



ESZTERHÁZY KÁROLY FŐISKOLA
MATEMATIKAI ÉS INFORMATIKAI INTÉZET

RoboLOGO projekt: Egy oktatási célú mobil robot koncepciója

Készítette:

Magyar Péter
informatikatanár MA 3. évf.
Gregus Tamás
informatikatanár MA 1. évf.

Témavezető:

Dr. Geda Gábor
főiskolai docens
Fazekas István
villamosmérnök

EGER, 2012

Tartalomjegyzék

1. Bevezetés, problémafelvetés	5
2. A LOGO programozási nyelv	7
2.1. Története	7
2.2. Általános jellemzői	7
2.3. Működése, használata	7
2.4. Főbb parancsok	8
2.5. Tanulási környezetek	9
2.5.1. Comenius Logo	9
2.5.2. Imagine Logo	10
2.5.3. LibreLogo	10
3. Néhány általános és középiskolai oktatásban használt eszköz bemutatása	12
3.1. Interaktív tábla	12
3.2. LEGO Mindstorms NXT	13
3.2.1. Hardver	13
3.2.2. Szoftver	17
3.2.3. Versenyek	19
3.2.4. Problémák a LEGO robot készletével	20
3.3. Arduino	21
3.3.1. Az Arduino mikrokontroller jellemzői	21
3.3.2. Felhasználási lehetőségei	22
3.3.3. Az Arduino hardvere	22
3.3.4. Szoftver	23
3.3.5. „Üzembe helyezés”, az IDE telepítése, kezdő lépések	25
3.3.6. Egy konkrét példa programozás oktatására	26
3.3.7. Az Arduino előnyei, hátrányai, lehetséges helye a közoktatásban	30
4. RoboLOGO projekt	31
4.1. A teknősök történelme	31
4.2. Egy oktatási célú mobil robot koncepció bemutatása	33

4.3.	A robot felépítése	34
4.4.	Mobil fejlesztői környezet - Android alkalmazás	36
4.4.1.	Felépítése	38
4.4.2.	Terméktámogatás	38
4.5.	Fejlesztői környezet - asztali alkalmazás	39
4.5.1.	Felépítése	40
5.	Iskolai tapasztalatok a RoboLOGO-val	41
6.	Jövőbeli tervek	46

Köszönetnyilvánítás

Ezúton mondunk köszönetet oktatóinknak: Dr. Geda Gábornak, Dr. Kovács Emődnek és Fazekas Istvánnak, valamint a projektben résztvevő programtervező informatikus BSc szakos hallgatótársainknak: Adamkó Dánielnek, Kiss Gergőnek, Bakos Dávidnak, Bezecski Bencének és Kozma Patriknak a projekt megvalósításához nyújtott segítségükért, hasznos szakmai tanácsaikért.

1. fejezet

Bevezetés, problémafelvetés

Maga az informatika tudományterület rohamosan fejlődik, de úgy érezzük, hogy oktatásmódszertana nem tartja vele kellő mértékben a lépést. Mivel ez a tantárgy a többihez viszonyítva fiatalnak tekinthető, még a tanárképzésből frissen kikerülő oktatók módszertani felkészültsége, képzettsége sem feltétlenül felel meg teljeskörűen a kor elvárásainak. Sok esetben hiányzik a tantárgy által közvetített tudás felhasználásában rejlő lehetőségek meglátása, megláttatása.

A 2012-es NAT [1] ráadásul jelentős mértékben lecsökkenti az informatikaórák számát, így még kevesebb lehetőségük van a diákoknak az általános és középiskolai fejlesztési feladatok elsajátítására. Ezek 6 téma köré csoportosulnak:

1. Az informatikai eszközök használata
2. Alkalmazói ismeretek
 - a) Írott és audiovizuális dokumentumok elektronikus létrehozása
 - b) Adatkezelés, adatfeldolgozás, információmegjelenítés
3. Problémamegoldás informatikai eszközökkel és módszerekkel
 - a) A probléma megoldásához szükséges módszerek és eszközök kiválasztása
 - b) Algoritmizálás és adatmodellezés
 - c) Egyszerűbb folyamatok modellezése
4. Infokommunikáció
 - a) Információkeresés, információközlési rendszerek
 - b) Az információs technológián alapuló kommunikációs formák
 - c) Médiainformatika

5. Az információs társadalom

- a) Az információkezelés jogi és etikai vonatkozásai
- b) Az e-szolgáltatások szerepe és használata

6. Könyvtári informatika

Ezen ismereteket passzív befogadói a tanulók, nincs szükségük, illetve kevés lehetőségük van egyéni kreativitásuk kibontakoztatására. Ezekből kifolyólag a diákok motiválatlanok, elvesztik kezdeti érdeklődésüket az informatika tárgy iránt és nem látják az órákon megszerezhető ismeretek hasznosítási lehetőségét.

Ezen helyzet javítása érdekében ugyan már léteznek eszközök, melyek közül például a LEGO MINDSTORMS NXT robot készlet, az interaktív tábla és az Arduino már megjelentek a magyar oktatási rendszerben is, de még ezek teljeskörű elterjedésének is több akadálya van (anyagi okok, tanári idegenkedés és képzetlenség, időhiány, tanulói előképzettség hiánya, stb.).

A LOGO programozási nyelvvel [2] már általános iskola 3. osztályában elkezdnek megismerkedni a gyerekek az informatika órákon vagy szakkörökön. E nyelv felhasználásával fejlesztik az algoritmikus gondolkodásmódot [3], az algoritmizálási készséget és megtanulják a programozás alapjait. Olyannyira beépült az általános és középiskolai informatikaoktatásba, mint tananyag, hogy egészen a középiskola 10. osztályáig is előkerülhet az órákon. Sőt, mitöbb, minden évben országos versenyt építenek rá, melyet Logo Országos Számítástechnikai Tanulmányi Versenynek [4] neveznek és 3-10. osztályig négy kategóriában rendezik meg, 2-2 évfolyam számára összevontan. A tehetségesebb és érdeklődőbb gyerekek (akik otthon is gyakorolnak, új dolgokat, ötleteket próbálgatnak ki segítségével) hamar eljutnak arra a szintre, hogy a LOGO nyújtotta lehetőségek már nem foglalkoztatják őket, „kinövik” a LOGO-t vagy egyszerűen csak unják már a „rajzolgatást”. És ezzel nem kell várni a középiskoláig, már általánosban is gyakran előfordul. Azonban a tanárnak egy egész osztályt kell tanítania, nem csak azt a pár tehetséges tanulót, a kevésbé érdeklődő, tehetséges, gyengébb képességű gyerekeket is el kell juttatnia egy bizonyos szintre.

