíg egy sötét csíkot el nem ér. Ekkor balra fordul kb. 90°-ot, majd egyenesen megy, amíg akadálynak nem ütközik!*



5. *Készíts programot, melyben a robot 20-as sebességgel halad előre, de ha zöld színt észlel, akkor megáll és pirosan felvillantja a ledjét!*

6. *Készíts programot, melyben a robot 20-as sebességgel halad előre, ha kék színt érzékel, akkor tesz egy fordulatot a tengelye körül, majd folytatja útját az eredeti irányban.*

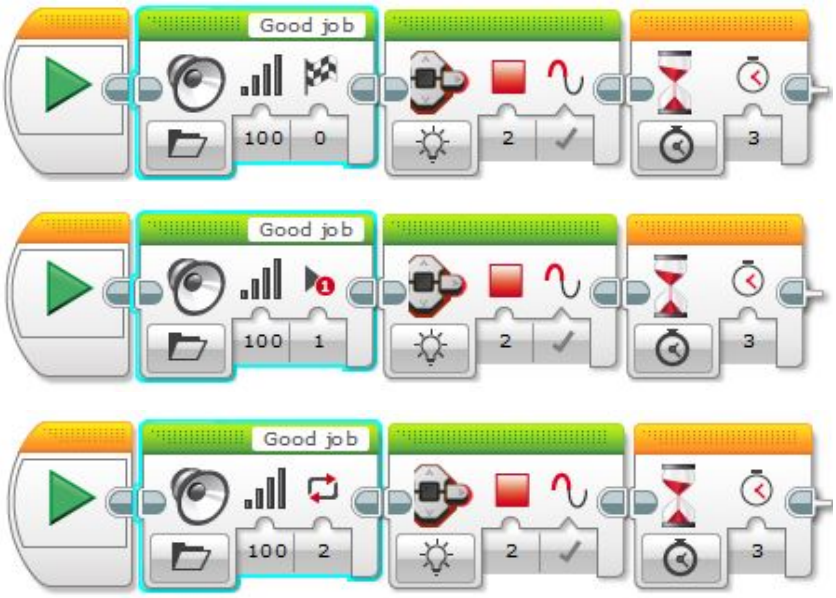
7. *Készíts programot, melyben a robot a következő feladatot oldja meg: Ha sárga színt érzékel a robot, akkor elindul, de ha pirosat, akkor megáll!*



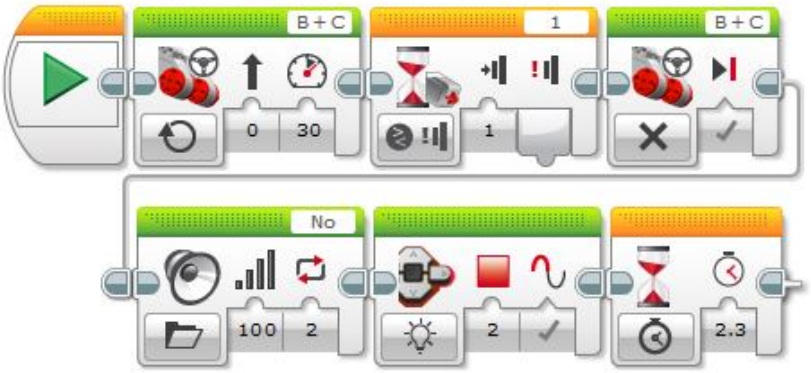
3.13.5. 5.alkalom

Hang lejátszása

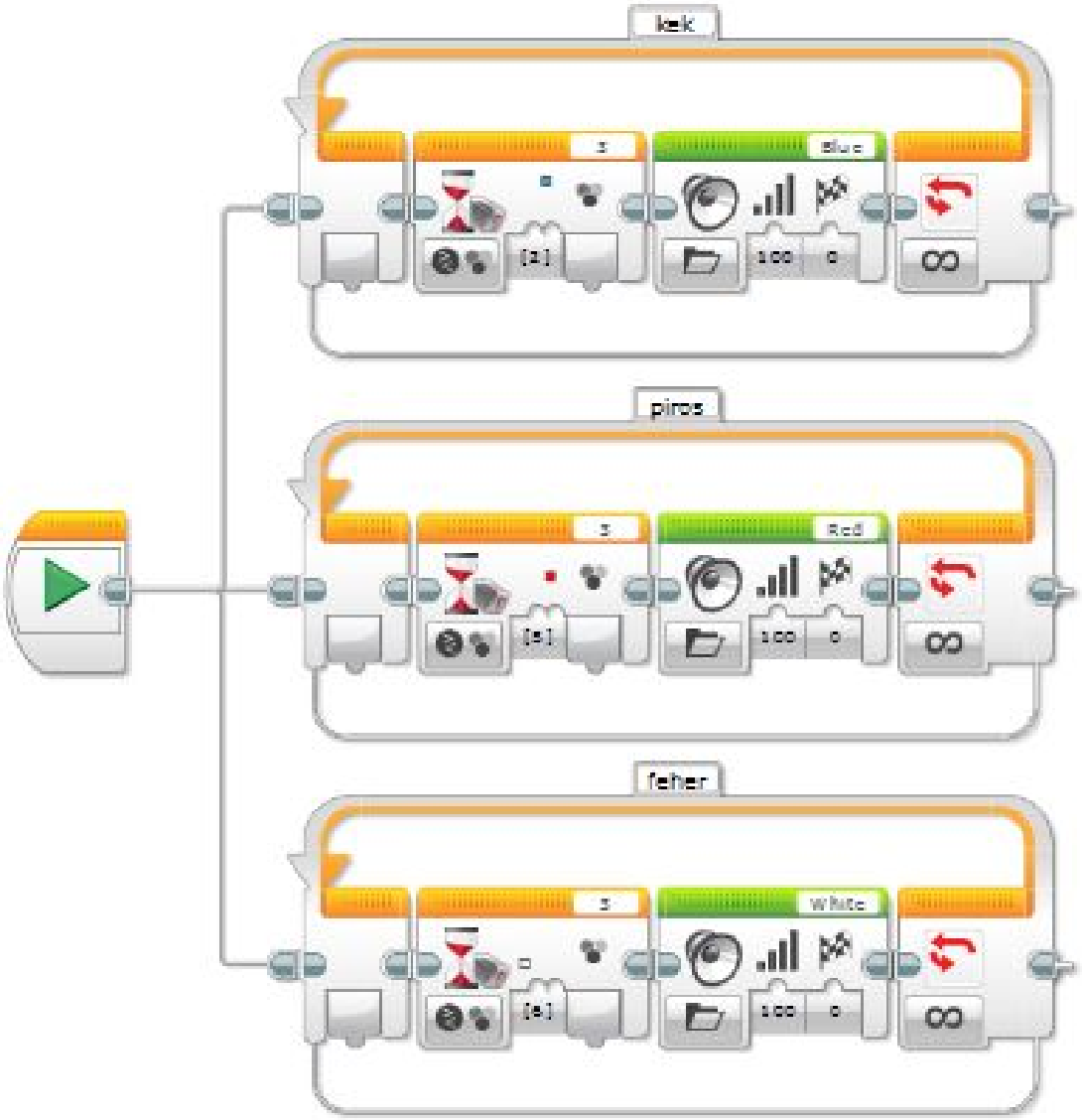
1. *Készíts prgoramot, melyben a Good job hangot játszod le úgy, hogy utána 3 másodpercig pirosan villog a lámpája a robotnak! Próbáld ki mindhárom lejátszási módot!*



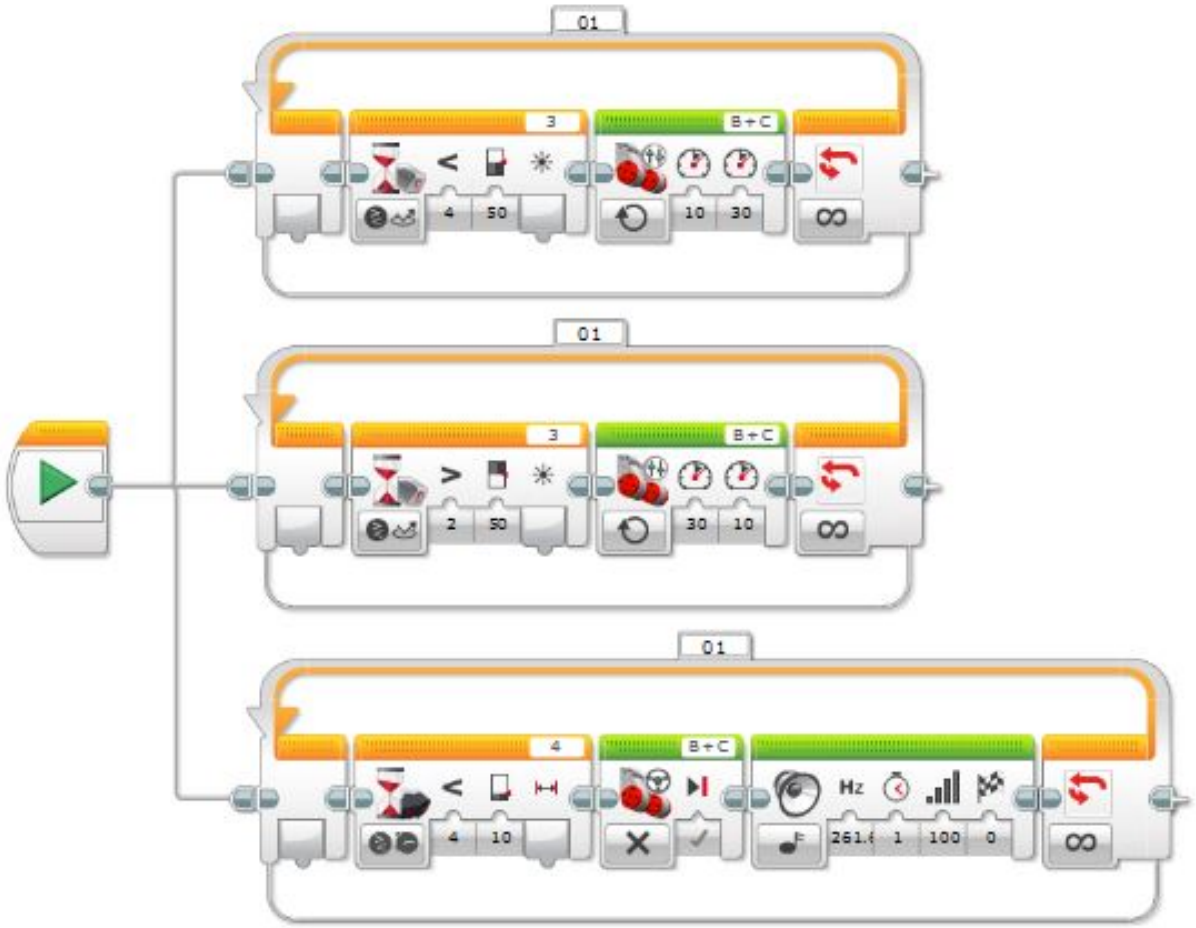
2. *Készíts programot, melyben a robot egyenesen halad előre, de ha nekiütközik valaminek, akkor a No hangot adja ki 2-es lejátszási módban és utána 2.3 másodpercig villogtatja pirosan a lámpáját!*



