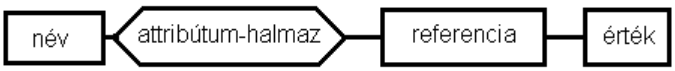
III.1.22. Változók és konstansok

A *változó* fogalma a matematikában egy értelmezési tartománnyal rendelkező, ebből bármilyen értéket felvehető objektum, melynek értéke logikailag határozatlan. Ugyanez a számítástechnikában egy memóriacímen levő memóriazónát jelent, amelynek tartalma mindig létezik, ez egy jól meghatározott érték, és fő jellemzője, hogy csak bizonyos algoritmusok által hozzáférhető és módosítható.

Egy változónak négy alapeleme van:

- név,
- attribútum-halmaz,
- referencia,
- érték.

Egy *változó neve* az illető nyelv által lexikálisan megengedett karaktersorozat, ez a változó azonosítója.



87. ábra: *Változók alapelemei*

Az *attribútum-halmaz* jellemzőket tartalmaz a változóról, például a változó típusát, a változó láthatósági területét, a változó élettartamát.

A *referencia* egy információ, amely megadja azt a fizikai vagy logikai helyet, amelynek tartalma a változó értéke.

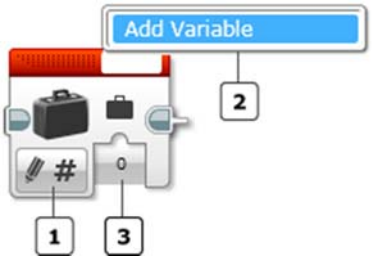
A változó negyedik alapeleme az *érték*: a program futása során a változónak ez a mezője változtatja az értékét. Egy változó értékének a kiolvasása a referencia tartalmának a kiolvasásaként történik. Egy változó értékének a megváltoztatása a referencia tartalmának felülírásaként történik. Az értékadás többnyire egy kifejezés kiértékelésének az eredménye, amely beíródik a változó referenciájának tartalmába.

Vizuális programozási nyelvekben, sajnos, kényelmetlenebb dolgozni változókkal, mint például imperatív nyelvekben.

A LEGO MINDSTORMS EV3-ban a változó a tégla memóriájának egy jól meghatározott helye, amely értéket képes tárolni. Ennek az értéknek a *szöveg*, *numerikus*, *logikai*, *numerikus tömb*, *logikai tömb* EV3 típusok valamelyike lehet a típusa.

A memóriazóna tartalmát, a változó értékét *írni* vagy *olvasni* lehet.

LEGO MINDSTORMS EV3 Home Edition-ben a változókat egy bőrönd jelképezi, a blokkon a változó nevét, típusát és értékét lehet beállítani, valamint azt, hogy írni, vagy olvasni akarjuk-e a változót.



88. ábra: *Változó*

Az 1-es *módszelektor* segítségével azt tudjuk beállítani, hogy olvasni (Read) vagy írni (Write) szeretnénk-e a változót, majd a kiválasztott módban megadhatjuk a változó típusát.

A 2-es gomb segítségével megadhatjuk az új változó nevét, vagy név szerint kiválaszthatunk egy már létező változót a listából.

A változó nevében az angol ábécé nagy és kisbetűi, valamint a szóköz, aláhúzás jel és mínusz karakterek szerepelhetnek.

Tehát egy lexikálisan elfogadott név a következő karakterekből állhat: „abcdefghiklmnopqrstvxyzABCDEFGHIKLMNOPQRSTVXYZ _-”.

A változó neve a fent említett karakterek bármelyikével kezdődhet, még szóközzel is.

Először mindig az 1-es gomb segítségével adjuk meg a típust, mert a 2-es gomb listájában csak a megfelelő típusú változók nevei jelennek meg!

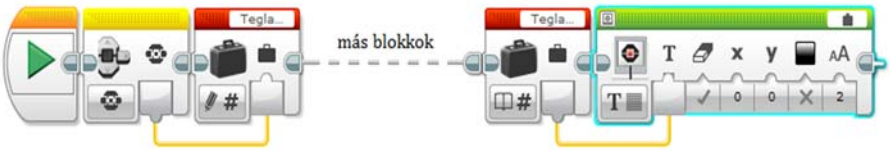
A 3-as gomb a változó értékét jelenti. Írásnál megadhatjuk ezt, olvasásnál innen olvashatjuk ki.