2. fejezet

A LOGO programozási nyelv

2.1. Története

A LOGO programozási nyelv megalkotása Seymour Papert és Wally Feurzeig nevéhez fűződik, akik az amerikai Massachusetts Institute of Technology egyetem és kutatóközpont berkein belül fejlesztették ki 1967-ben. Őse a LISP nyelv volt, mely 1958-ban jelent meg és mesterséges intelligencia kutatásokhoz használták. A nyelv kialakításának célja, hogy az ember és gép közötti kommunikációt biztosítsák, ezáltal minél eredményesebben taníthassák meg kezdőknek is a számítógép működésének logikáját.

2.2. Általános jellemzői

- Gyermekközpontú
- Fejleszti az algoritmikus gondolkodásmódot
- Programozás oktatásra széles körben alkalmazható
- Gyors visszacsatolást biztosít a diákok számára

2.3. Működése, használata

A LOGO nyelv egy későbbi kiegészítése az ún. teknőcgrafika, mely manapság a többségnek egyet jelent a LOGO nyelvvel. Ennek az az oka, hogy a fejlesztők igyekeztek a LOGO-t oktatásközpontúvá tenni, így megjelenhetett a különféle tanulási környezetekben, mint szemléltető eszköz.

A programozás és algoritmizálás tanulás során a diákok a képernyőn egy rajzoló teknőst irányítanak a megfelelő parancsszavak segítségével és újak tanításával (eljárások).

2.4. Főbb parancsok

A LOGO nyelvben a következő fontosabb utasítások [5] kaptak helyet:

- előre szám:
Előreküldi a teknőcöt „szám” értékkel. Az értékek pixelben értendők. Pl.: előre 100, ekkor 100 pixelt halad előre a teknőc.
- hátra szám:
Hátraküldi a teknőcöt „szám” értékkel.
- jobbra szám:
A teknőcöt elforgatja a megadott fokszámmal.
- balra szám:
Balra forgatja el a teknőcöt a megadott fokszámmal.
- törölkép:
Letörli a vászont, visszaállítja a teknőcöt a kiindulási helyére.
- tollatfel:
A teknőc felemeli a tollat, „tollatle” utasításig nem rajzol az útja során.
- tollatle:
A teknőc leteszi a tollat, „tollatfel” utasításig rajzol az útja során.
- ha logikai [utasításlista]:
Ha a logikai feltétel teljesül, az utasításlista végrehajtódik. Pl.: ha $1 < 2$ [előre 1], ha 1 kisebb, mint 2, akkor előre megy 1-et a teknőc.
- ismétlés szám [utasításlista]:
Ismétli „számszor” az utasításlistában lévő utasításokat. Pl.: ismétlés 4[előre 1], ekkor négyszer megy előre 1-et a teknőc.
- eljárás név utasításlista vége:
Ezzel egy eljárást (képeséget) adhatunk a teknőcnek. Pl.: megtanítjuk neki a négyzet rajzolás képeségét:
? eljárás négyzet
> ism 4[előre 100 jobbra 90]
> vége
A ?, > karakterek az Imagine Logo szövegbeviteli mezőjében jelennek meg, külön beírni nem kell őket.
Elindítani a betanított utasítást a nevével tudjuk: Pl.: négyzet, ezzel a teknőc végrehajtja a négyzet nevű eljárást.

Ez a parancskészlet már elegendő ahhoz, hogy különféle szabályos alakzatokat egyszerűen megrajzolhassunk.

2.5. Tanulási környezetek

Olyan fejlesztői eszközökről van szó, melyek implementálják a technócgrafika modult és már ismert funkciókkal segítik a programozás és algoritmizálás tanítását az iskolákban.

Ismertebb környezetek:

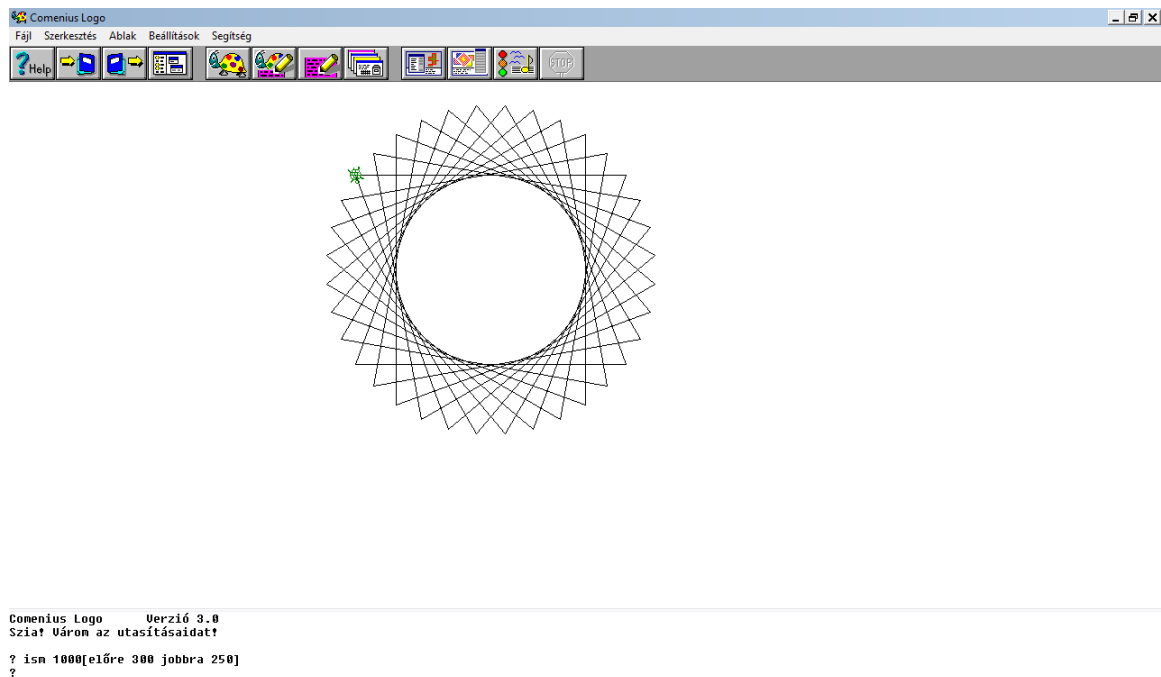
- UCBLogo
- Apple Logo
- MSWLogo
- StarLogo
- Comenius Logo
- Imagine Logo
- LibreLogo