3. *Készíts programot, melyben a robot ha érzékel egy színt, akkor kimondja angolul a nevét! (Hangos színek)*

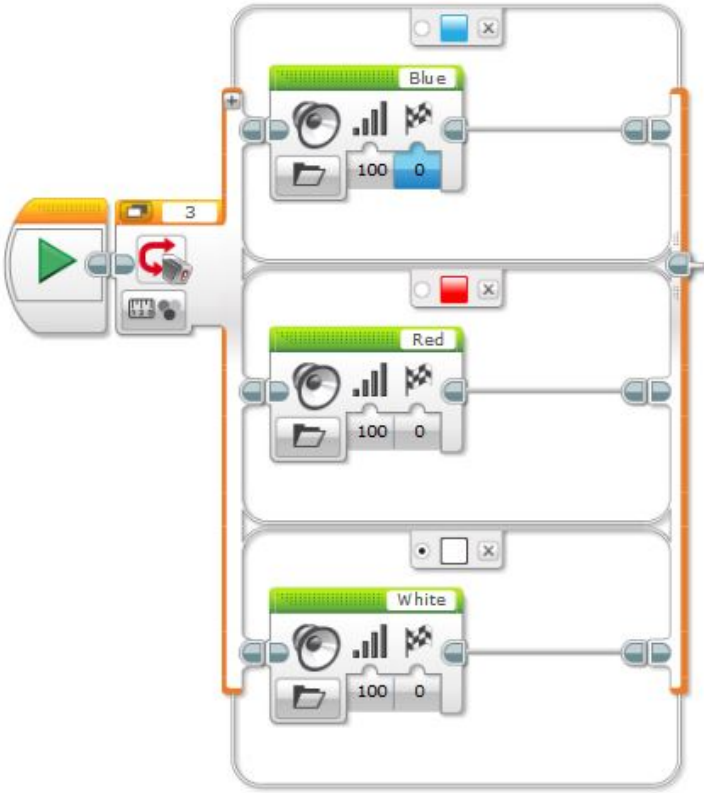


4. *Készíts programot, melyben a robot egy vonalon halad, de ha érzékel maga előtt 10 cm-en belül valamilyen tárgyat, akkor dudál egyet a C frekvencián, majd folytatja útját!*

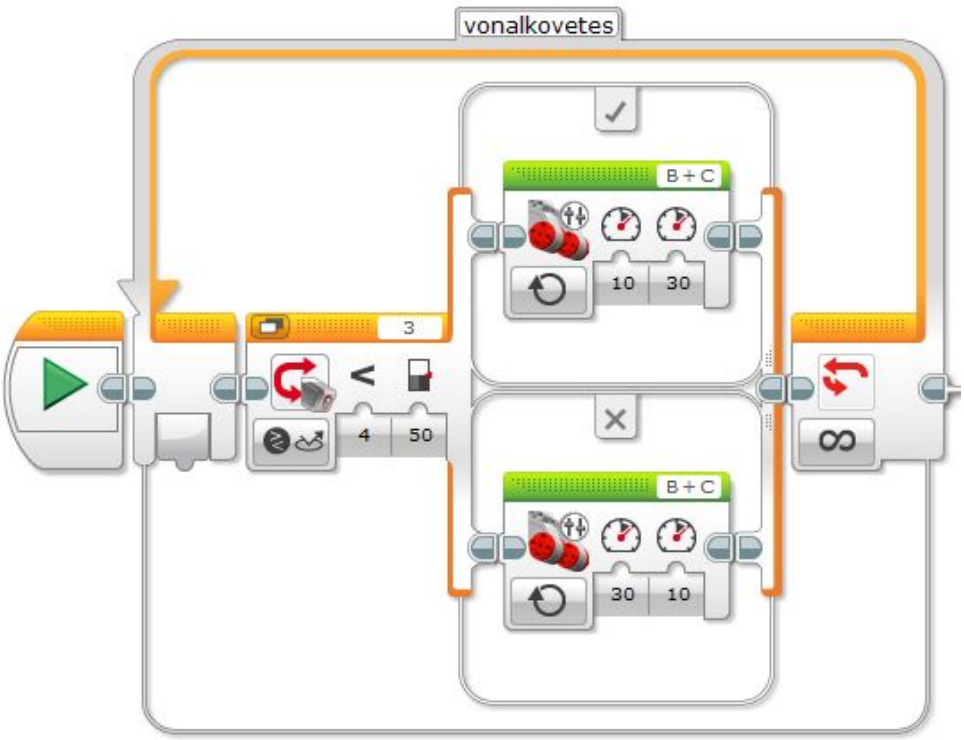


Elágazás

1. *Készítsd el a Hangos színek c. feladatot a switch blokk használatával!*



2. *Készítsd el a Vonalkövető feladatot az elágazás használatával!*



3. *Készíts programot, melyben a robot a következő feltételek alapján cselekszik!:*

   Ha piros színt érzékel, akkor pirosan villogtatja a ledjét 2 másodpercig,

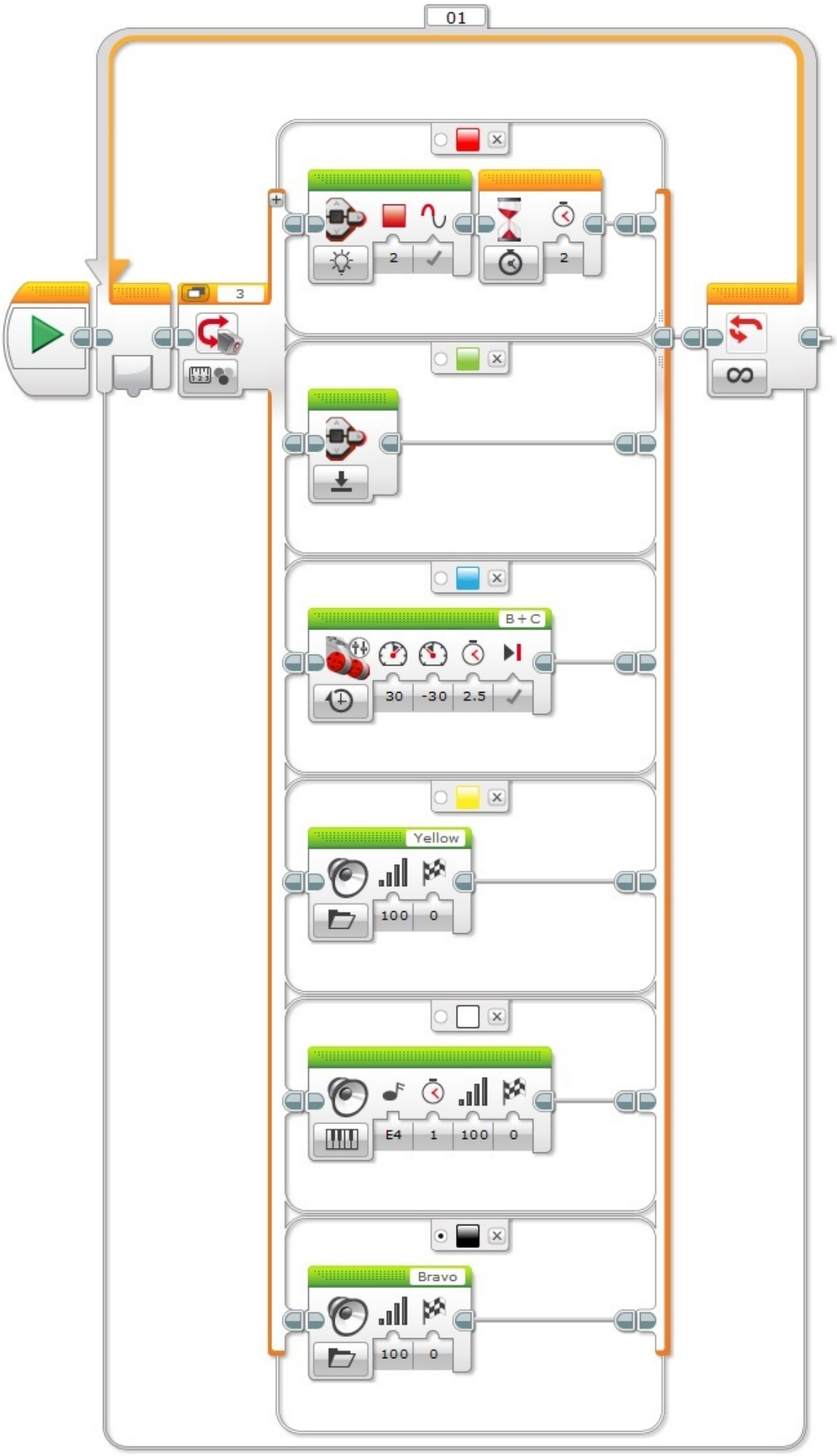
   ha zöldet, akkor reset módban villog,