Ha egy változónak nem adunk értéket, a LEGO MINDSTORMS EV3 Home Edition a típusának megfelelő kezdőértékkel látja ezt el: a numerikus értékek kezdőértéke 0, a szövegeké az üres string, a logikai értékeké a false, a tömbök esetében pedig üres tömbbel inicializálja a változókat.

Akárhányszor adhatunk értéket egy változónak a program során, a változó értéke az utoljára beírt érték lesz. Az értéket megadhatjuk közvetlenül beírással, vagy adatdrót segítségével is.

Ha létrehoztunk egy változót a projekt összes programjában látható, használható és párhuzamosan elérhető lesz.

Sok esetben a változók használata megkerülhető az adatdrótok használatával, azonban, ha hosszú a program, a változók használata olvashatóbbá teszi ezt, mint egy nagyon hosszú és kusza adatdrót.

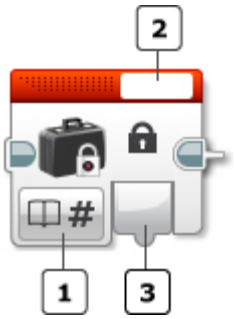


89. ábra: *Változó használata*

A változóktól eltérően a *konstansok* a program futása során megőrzik értéküket. Használatuk egyszerűbbé és kifejezőbbé teszi a programírást.

LEGO MINDSTORMS EV3 Home Edition-ben a konstansokat a változókhoz hasonló bőrönd jelképezi, ám egy lakat jelzi, hogy értéküket csak olvasni tudjuk. A konstansok érték konstansok, vagyis külön azonosítóval nem kell ellátni őket, nevük nincs, csak maga az érték jelenti a konstanst.

Ha konstanst használunk, és megváltoztatjuk a blokkon az értéket, akkor a teljes programban mindenhol megváltozik a konstans értéke.



90. ábra: *Konstans*

Az 1-es *módszelektor* segítségével a típust tudjuk beállítani. A változókhoz hasonlóan a típus *szöveg*, *numerikus*, *logikai*, *numerikus tömb*, vagy *logikai tömb* lehet.

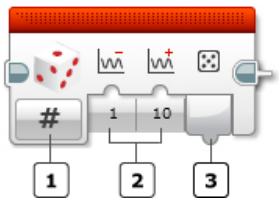
A 2-es gomb segítségével a konstans értékét adhatjuk meg.

A 3-as gombról pedig az értéket olvashatjuk vissza.

91. ábra: *Konstansok*

III.1.23. A véletlenszám generátor

A Random blokk véletlen számokat generál. Tulajdonképpen pszeudo-véletlen számokról van szó, hisz a processzor működése determinisztikus, és az előállított számok véletlenszerűek, de mégis megfelelnek bizonyos matematikai szabályoknak. Elég sokszor lefuttatva a számítást, előbb-utóbb ismétlésbe botlunk.



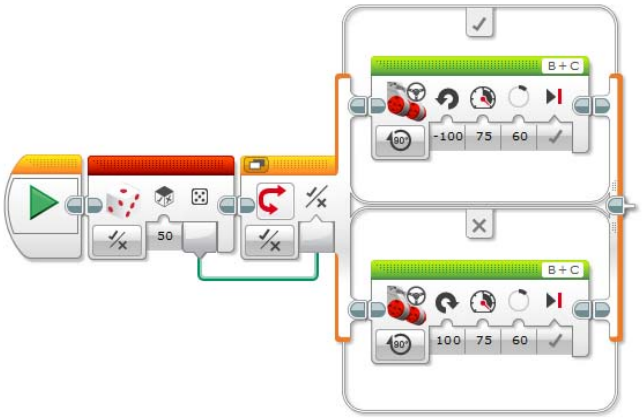
92. ábra: *Véletlenszám generátor*

Az 1-es *módszelektor* gomb segítségével a véletlen szám típusát választhatjuk ki. Ez numerikus vagy logikai lehet.

Ha numerikus típust választunk, akkor a 2-es gombok segítségével megadhatjuk annak az intervallumnak az alsó és a felső határát, amelyből a véletlen számot kérjük. A megadott tartományon belül minden egyes értéket azonos valószínűséggel választ ki a generátor, tehát a változó normális eloszlású.