Ezek közül Magyarországon a Comenius Logo és az Imagine Logo terjedtek el, a LibreLogo pedig egy új, ígéretes magyar kezdeményezés.

2.5.1. Comenius Logo

Ezt a környezetet a szlovák Comenius Egyetemen Andrej Blaho, Ivan Kalas és Péter Tomcsányi hozták létre. Nevét az egyetem névadójáról, Comeniusról kapta, aki a 16. század egyik meghatározó pedagógusa és írója volt. Széleskörű elterjedése és sikere annak köszönhető, hogy számtalan nyelvre lefordították az utasításkészletét, így a diákok nem csak angol, hanem az anyanyelvüknek megfelelő parancsszavakkal is irányíthatták a teknőcöt. Számtalan irodalom is foglalkozott a Comenius Logo-val, animációkészítést [6], egyszerű programok készítését [7], rekurziót és eljárásokat [8], LOGO függvényeket, módszertani tanácsokat [9] sajátíthatnak el a diákok és a tanárok a segítségükkel.

A magyar fordítás az ELTE TeaM labor jóvoltából készülhetett el, melyet a Kossuth Kiadó Rt. forgalmaz. Lásd a [2.1.](#) ábrán.



2.1. ábra. Comenius Logo

2.5.2. Imagine Logo

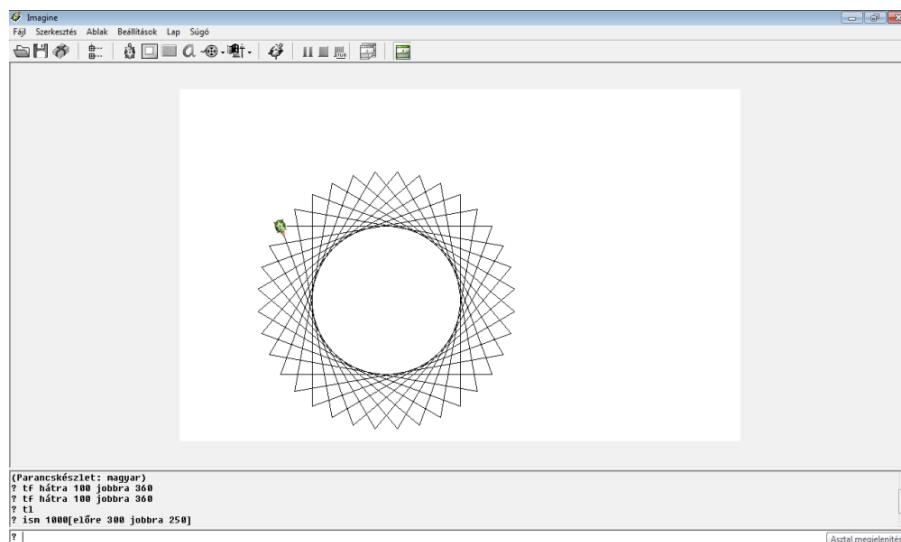
Hasonlóan az elődjéhez, az Imagine Logo is a Comenius Egyetem produktuma, amely már jóval több lehetőséget biztosít a tanulók játékos oktatásához. Az Imagine Logo objektumorientált struktúrát tartalmaz, mely másfajta programozási szemléletet követ meg a diákoktól [10]. Tartalmazza még a LogoMotion nevű grafikai programot is [11], mellyel animációkat, rajzokat tudunk létrehozni, valamint képeket tudunk átszerkeszteni. Bár sok előnye van az új, továbbfejlesztett környezet használatának, de a Comeniusban fejlesztett alkalmazást nem minden esetben tudjuk használni az Imagine alatt. Szintén irodalmak dolgozták fel a képességeit

A program megtekinthető a 2.2. ábrán. Az új és a megváltoztatott parancsok ismerhetők meg az alábbi linken:

<http://bin.sulinet.hu/inform/comlogoimagine/>

2.5.3. LibreLogo

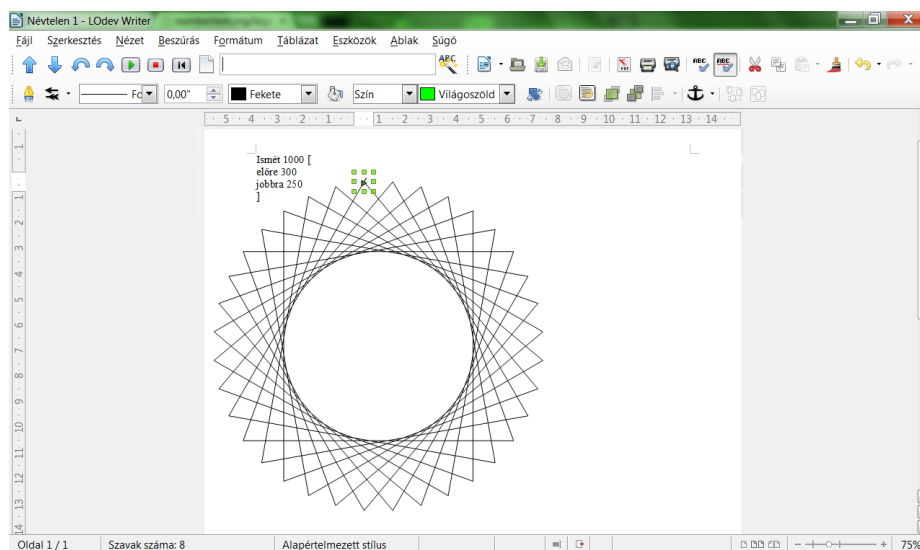
A LibreOffice nevű szabad irodai programcsomag egyik új kiegészítéseként látott napvilágot a magyar, Németh László által fejlesztett LibreLogo programozási környezet, melyben a csomag Writer alkalmazásában rajzolhatunk. Az így készült képek szöveg- és kiadványszerkesztéshez használhatóak fel. A technóc programozása és a különféle adatszerkezetek létrehozása és használata Python nyelven történhet, mely elősegíti a diákok könnyebb felkészülését az emelt szintű informatika érettségire is, melyben 2012-től a



2.2. ábra. Imagine Logo

Python nyelv is választható. A LibreOffice 4.0 megjelenésével megszűnnek a bővítmények, a LibreLogo pedig részévé válik az irodai programcsomagnak.

A program a 2.3. ábrán látható. A LibreLogo kézikönyvével és bemutató füzetével [12] elsajátíthatjuk a nyelvet és emellett sok gyakorlati példát is találunk bennük.



2.3. ábra. LibreLogo

3. fejezet

Néhány általános és középiskolai oktatásban használt eszköz bemutatása

A diákok érdeklődésének felkeltése, motiválása és az oktatás színvonalának emelése céljából Magyarországon is bevezetésre kerültek, kerülnek különböző oktatási eszközök. Az informatikaoktatáson belüli hasznosságuk ár-érték arányban nem a legmegfelelőbb, tetszetős csomagolásban jelennek meg a használhatóságuk, eredményességük korlátai. Ezek közül csak néhányat emelnénk ki.

3.1. Interaktív tábla

Az interaktív tábla egy olyan IKT¹ (3.1. ábra) eszköz, amely egy szoftver segítségével kapcsolja össze a speciális táblát a számítógéppel egy projektor közreműködésével. Ennek köszönhetően a vezérlés a tábláról történhet (toll vagy érintés segítségével) valamint a táblára került tartalmak a számítógép háttértárolójára menthetők digitálisan. Magyarországon a Smart és a Cleverboard táblák terjedtek el. Ezek a jelentős anyagi ráfordítást igénylő eszközök eredményesen az általános iskola alsóbb évfolyamain szinte minden tárgy keretein belül univerzálisan használhatóak, építve a tanulók mozgékony-ságára, játékos-ságára. Jelentősége a magasabb évfolyamokon, az életkor növekedésével egyre kisebb (a tananyag mennyisége, összetettsége miatt). Az informatikaoktatás szempontjából az interaktív tábla elég szűk szeletet vesz el. Megemelte a diákok ingerküszöbét. Felvetődik viszont a kérdés, hogy mi lesz a következő főbb motiváló tényező, ami a jövőben leválthatja, mikor már nem hat az újdonság erejével.

¹ Információs és Kommunikációs Technológia



3.1. ábra. Interaktív tábla számítógéppel és projektorral

3.2. LEGO Mindstorms NXT

A LEGO Mindstorms a LEGO cég programozható robotkészlete [13], [14], [15], [16], melyből létezik kereskedelmi és oktatási verzió is. Az aktuális NXT verzió a 2.0, mely 2009-ben jelent meg és a 2006-os 1.0 verzióhoz képest csak minimális fejlesztésekkel büszkélkedhet. Lecserélték például a fény szenzort (érzékelőt) szín szenzorra, bővült a fejlesztői környezet¹ funkcionalitása és a megvásárolható szenzorok választéka, de az alapok ugyanazok maradtak.

3.2.1. Hardver

- NXT intelligens tégl (3.2. ábra)

Egy téglatest alakú burkolat alatt lapul a készlet „agya”, egy programozható, kisméretű számítógép. Egyszerre maximum 4 szenzortól kapott értékek feldolgozására és 3 motor vezérlésére képes. Tulajdonságai:

- 32 bites ARM7 processzor
- 256 KB flash memória²
- 64 KB RAM³
- 8 bites AVR mikrokontroller⁴
- USB és bluetooth kapcsolódási lehetőség

¹ másnéven programozási környezet (IDE), ahol a programozó a programot meg tudja írni

² áramellátás után is emlékező adattár

³ olyan adattár, melynek tartalmához bármikor hozzá lehet férni és az áramellátás megszűnésekor elveszik a tartalma

⁴ szinte mindenhol megtalálható miniatűr számítógép, ilyen van a digitális karórában, a távirányítóban stb.

- beépített hangszóró
- 100 x 60 pixeles monochrome LCD kijelző
- 4 szenzor bemenet
- 3 motor kimenet
- RJ12 egyedi csatlakozók
- 6 db AA elemes vagy Li-Ion újratölthető, akkumulátoros áramellátás



3.2. ábra. LEGO MINDSTORMS NXT intelligens
tégla

– Szenzorok

Az eszközhöz a LEGO-n kívül több „külsős” cég is gyárt szenzorokat. Az NXT 2.0-ás készletben 2 darab érintésérzékelő, 1 darab ultrahangos (távolság) érzékelő és 1 darab színérzékelő kapott helyet. Ez utóbbi az 1.0-ás készlet fény és hangérzékelőjét váltotta fel.