   ha kéket, akkor egyhelyben megfordul,

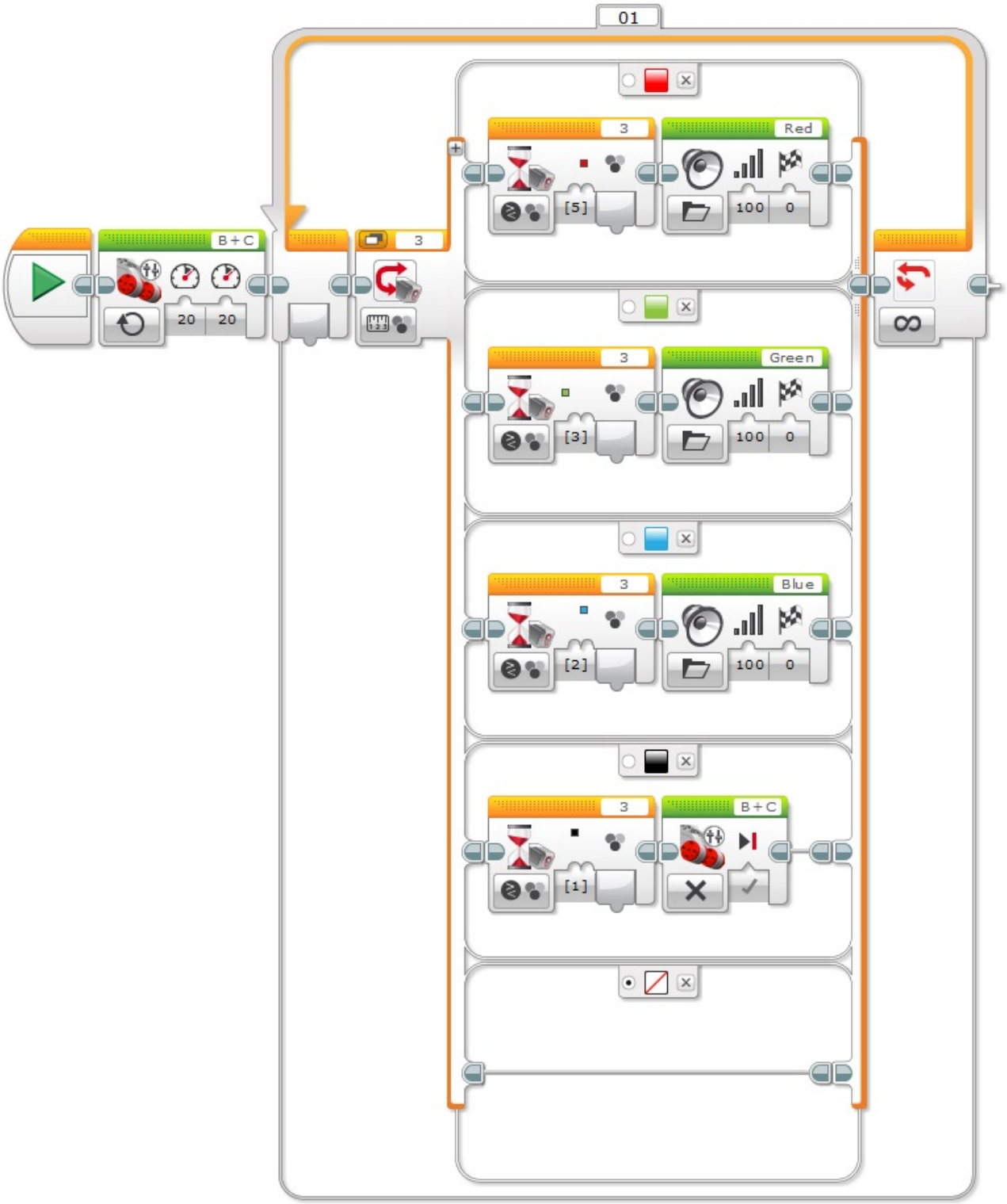
   ha sárgát, akkor angolul kimondja, h sárga,

   ha fehéret, akkor egy zongora hangot játszik le,

   ha pedig feketét észlel, akkor azt mondja, hogy Bravo.


01
3
2
2
B+C
30
-30
2.5
Yellow
100
0
E4
1
100
0
Bravo
100
0

4. *Készíts programot, melyben egyenesen halad előre a robot és ha piros/zöld/kék csíkot érzékel, akkor kimondja a színt angolul, de ha feketét érzékel akkor leállítja a motort!*

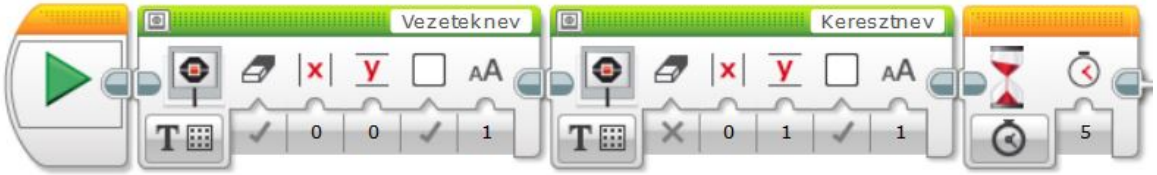


3.13.6. 6.alkalom

Rajzolás a tégla képernyőjére

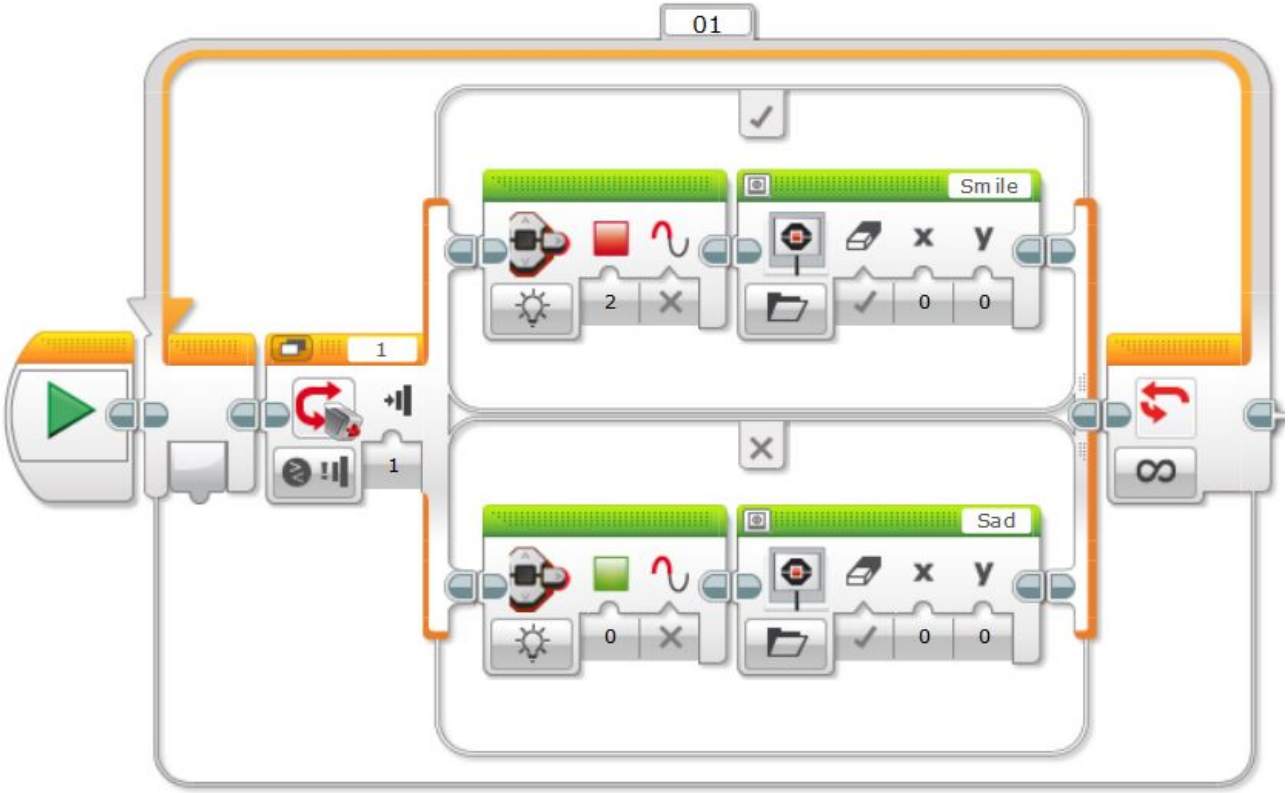
1. *Írd ki saját becenevedet a robot képernyőjére!*

2. *Írd ki a saját nevedet a robot képernyőjére úgy, hogy vezetékneved a bal felső sarokban legyen, keresztneved pedig az alatt! Használj félkövér betűtípust!*

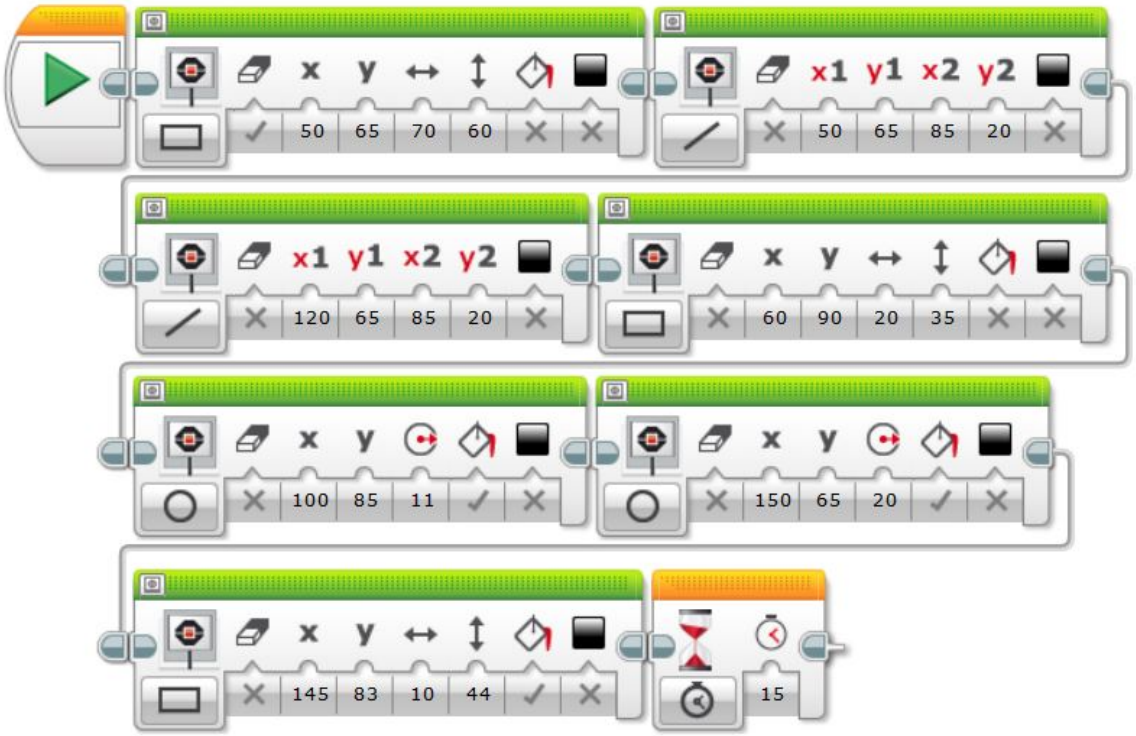


3. *Jeleníts meg robotod képernyőjén egy tetszőleges képet fájlból 3 másodpercig!*