Ha logikai típusra kérünk véletlen eredményt, akkor a 2-es gomb segítségével az Igaz (True) válasz valószínűségét adhatjuk meg százalékban.

A 3-as gombról a generált értéket olvashatjuk le.



93. ábra: *Jobbra vagy balra fordul?*

A 93. ábrán látható program 50%-os Igaz valószínűséggel generáltat egy véletlen logikai értéket. Ha az érték Igaz (True), akkor a robot balra, ha Hamis (False), akkor a robot jobbra tér 60 fokos szögben, 75%-os motorerővel.

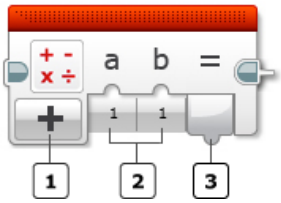
III.1.24. Műveletek

A LEGO MINDSTORMS EV3 Home Edition blokkokat biztosít a következő műveletek elvégzésére:

- matematikai;
- logikai;
- szöveg;
- tömb;
- összehasonlító;
- intervallum teszt;
- kerekítés.

Matematikai műveletek

A matematikai műveletek blokkja (Math) egyszerű matematikai műveleteket végez el a megadott bemeneten, az eredményt pedig megjelenteti a kimeneten.



94. ábra: *Matematikai műveletek*

Az 1-es *módszelektor* gomb segítségével a műveletet választhatjuk ki, ez összeadás, kivonás, szorzás, osztás, abszolút érték, négyzetgyök, hatványozás lehet, illetve az ADV lehetőség kiválasztásával tetszőleges, legtöbb négy változót használó matematikai kifejezést is megadhatunk.

Ha összeadást, kivonást, szorzást, osztást, vagy hatványozást választunk, akkor a 2-es gomb segítségével megadhatjuk a műveletekhez szükséges két operandust.

Az abszolút érték és a négyzetgyökvonás egy operandust vár.

A 3-as gombon a művelet eredményét kapjuk meg.

Érdekességként megjegyezzük, hogy a nem értelmezett műveletek (pl. zéróval való osztás, negatív számból gyökvonás stb.) eredménye egy hiba, ám ezt a hibát, ha bemenetként adjuk meg egy másik blokk számára, akkor az 0-nak értelmezi és veszi.

Érdekes például az is, hogy $-1 \times 0 = -0$.

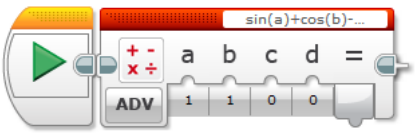
ADV módban legtöbb négy változós matematikai kifejezéseket adhatunk meg. Az összeadás, kivonás, szorzás, osztás, maradékképzés, előjelváltás műveletek mellett kerekítő függvényeket (Floor, Ceil, Round), abszolút értéket, tízes és természetes alapú logaritmusokat, szinusz, koszinusz, tangens, arkusz-szinusz, arkusz-koszinusz, arkusz-tangens szögfüggvényeket, valamint négyzetgyökvonást is használhatunk. Zárójelezéssel megváltoztathatjuk a műveletek prioritását is.

Ami a Floor, Ceil, Round kerekítési függvényeket illeti, a következő különbségekről beszélünk:

A Ceil függvény azt a legkisebb egész számot adja vissza, amely nem kisebb az argumentumnál (*ceiling*: mennyezet). Például: Ceil (3,1) = 4; Ceil(5) = 5; Ceil(–3,9) = –3. A függvény tehát felfelé kerekít minden esetben.

A Floor függvény azt a legnagyobb egész számot adja vissza, amely nem nagyobb az argumentumnál (*floor*: padló). Például: Floor(3,9) = 3; Floor(5) = 5; Floor(–3,1) = –4. A függvény tehát lefelé kerekít minden esetben. Ez a függvény megfelel a matematika *egészrész* függvényének.

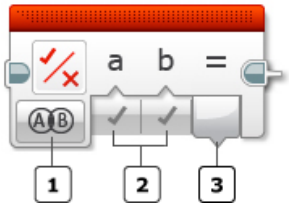
A Round függvény az argumentumként megadott kifejezést a legközelebbi egészre kerekíti, vagyis …,5 alatt lefelé, …,5 felett felfelé kerekít.