Alapszenzorok:

- Érintésérzékelő (3.3. ábra)
Egy kétállású kapcsolóként működő szenzor, mely azt érzékeli, hogy be van-e nyomva a gomb vagy sem. Ennek megfelelően két (logikai) értéket tud a téglának továbbítani, 1-et vagy 0-át (igaz vagy hamis).
- Távolságérzékelő (3.4. ábra)
A denevérek tájékozódásához hasonlóan az ultrahangos távolságmérés elvét



3.3. ábra. Érintésérzékelő

használja ez a szenzor, mely a mért értéket cm-ben vagy hüvelykben képes megadni. Az ultrahang kibocsátása és az útjába eső tárgyról történő visszaverődése között eltelt időből számítja ki a távolság értékét.



3.4. ábra. Ultrahangos távolságérzékelő

- Fényérzékelő (3.5. ábra)

A kontrasztot, vagyis a világos és sötét árnyalatok közötti különbséget érzékeli. A mért érték nagyban függ a környezet fényviszonyaitól és a mérési felületről.



3.5. ábra. Fényérzékelő

- Hangérzékelő (3.6. ábra)
Hangintenzitást mér decibelben.



3.6. ábra. Hangérzékelő

- Színérzékelő (3.7. ábra)
A fényérzékelőtől eltérően valós RGB színlátással vérteli fel robotunkat, ugyanakkor használható fényérzékelőként is.



3.7. ábra. Színérzékelő

- Motorok (3.8. ábra)

NXT robotjaink egyszerre maximum 3 szervo motor segítségével tudnak kereken vagy láncfalpon gurulni, lépkedni vagy tárgyakat emelni. Ezen motorok elfordulásérzékelővel vannak felszerelve, így az elfordulás értéke lekérdezhető és viszonylag finom mozgás valósítható meg velük.



3.8. ábra. LEGO Mindstorms NXT szervo motor

Egyéb (nem a LEGO által gyártott) szenzorok:

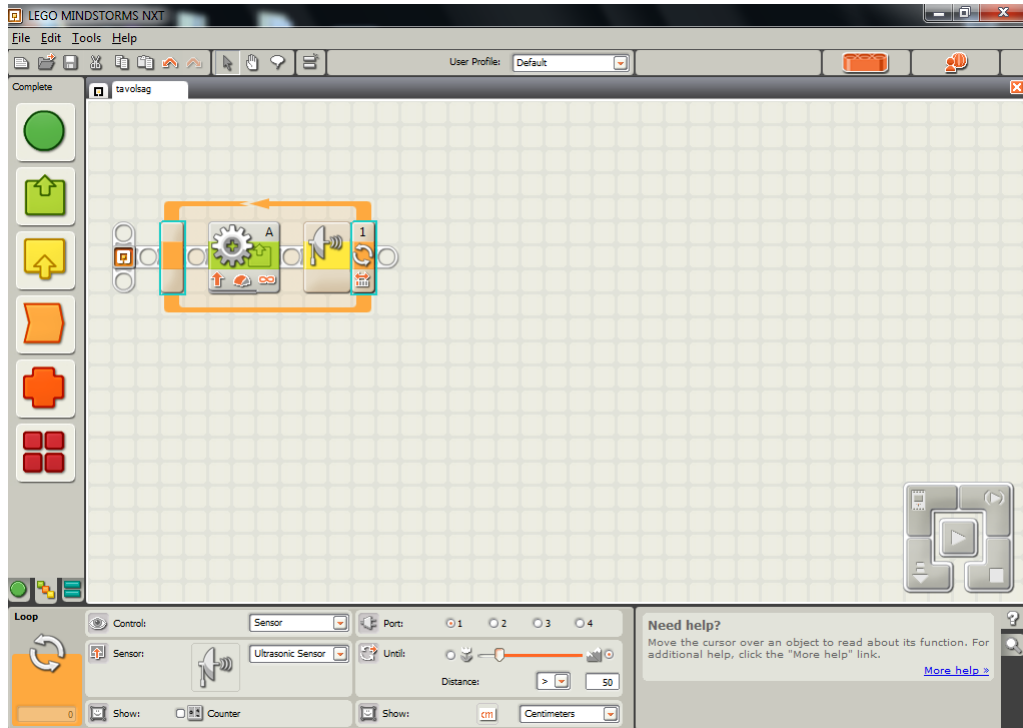
- Digitális iránytű
- Gyorsulásmérő
- Hőmérsékletmérő
- Wifi
- Infra hőérzékelő
- stb.

3.2.2. Szoftver

NXT-G. (3.9. ábra) Így hívják a hivatalos programozási környezetet, melyet a LEGO biztosít a készletéhez. A LEGO és a National Instruments közös fejlesztéséről van szó, mely LabVIEW¹ alapokra építkezik. (<http://hungary.ni.com/labview>) Az NXT-G egy vizuális programozási eszköz, benne programkód írás nélkül készíthetjük el a robotot működtető, vezérlő programokat. Ezek folyamatábra-szerűen épülnek fel, a hardver elemeknek és programozástechnológiai eszközöknek egy-egy ikont feleltettek meg, melyet a megfelelő helyre húzva az egérrel állíthatjuk össze programunkat. Az egyes elemek paraméterezése a már weboldalakon, kitöltendő űrlapokon megszokott beviteli eszközök (legördülő lista, jelölő négyzet, rádiógomb, szövegdoboz) segítségével történik. Egyszerű kezelhetőségük és korlátozásaik miatt, valamint amiatt, hogy nem

¹ a National Instruments fejlesztői környezete

kell tartani a programkód írás okozta szintaktikai hibáktól sem, már fiatal korban is könnyen elsajátítható a használata. Ugyanakkor egy már programozásban jártas diák vagy tanár számára körülményes, lassú lehet összetettebb programok készítése, ugyan- is meg kell tanulnia, hogy melyik ábra mit jelent. A különféle programozási nyelvek parancsai viszont sokszor elég hasonlóak, megkönnyítve az egyikről a másikra való át- térést.