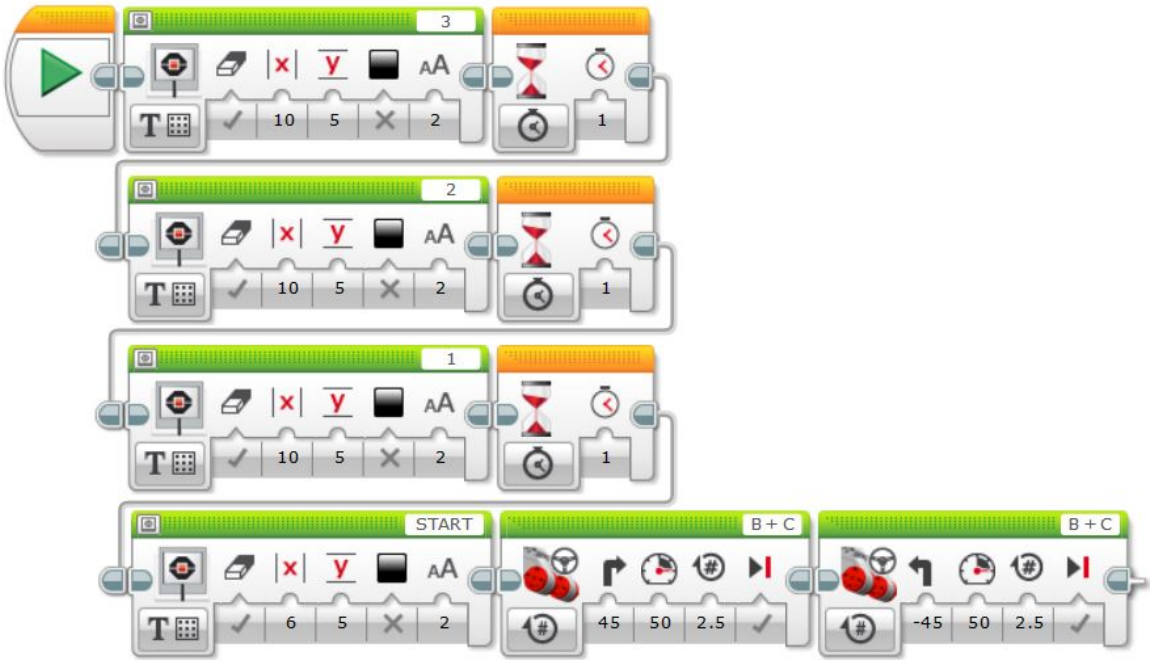
4. *Írj programot, melyben a roboton lévő led pirosan világít, amíg be van nyomva az érintésérzékelő és zölden, ha ki van engedve! Világítás mellett mosolyogjon, amíg be van nyomva, szomorú arcot mutasson, ha ki van engedve!*



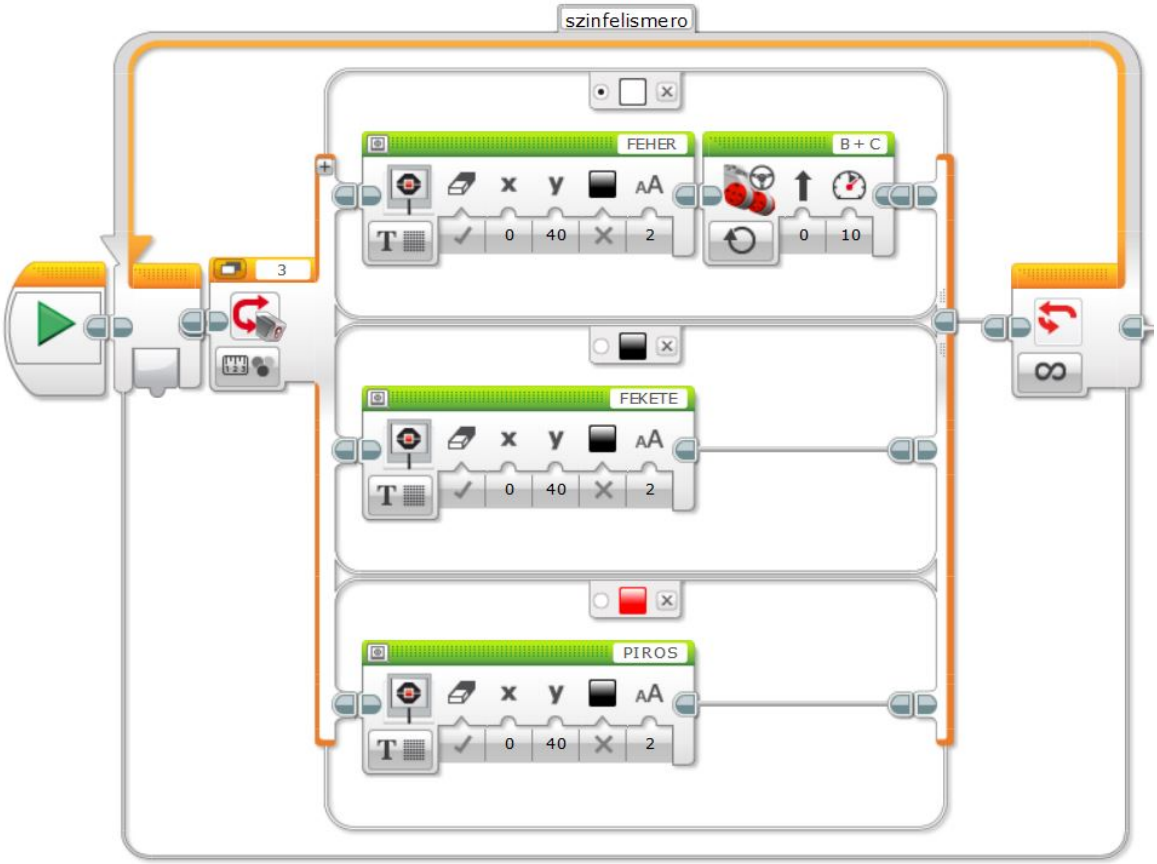
5. *Rajzolj egy házikót alakzatok segítségével a robot képernyőjére! A részletek rád vannak bízva!*



6. *Készíts programot, melyben a robotod a következőt hajtja végre: amikor elindítod a programot a képernyőjén 3-tól visszaszámol 1-ig 1 mp-enként, majd kiírja, hogy START a képernyő közepére és motorjait beindítva leír egy tetszőleges alakzatot (pl.: négyzetet, S-alakot).*

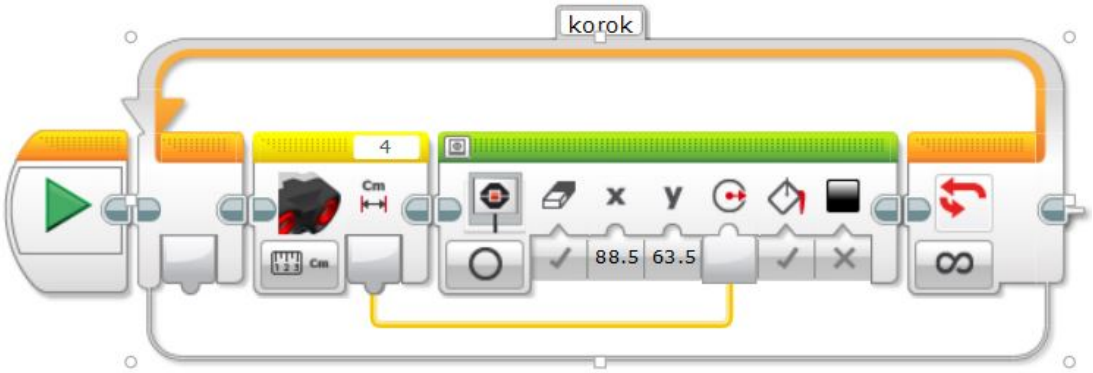


7. *Készíts programot, mely során robotod lassan egyenesen előre halad a pályán (fehér nagy lap/felület fekete, piros alakzatokkal). Ha fekete színt lát írja ki a képernyőre, hogy „FEKETE", ha pirosat, akkor azt, hogy „PIROS", ha fehéret, akkor pedig azt, hogy „FEHER"!*