95. ábra: *Tetszőleges matematikai kifejezés*

Logikai műveletek

A logikai műveletek blokk (Logic Operations) az És (And), Vagy (Or), Kizáró vagy (Xor) és Nem (Not) logikai műveletek elvégzésére szolgál.



96. ábra: *Logikai műveletek*

Az 1-es *módszelektor* segítségével a megfelelő műveletet választhatjuk ki. Az És, Vagy, valamint Kizáró Vagy műveletek kétoperandusúak, a Nem egyoperandusú. Az operandusokat a 2-es gombok segítségével lehet megadni, a 3-as gombon pedig megkapjuk a művelet eredményét.

A négy logikai műveletet a következő művelettáblák írják le. Az És eredménye akkor Igaz, ha mindkét operandus Igaz, a Vagy eredménye akkor Hamis, ha mindkét operandus Hamis, a Kizáró Vagy akkor Igaz, ha csak az egyik operandus Igaz, A Nem pedig megfordítja az argumentuma igazságértékét.

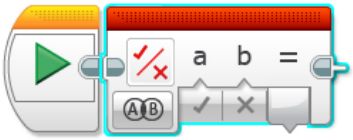
A	B	A És B
Igaz	Igaz	Igaz
Igaz	Hamis	Hamis
Hamis	Igaz	Hamis
Hamis	Hamis	Hamis

A	B	A Vagy B
Igaz	Igaz	Igaz
Igaz	Hamis	Igaz
Hamis	Igaz	Igaz
Hamis	Hamis	Hamis

A	Nem A
Igaz	Hamis
Hamis	Igaz

A	B	A Kiz. Vagy B
Igaz	Igaz	Hamis
Igaz	Hamis	Igaz
Hamis	Igaz	Igaz
Hamis	Hamis	Hamis

22. táblázat: *Logikai műveletek igazságtáblázatai*

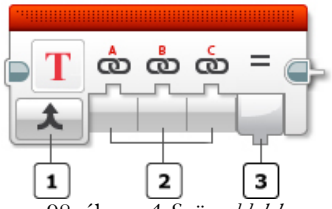


97. ábra: *Az És művelet*

Szövegműveletek

Szövegekkel az összefűzés (konkatenálás) művelete végezhető el.

A Szöveg (Text) blokk legtöbb három szöveget tud egy szöveggé fűzni úgy, hogy egymás után másolja a karakterláncokat.



98. ábra: *A Szöveg blokk*

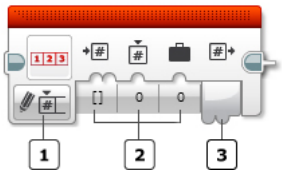
Az 1-es *módszelektor*nak itt igazán nincs is szerepe, mert más művelet nem választható ki, a 2-es gombokon a legtöbb 3 argumentum adható meg, a 3-as gomb pedig az eredményt, az összefűzött karakterláncot, szöveget adja vissza.

99. ábra: *Szövegek összefűzése*

Tömbműveletek

A Tömbműveletek (Array Operations) blokk segítségével tömbökhöz tudunk elemet adni, egy adott indexű elemet tudunk írni vagy olvasni, illetve tömbök hosszát tudjuk megállapítani.

A műveleteket numerikus vagy logikai elemeket tartalmazó tömbökön tudjuk elvégezni.



100. ábra: *Tömbműveletek*

Az 1-es *módszelektor* segítségével választhatjuk ki a kívánt műveletet (hozzáadás, írás, olvasás, hossz), valamint azt, hogy milyen típusú (numerikus, logikai) tömbökkel dolgozunk.

A 2-es gomb segítségével a bemeneti paramétereket adhatjuk meg.

Például, ha a hozzáadást választjuk, akkor meg kell adjunk egy tömböt, valamint egy elemet, amelyet hozzáadunk a tömbhöz. Ha a bemeneti tömböt adatdrót segítségével adjuk meg, akkor az nem változik a hozzáadás során, hanem egy új tömböt hoz létre az eredeti tömb alapján, amelyhez hozzáadja a megadott értéket. Ha egy adott indexű elemet akarunk kiolvasni a tömbből, akkor megadjuk a tömböt, valamint a kívánt indexet, az eredmény pedig az adott indexű elem értéke lesz. Ha írni akarunk egy kívánt indexű elemet, akkor megadjuk a tömböt, az indexet, valamint az új értéket, amit beleír az eredmény tömbbe. A tömb hosszánál megadjuk a tömböt, és a blokk visszatéríti ennek a hosszát.