3.9. ábra. LEGO Mindstorms NXT-G

Alternatív programozási környezetek

- Bricx Command Center [17]

Komplett, ingyenes integrált fejlesztő környezet, melynek segítségével NXC (Not eXactly C - nem pontosan/egészen C¹) nyelven programozhatjuk és vezérelhetjük NXT robotjainkat.

- RobotC [18]

A Carnegie Mellon Egyetem robotakadémiája által fejlesztett, C-re épülő programozási nyelv és a hozzá tartozó integrált fejlesztői környezet. Az NXC-vel ellentétben a teljes C nyelv funkcionalitását biztosítja, azonban sajnos fizetni kell érte. Kipróbálásra viszont létezik 30 napos próbaverzió.

¹ A C egy programozási nyelv, amivel valamilyen program megírható

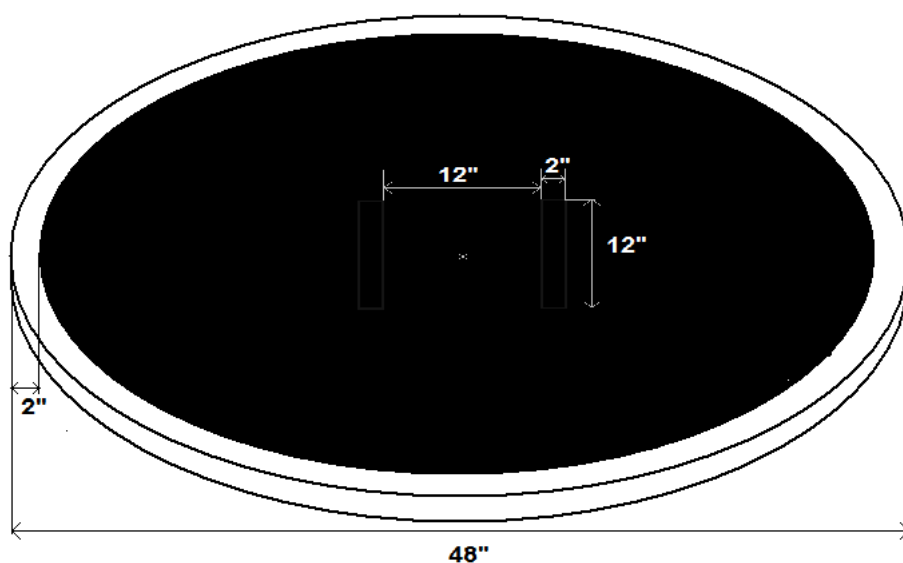
Szükségesnek érezzük megemlíteni, hogy attól függően milyen programozási környezetet szeretnénk használni a robotunk programozásához, a roboton futó firmware-t¹ le kell cserélnünk az adott fejlesztői környezethez valóra.

3.2.3. Versenyek

LEGO Szumó

A küzdelem egy 122 cm átmérőjű, matt feketére festett korong alakú küzdőtéren (3.10. ábra) zajlik, melyet egy 5 cm széles fehér gyűrű határol körkörösén. A verseny elkezdése után 60 perc áll rendelkezésükre a csapatoknak a LEGO szumó robotjaik megépítésére és felprogramozására, melynek során csak az NXT és RCX alapkészletek elemeit használhatják, a felhasználható szenzorok száma pedig korlátozva van. A küzdelem addig tart, amíg a bíró úgy nem ítéli, hogy az egyik robot elhagyta a küzdőteret (azaz a másik robotnak sikerült letolnia).

A részletes szabályok megtalálhatóak a szegedi Ságvári Endre Gyakorló Gimnázium weboldalán. [19]



3.10. ábra. LEGO Szumó küzdőtér

LEGO Robotprogramozó országos csapatverseny

A versenyen egy LEGO Mindstorms NXT robotot kell felprogramozniuk a csapatoknak annak érdekében, hogy egy előre elkészített pályán akadálykerülő és útvonalkövető feladatokat teljesítsen minél precízebben és minél rövidebb idő alatt. A csapatok a robotokat egységesen megépítve kapják, azokhoz hozzányúlni, azokat átépíteni nem

¹ itt operációs rendszer, elektronikai eszközök vezérlését végző, „beépített” program

szabad. A csapatok teljesítményét egy 3 fős szakmai zsűri értékeli.

A részletes szabályok megtalálhatóak a kecskeméti Bányai Júlia Gimnázium weboldala-lán. [20]

First Lego League

Az előző két hazai versenytől eltérően ez egy világverseny, mely több fordulóból áll, ezek:

- Regionális „hazai selejtező” verseny (Budapest és Szeged)
- Közép-Európai középdöntő
- Közép-Európai döntő
- Világdöntő

Magyarország a Közép-Európai régióba esik Németországgal, Svájjal, Ausztriával, Szlovákiával, Csehországgal, és Lengyelországgal együtt.

Minden évben van egy konkrét téma, melyhez a verseny kapcsolódik. A verseny alapvetően két részből áll:

- Kutatás

Az adott témával kapcsolatban kell egy problémát megfogalmaznia és körbejárnia a csapatnak, majd egy 5 perces prezentációban bemutatni az elvégzett kutatási munkájukat és megoldási javaslatukat.

- Robotika

Az adott témához kapcsolódóan minden évben készül egy tesztpálya, melyen a csapatok által épített és programozott LEGO Mindstorms NXT robotnak kb. 10-15 kitűzött feladatot kell végrehajtania maximum 150 másodperc alatt (átépítésekkel együtt). A zsűri pontozza a robotot, a programot és az akadálypályán való működését is.