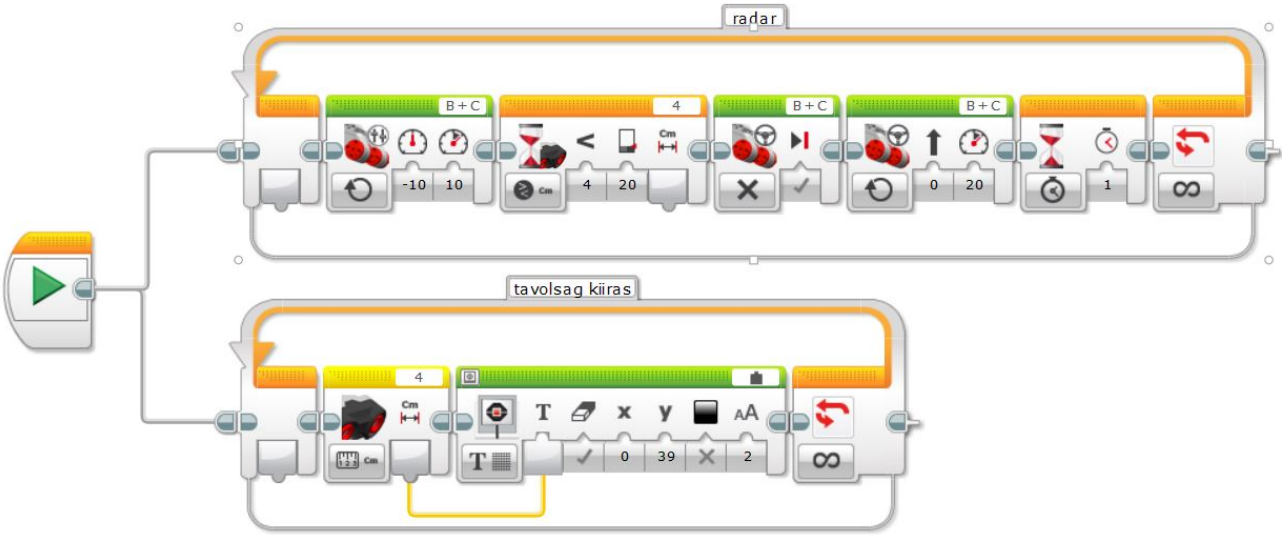


Érzékelő blokkok használata

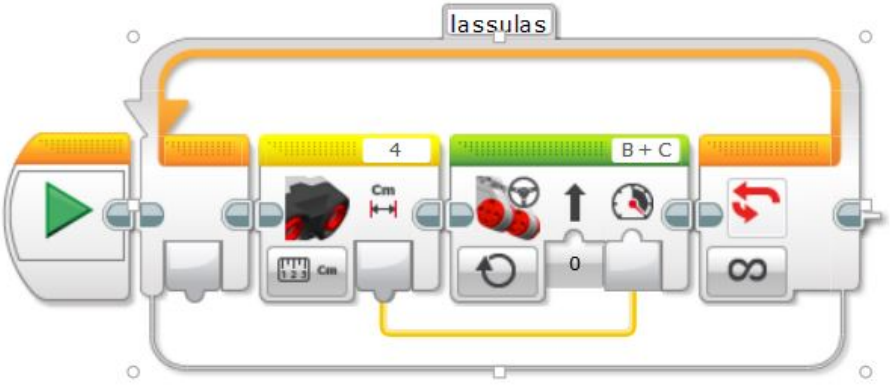
1. *Készíts olyan programot, melynek eredményeképp a robotod akkora sugarú körlapot rajzol a kijelző közepére, amilyen távol észlel magától valamit távolságérzékelőjével*



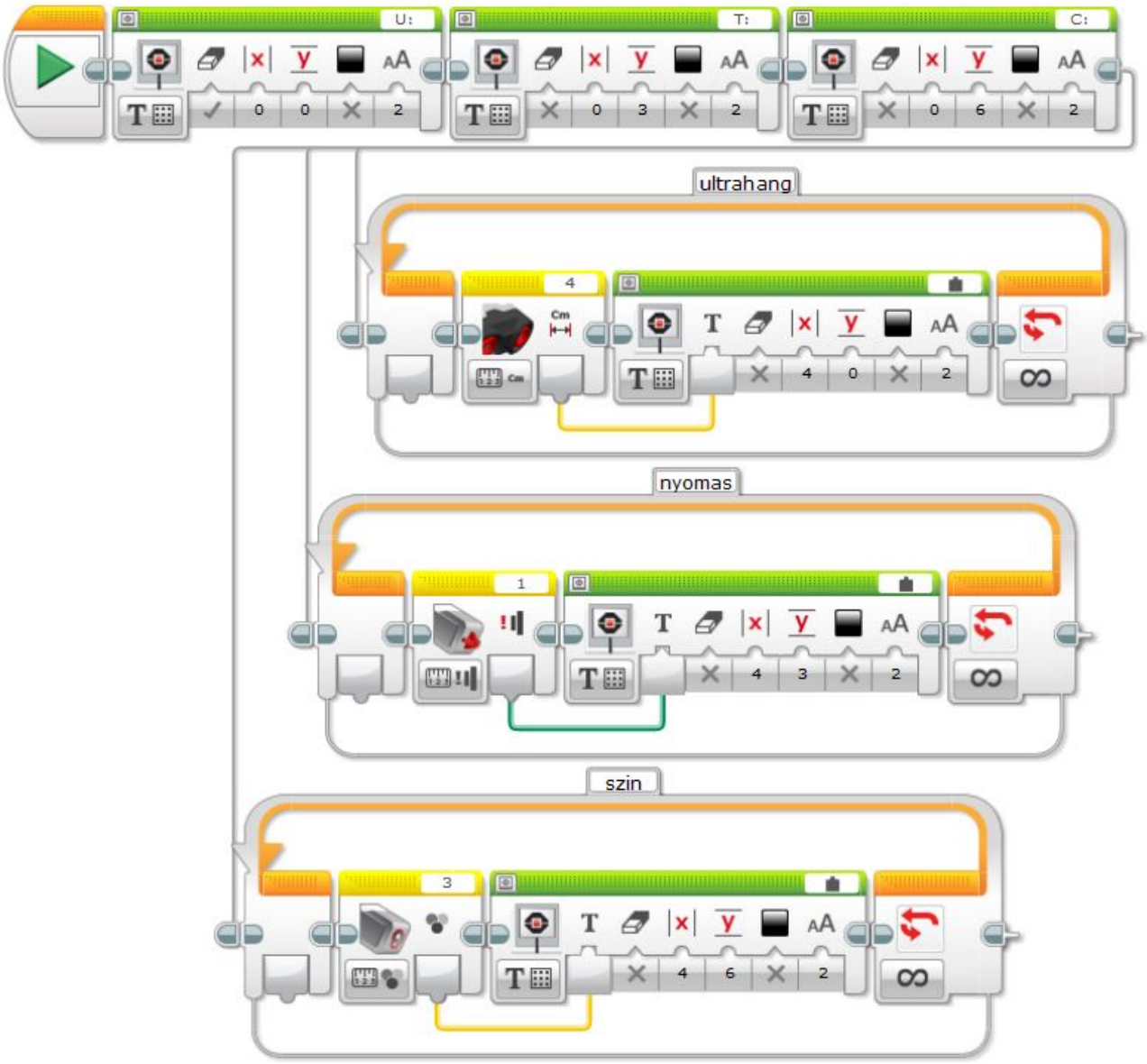
2. *Fejleszd tovább a már korábban megvalósított radar programodat! Miközben a robot körbe-körbe pörög folyamatosan írja ki a képernyőre az általa mért távolság értékét!*



3. *Valósítsd meg a következő programot: a robot haladjon egyenesen, a sebességét az ultrahangos távolságérzékelő értéke határozza meg!*



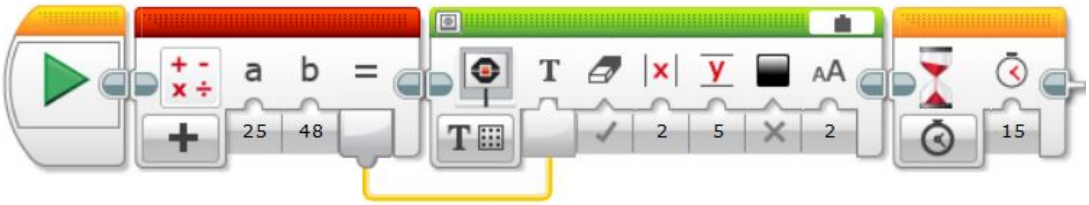
4. *Robotod írja ki mindhárom tanult érzékelő értékét egymás alá a következő elrendezésben: érzékelő neve: érték*



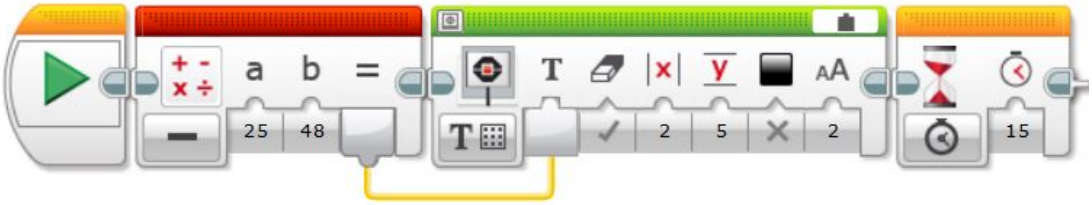
3.13.7. 7.alkalom

Matematikai műveltek

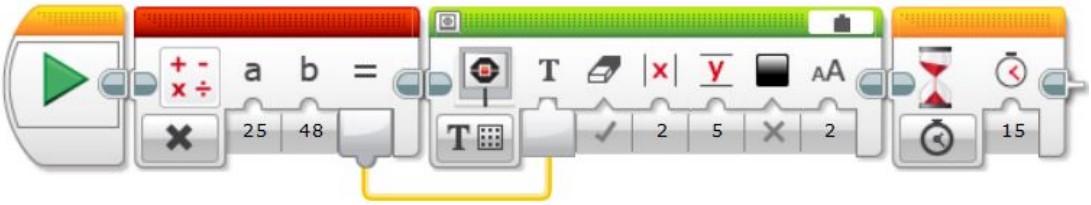
1. *Próbáld ki tetszőleges a és b értékekkel az összeadás, kivonás, szorzás műveleteket! Írd ki eredményed a robot képernyőjére!*