A 3-as gombon kapjuk meg az eredményt.

A LEGO MINDSTORMS EV3 Home Edition 0 indexalapú tömbökkel dolgozik, vagyis egy *n* elemű tömb utolsó elemének az indexe $n-1$.

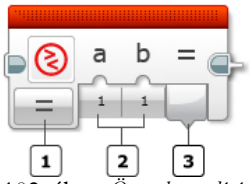
Egy üres tömb hossza 0.

Ha nem létező indexet adunk meg, a tégla hibát jelez.

101. ábra: *Adott indexű elem olvasása*

Összehasonlító műveletek

Az összehasonlító műveletek (Compare) blokk segítségével eldönthetjük, hogy két érték egyenlő, nem egyenlő, kisebb, nagyobb, kisebb vagy egyenlő, nagyobb vagy egyenlő.



102. ábra: *Összehasonlítás*

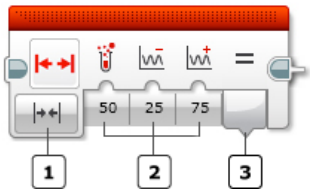
Az 1-es *módszelektor* a hat összehasonlító művelet valamelyike lehet (egyenlő, nem egyenlő, kisebb, nagyobb, kisebb vagy egyenlő, nagyobb vagy egyenlő), a 2-es gomb segítségével a két argumentumot (összehasonlítandó értéket) adjuk meg, a 3-as gomb pedig logikai értékként (Igaz vagy Hamis) visszatéríti az eredményt.

Mód	**Jelentés**	**Bemenet**	**Kimenet**
=	egyenlő	a, b	Igaz, ha a = b
≠	nem egyenlő	a, b	Igaz, ha a ≠ b
>	nagyobb	a, b	Igaz, ha a > b
<	kisebb	a, b	Igaz, ha a < b
≥	nagyobb vagy egyenlő	a, b	Igaz, ha a ≥ b
≤	kisebb vagy egyenlő	a, b	Igaz, ha a ≤ b

23. táblázat: *Összehasonlító műveletek*

Intervallum teszt

Az Intervallum teszt (Range) blokk ellenőrzi, hogy egy megadott szám egy megadott intervallumon belül vagy kívül esik. Az intervallumot az alsó és a felső határának megadásával tudjuk specifikálni.



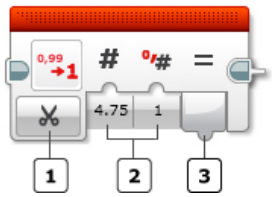
103. ábra: *Intervallum teszt*

Az 1-es *módszelektor* segítségével megadhatjuk, hogy a határokon belül vagy kívül akarunk-e tesztelni. A 2-es gomb segítségével az értéket valamint az alsó és a felső határt adhatjuk meg, a 3-as gombon Igaz vagy Hamis értékkel megkapjuk a teszt eredményét.

A teszt alul, felül zárt intervallumot vesz. Tehát például az $50 \in [50, 75]$ teszt eredménye Igaz (True).

Kerekítés

A Kerekítés (Round) blokk különböző kerekítési módszereket implementál. Egy tizedes számot egésszé kerekíthetünk le, fel vagy a legközelebbi egészhez, illetve megadott tizedesre kerekíthetünk segítségével.



104. ábra: *Kerekítés*

Az 1-es *módszelektor* segítségével a kerekítés módját adhatjuk meg. Ez a legközelebbi egészhez (To Nearest – Round), felkerekítés (Roud Up – Ceil), lekerekítés (Round Down – Floor), vagy tetszőleges tizedesre való kerekítés (Truncate).

A 2-es gomb a bemeneti érték, valamint Truncate esetében a kívánt tizedesek száma is.

A 3-as gombon az eredményt kapjuk meg.

Bemenet	Tizedesek száma	Kimenet
1,253	0	1
1,253	1	1,2
1,253	2	1,25
1,253	6	1,253

24. táblázat: *Példa a Truncate-ra*

Kovács Lehel István