Az értékelés 4 kategóriában történik: kutatási prezentáció, csapatmunka, robot design, akadálypályán nyújtott teljesítmény. A hivatalos weboldal itt található:

<http://firstlegoleague.org/>

3.2.4. Problémák a LEGO robot készletével

A LEGO Mindstorms NXT alapkészlet ára Magyarországon összemérhető a minimálbér összegével. Az összetettebb, érdekesebb feladatok megoldásához szükséges további kiegészítők (szenzorok) beszerzése újabb, nagyobb összegeket (10-20 ezer forint egy darab szenzor) igényel, ráadásul a LEGO nagyon ráérősen fejleszti készletét. Az utóbbi

években nincs olyan jelentős változás az árában, ahogyan ezt más informatikai eszközök esetében már megszoktuk. Ez nem kedvez az általános- és középiskolákban történő elterjedésnek. Az RJ12-es „speciális” csatlakozók zárttá teszik a rendszert, korlátozzák a bővíthetőségét, más rendszerekhez való csatlakoztatását. Az a néhány iskola, mely magáénak tudhat ilyen készleteket, általában valamilyen pályázatnak köszönheti azt, vagy alapítványi, tandíjas iskola, ugyanis saját költségvetésből szinte egy állami iskola sem engedheti meg magának. A másik nagy probléma a LEGO termékével, hogy mind a tanároknak, mind a tanulóknak meg kellene tanulniuk használni és ez nem kevés időt, energiát igényelne. Az informatika alacsony óraszámja sem kedvez ennek. Így sokan egyáltalán nem hajlandóak vagy nem látják értelmét, hogy az informatika ezen, különben igen fontos területével tanórai keretek között foglalkozzanak.

A LEGO Mindstormst viszonylag széles életkori spektrumon lehet használni. Nem annyira univerzális, viszonylag kevés tárgyhoz köthető az alkalmazása (kevesebbhez, mint az interaktív tábla). Önmagához mérten elég drága is. Nagy korlátnak látjuk azt, hogy a LEGO-hoz 4 szenzor és 3 szervo motor csatlakoztatható. Előnyére szolgál, hogy bár korlátokkal, de a robot testének megépítése időben hatékonyabban történhet, ugyanakkor mechanikai terhelhetősége nem túl jó.

Az Arduino egy kitörési pont lehetne, mert ugyan nincs elterjedve, de az elterjedésének lehetősége az anyagi elérhetősége miatt realisabb. Ráadásul rugalmasabban is alkalmazható egy bizonyos tudásszint felett.

3.3. Arduino

Nyílt forrású physical computing¹ platform [21], [22], [23], mely egy egyszerű mikrokontrolleres I/O lapkából² és a hozzá tartozó fejlesztői környezetből áll, azaz lényegében egy kis számítógép. Tudását tekintve ugyanezt tartalmazza a LEGO robotok agyát jelentő „tégla” is.

3.3.1. Az Arduino mikrokontroller jellemzői

- Nyílt forrású hardver és szoftver, ami biztosítja a könnyű hozzáférhetőséget, és a szabad módosítás, továbbfejlesztés lehetőségét
- Alacsony ára (kb. 30 euro) szintén kedvez a könnyű elérhetőségnek (alkatrészenként megvásárolva akár saját kezűleg is összerakható, amivel további árcsökkenést érhetünk el

¹ A fizikai világ érzékelése és irányítása számítógéppel

² feladata a bemenő jelek feldolgozása és egy (válasz)kimenet küldése

- A hozzá tartozó fejlesztői környezet ingyenesen letölthető
- Használata egyszerű, a számítógéppel USB porton keresztül szabványos soros kommunikációt folytat
- Multiplatform - Linux, Microsoft Windows, Apple OS X (korábban Mac OSX) operációs rendszereket futtató számítógépekkel egyaránt használható
- Elérhetőek előre megírt, használatra kész, a legkülönbözőbb hardverelemekhez létrehozott függvény-könyvtárak (library)
- Alkalmazóinak tábora egyre növekszik, világszerte aktív közösséget alkot, ami segíti az egyén munkáját

3.3.2. Felhasználási lehetőségei

Az Arduino interaktív tárgyak készítésére használható. Számtalan kapcsolót vagy szenzort bemenetként olvasva, LED-ek, motorok és egyéb kimenetek kimeríthetetlen választékát képes vezérelni. Arduino projektek állhatnak önmagukban vagy kommunikálhatnak egy számítógépen futó programmal is, melyeket különféle programozási nyelveken írhatunk meg (Processing, MaxMSP, C#, Java, stb.). Az Arduino a számítógép számára szabványos soros port interfészként¹ jelenik meg. A legtöbb modern programozási nyelv rendelkezik soros port kezelő API-val², nem kell nekünk megírniuk hozzá a rutinokat³. Ezen lehetőségek révén a felsőoktatásban már használják a robotika tantárgyak, de a középiskolákban is kezdenek felfigyelni rá. Remek lehetőség azok számára, akik szeretnének gyakorlat útján megszerzett hardver ismerethez jutni és mikrokontrolleres (AVR⁴) környezetben hardver programozással foglalkozni.

3.3.3. Az Arduino hardvere

Az Arduino (3.11. ábra) egy ATMEL⁵ AVR mikrokontroller köré épülő kis méretű nyomtatott áramkörü lapka, melyből számos különböző modell létezik. Ezek különbözőnek méretükben, alkotóelemeikben, felhasználási céljukban, árukban, stb.

Az Arduino lapkák funkcionalitása bővíthető ún. shieldek (3.12. ábra) segítségével. Ezek ad hoc⁶ tervezett nyomtatott áramkörü lapok, melyek az Arduino elveit követik: olcsó előállítási költség és egyszerű használat. Az Arduino tetejére illeszthetőek, port

¹ felületként

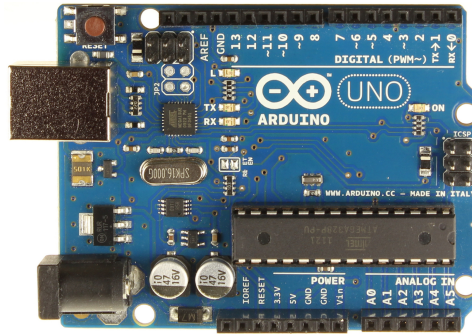
² application programming interface - alkalmazásprogramozási felület