69. ábra. Összeadás



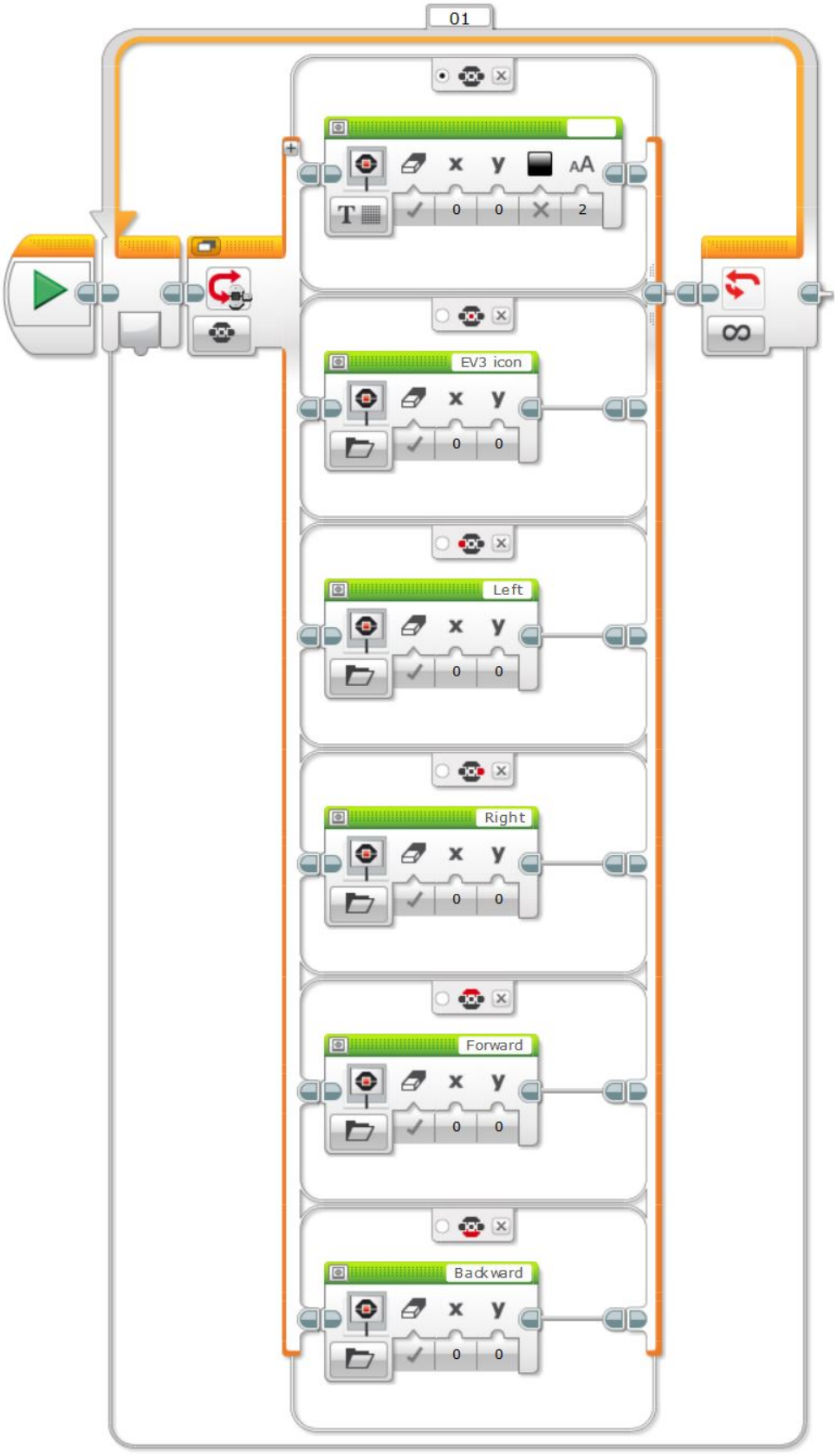
70. ábra. Kivonás



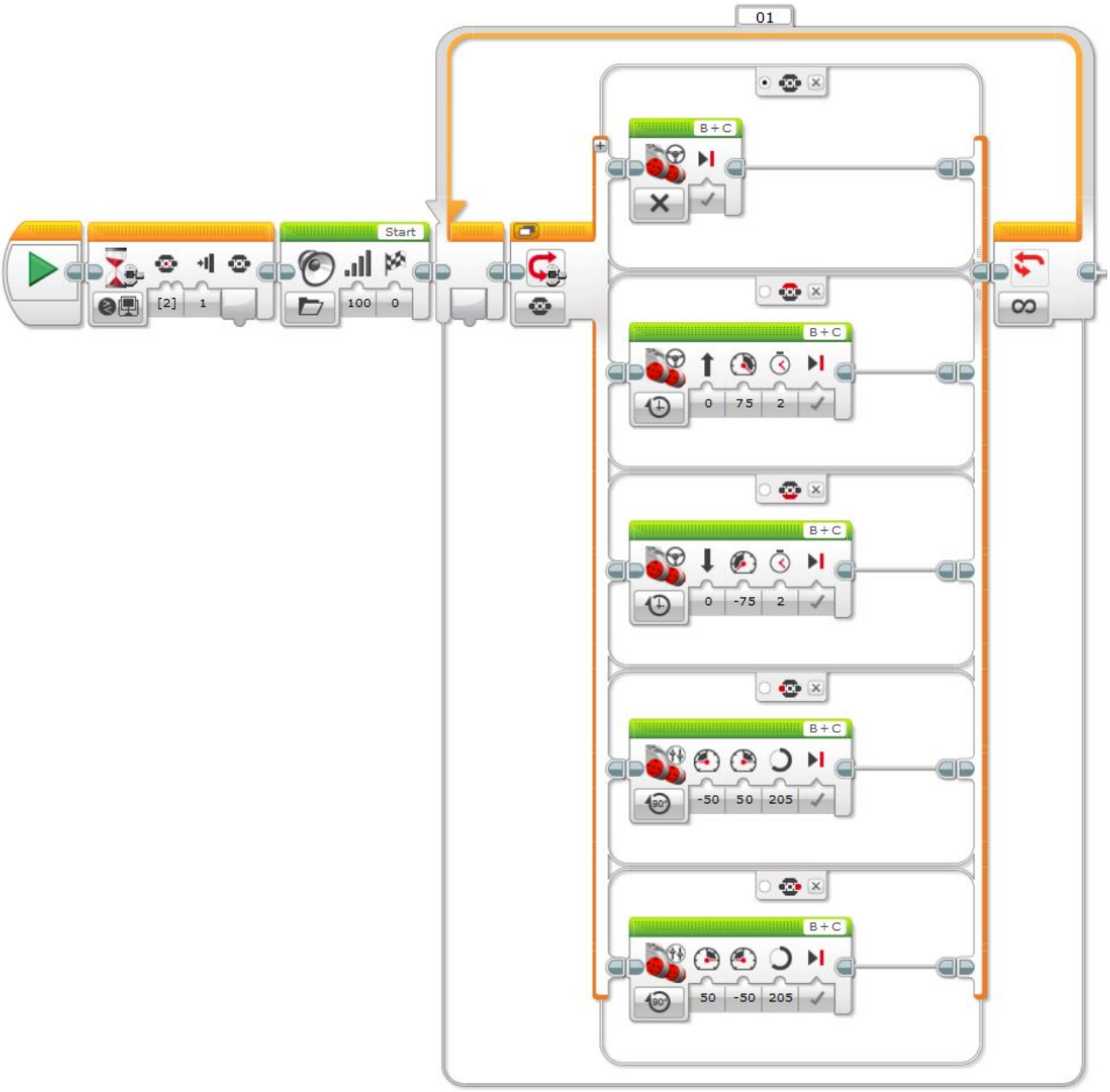
71. ábra. Szorzás

A tégla gombjainak használata

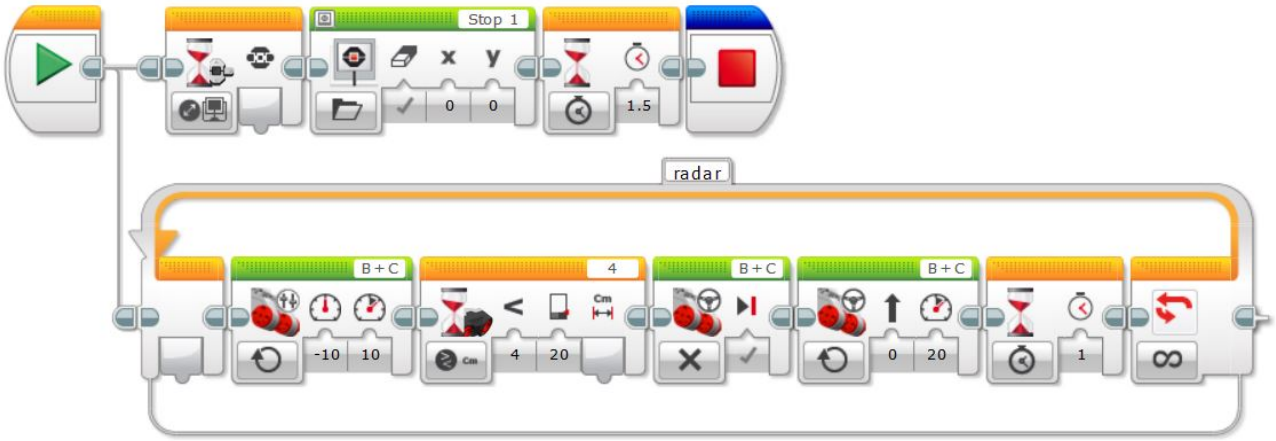
1. *Írj programot, melynek hatására robotod a középső gomb megnyomásakor kirajzol a képernyőre az EV3 ikont. A bal gomb megnyomására balra mutató nyilat rajzol és így tovább, minden gomb megnyomására egy felé mutató nyilat jelenít meg!*



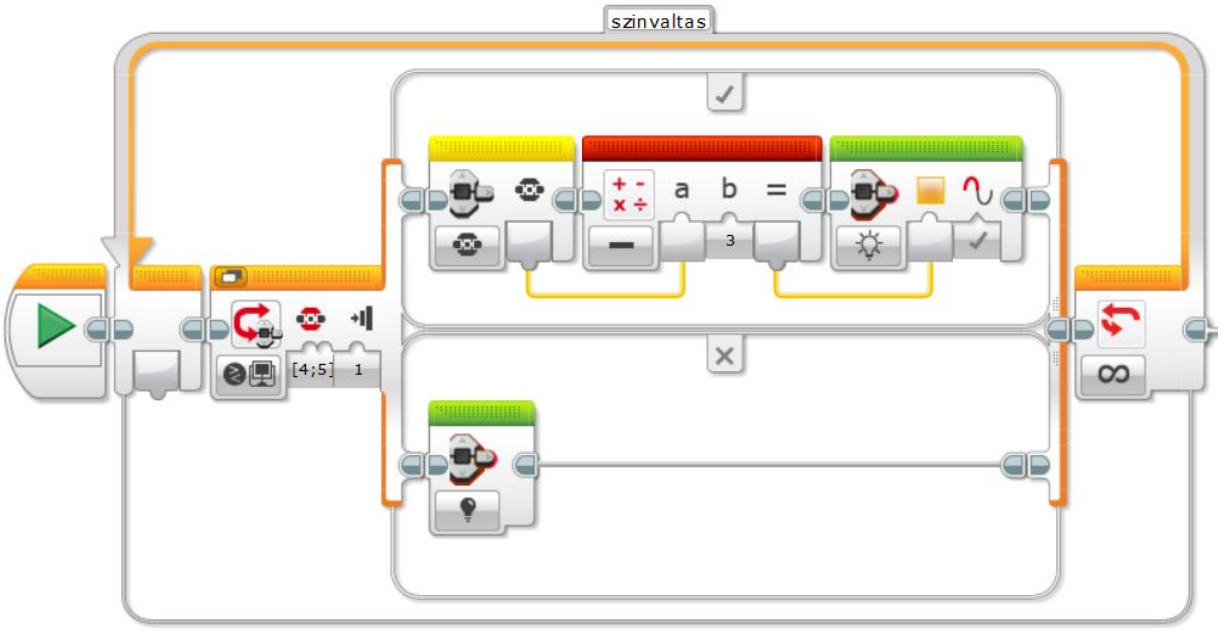
2. *Készíts programot, melyben robotod a középső gomb megnyomására kiad egy hangot röviden (ez jelzi, hogy elkezdett futni a program)! Ezek után úgy tudod mozgatni a robotot, hogyha megnyomod a felfelé gombot, robotod előre megy 2 mp-ig, a lefelé gomb hatására pedig tolat 2 mp-ig. A balra gomb lenyomására balra fordul 90°-ot, a jobb gombra pedig jobbra ugyanennyit.*



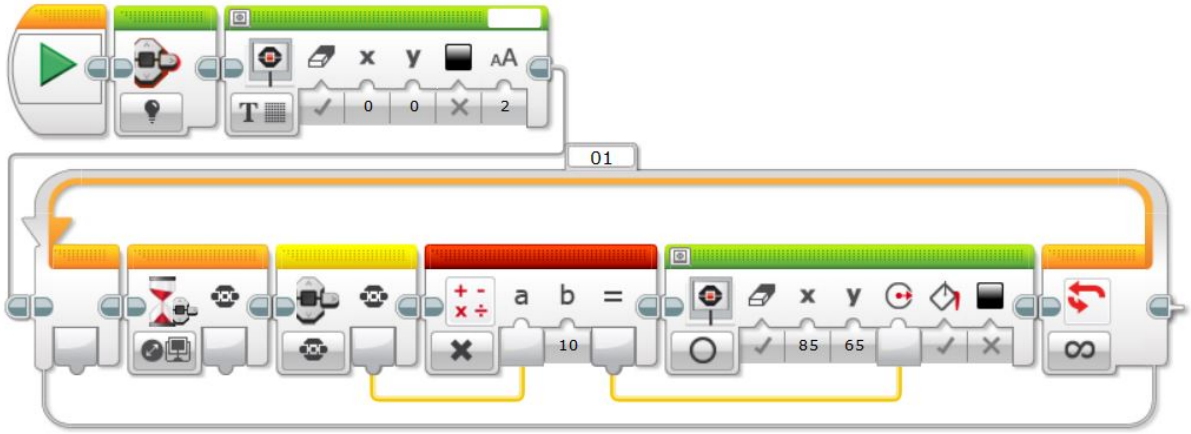
3. *Fejleszd tovább a Radar programot(alap feladatot, ne a már továbbfejlesztettet!), melyet már az előző alkalmak egyikén elkészítettél! Építsd programodba azt a funkciót, melynek hatására a tégla tetején lévő bármely gombot megnyomva robotod képernyőjén egy Stop tábla rajza jelenik meg, majd 1 másodperc múlva kilép a programból!*



4. *Készíts olyan programot, melynek során a tégla tetején lévő fölső gombot nyomva tartva a led narancssárgán, míg az alsó gombot nyomva tartva pirosan világít!*
   Ha sikerült megoldanod a feladatot, akkor keress más jó megoldást is! Programodat próbáld a gombok értékének felhasználásával megvalósítani!



5. *Valósítsd meg, hogy robotod képernyőjén egyre növekvő sugarú körök jelenjenek meg, ahogy a tégla gombjait megnyomod! A körök a következő gombnyomás sorrendre növekedjenek: bal, középső, jobb, felső, alsó!*
   Gondoskodj róla, hogy robot a program futása alatt ne világítson és a körök megjelenésekor semmi más ne legyen a képernyőn!



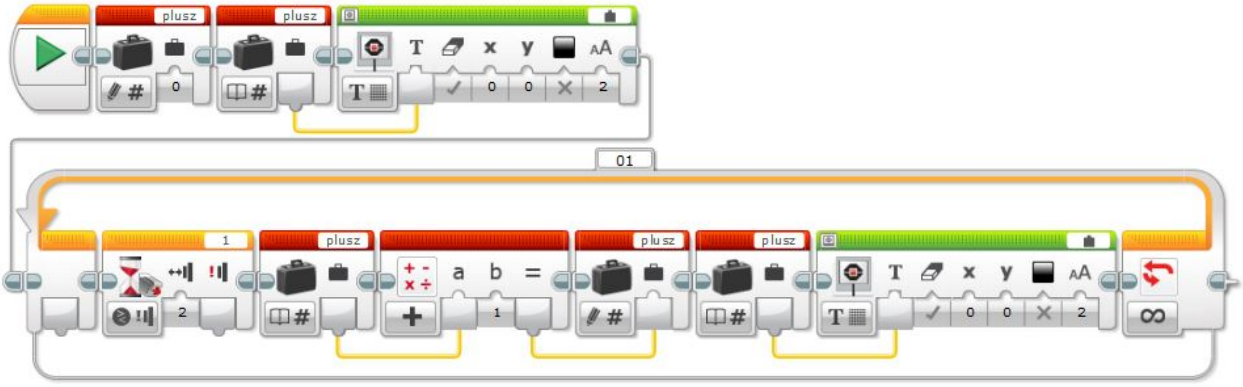
3.13.8. 8.alkalom

Változók használata

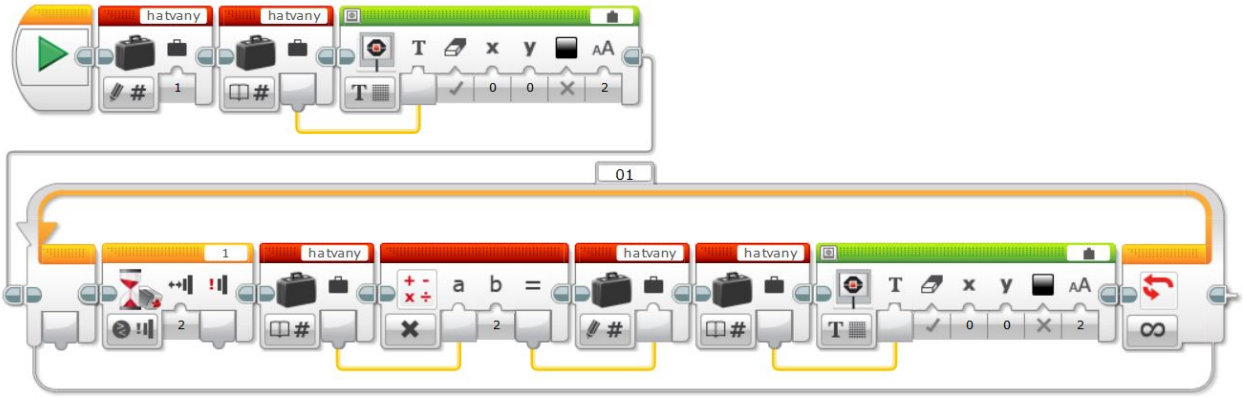
1. *Írj számsorozatokat robotod képernyőjére! A számok az érintésérzékelő megnyomásával változzanak a következő módon:*

   > *Gondoskodj arról, hogy egy megnyomásra tényleg csak egyszer végezze el robotod az adott műveltet!*

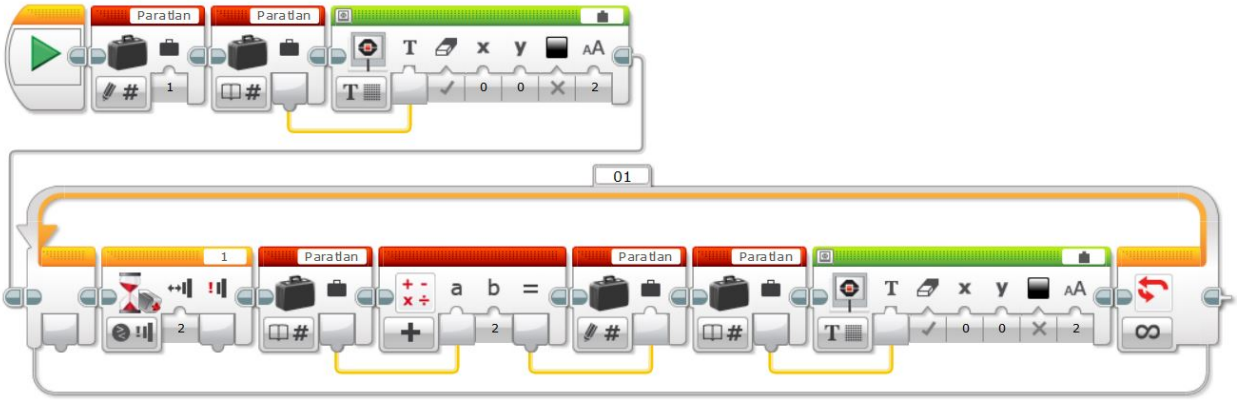
   (a) *érzékelő megnyomására + 1*



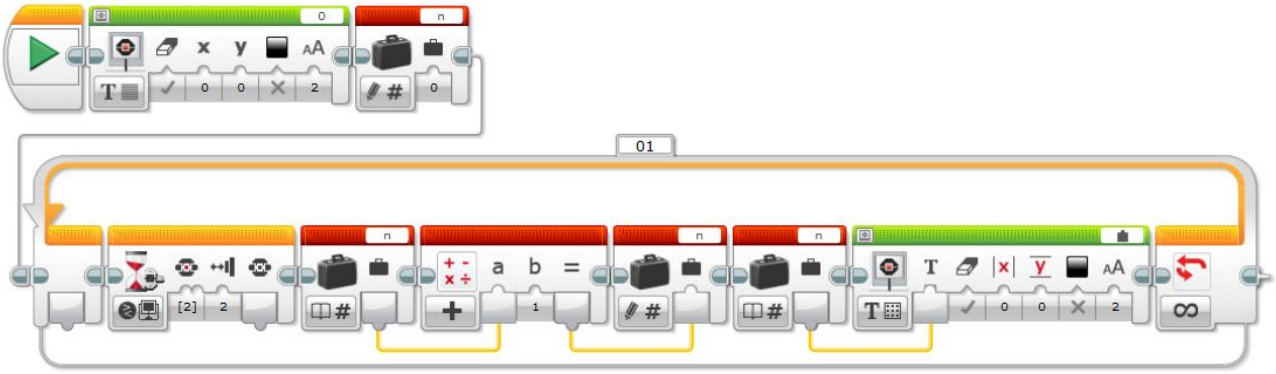
   (b) *érzékelő megnyomására * 2*



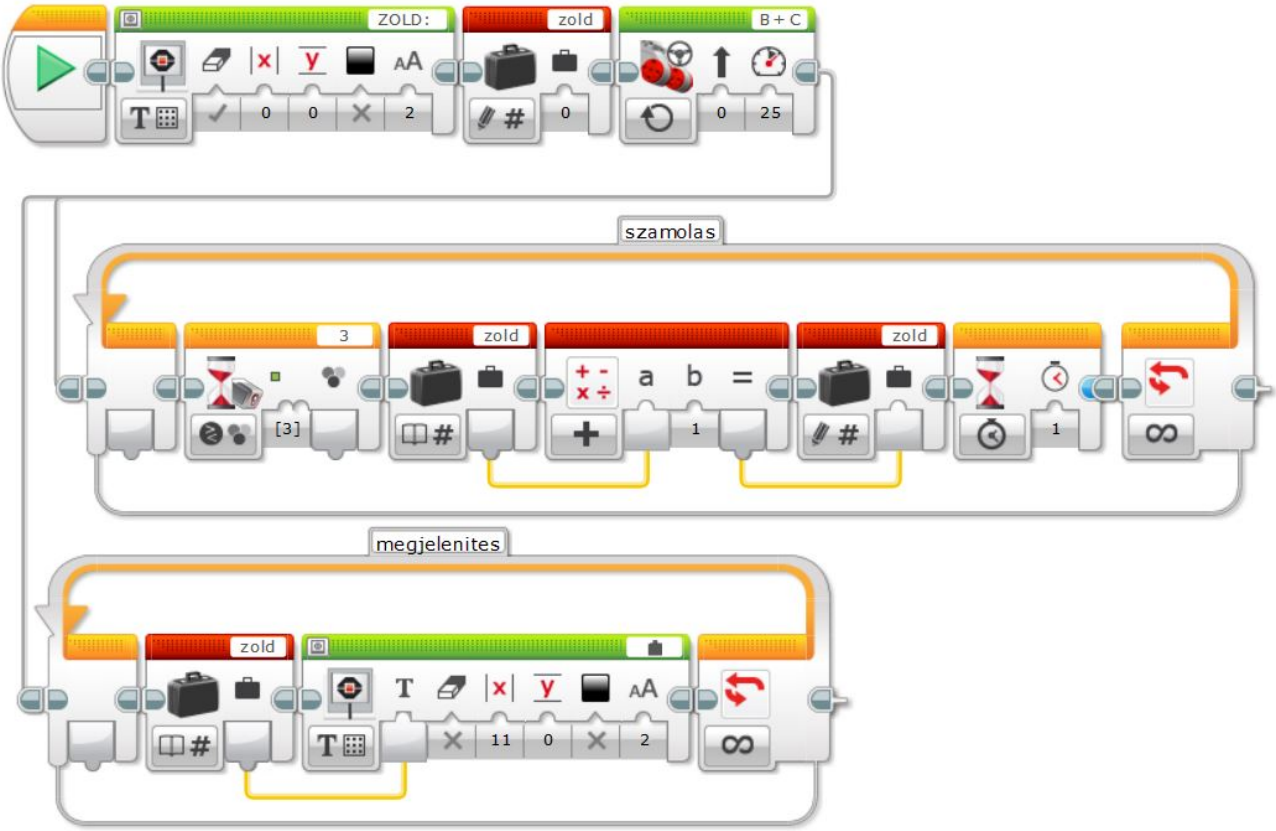
   (c) *csak páratlan számok*



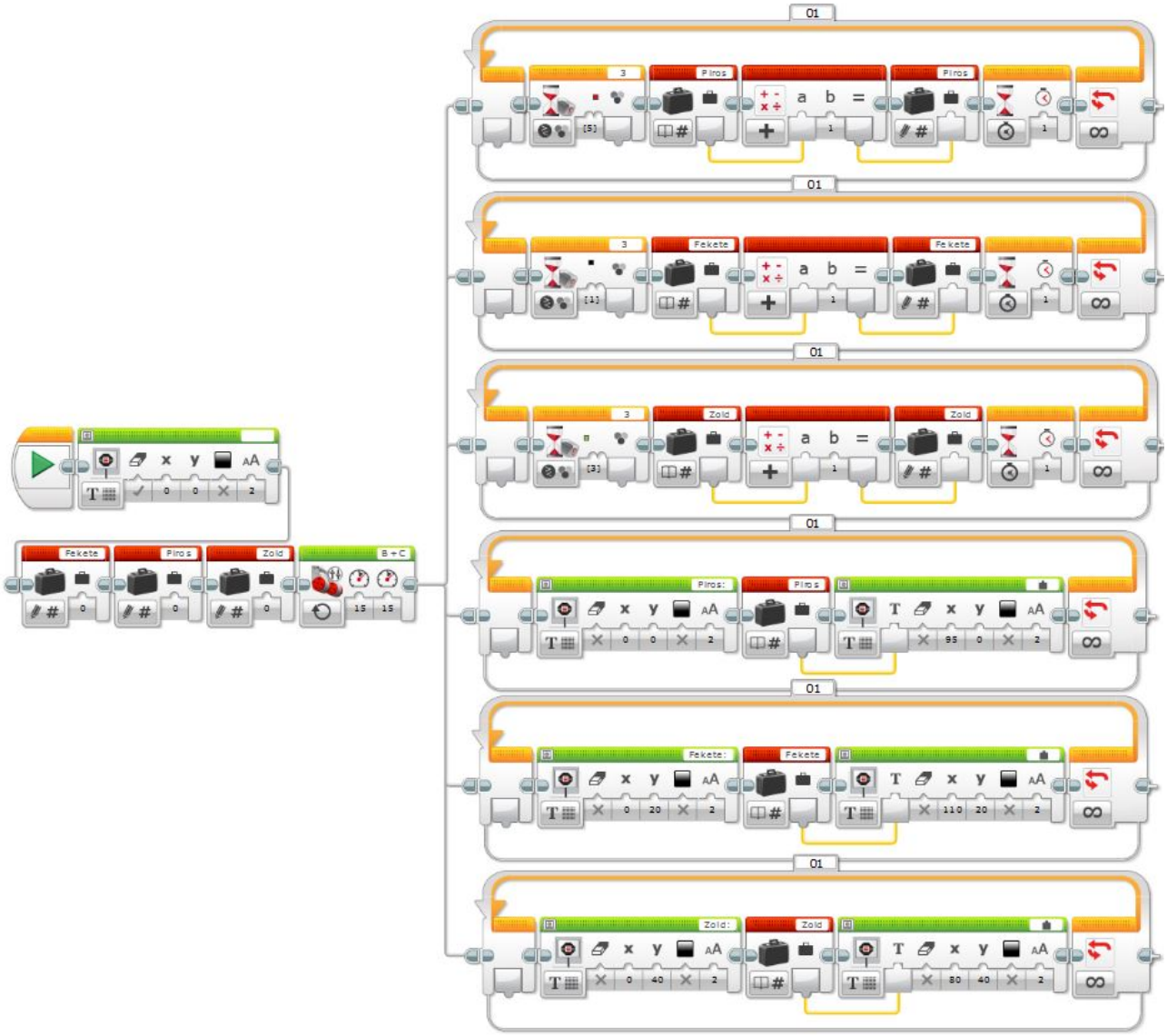
2. *Írj programot, mely kiírja a robot képernyőjére, hogy hányszor nyomtuk meg a tégla középső gombját!*



3. *Írj programot, mely megszámolja, hogy hány zöld vonal van a pályán és kiírja a képernyőre a következőképpen:*
   ZOLD: …db



4. *Írj programot, mely megszámolja, hogy hány piros, fekete, zöld vonal van a pályán és kiírja a képernyőre a következőképpen:*
   PIROS: ...db
   FEKETE: ...db
   ZOLD: ...db



5. *Készíts egy robot mérőszalagot! Mérje fel a robot, hogy előtte és mögötte az éppeni helyzetéhez viszonyítva pontosan mennyi a távolságösszeg. Vagyis mekkora a távolság az előtte és a mögötte lévő két akadály között!*