³ eljárásokat, függvényeket, metódusokat

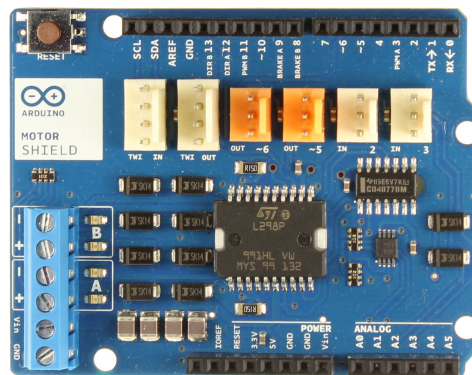
⁴ egy mikrokontroller típus

⁵ Az ATMEL az AVR típusú mikrokontrollerek gyártója

⁶ „csak úgy”, hirtelen kigondolt ötlet miatt



3.11. ábra. Arduino Uno



3.12. ábra. Arduino Motor Shield

kiosztásuk megegyezik az Arduino-éval. Sokféle shield létezik, általában mindet egy konkrét funkcióra tervezik. Néhányat maga az Arduino fejlesztő csapata készíti el, a legtöbbet azonban valamelyik third-party¹ cég. Léteznek motorvezérlő, GPS, Ethernet, Bluetooth, MP3 lejátszó, WiFi, stb. shieldek. Tehát a LEGO termékével ellentétben szinte végtelen számú bővítési lehetőség áll rendelkezésünkre, méghezozzá olcsón.

3.3.4. Szoftver

A nyílt forráskódú, szabadon (ingyen) letölthető Arduino IDE² (3.13. ábra) felülete öt egységből épül fel, ezek: szövegszerkesztő, értesítési terület, konzol, eszköztár a fontosabb funkció gombokkal, menüsor. Nem nyújt kihasználhatatlan lehetőségeket, képességei viszont bőven elegendőek komolyabb programok írásához is.

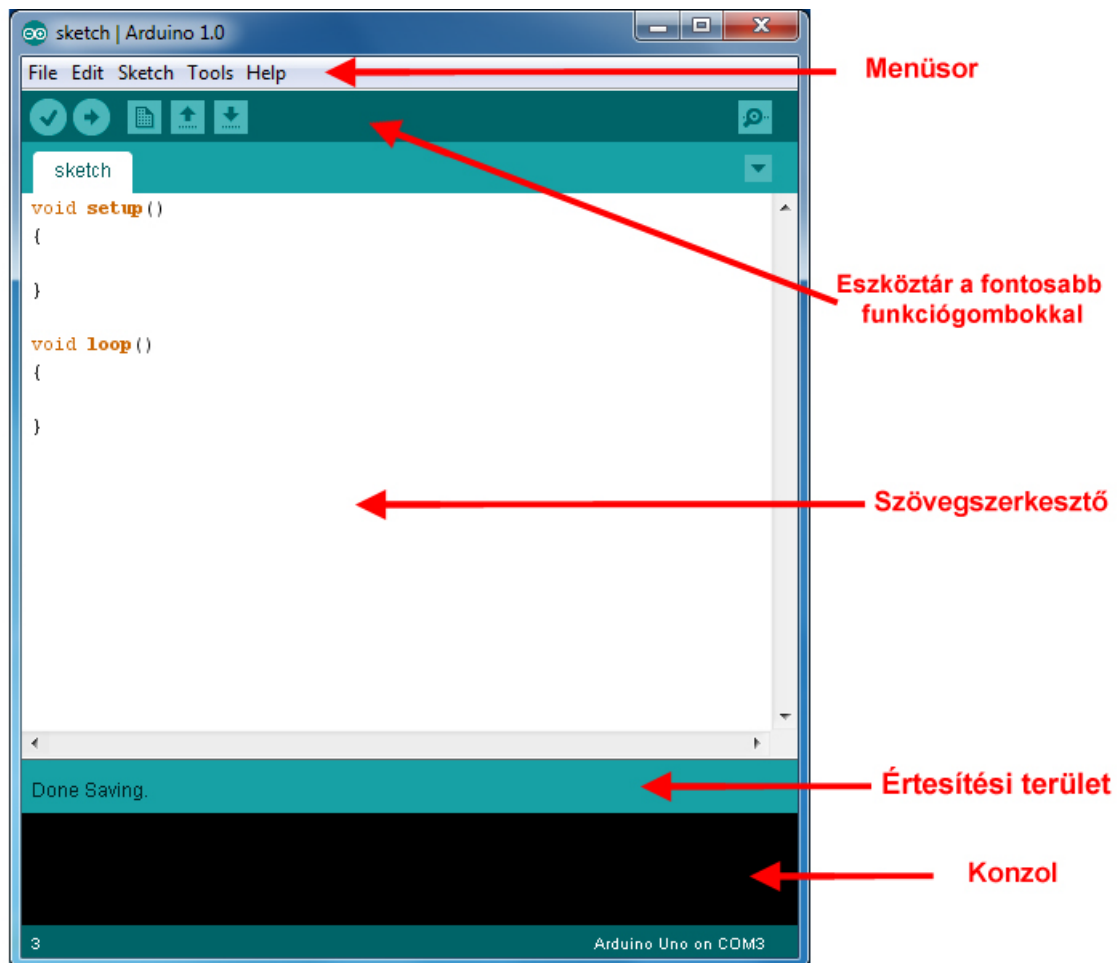
A következő főbb funkciókkal rendelkezik:

- Szintaxis kiemelés (a parancsszavak színezése)
- Kód indentálás (a kódsorok mozgatása az átláthatóság érdekében)

¹ külső cég

² Integrált fejlesztői környezet

- Keresés és csere
- Kommentezés (megjegyzések, magyarázó szövegek írásának lehetősége)
- Soros port monitor (figyelhető rajta a soros porton¹ bejövő és kimenő kommunikáció)



3.13. ábra. Arduino IDE

Az Arduinora írt programokat „sketch”-nek, vázlatnak nevezik. Ezek szövegfájlok, melyek tartalmát az Arduino IDE **szövegszerkesztő**jével módosíthatjuk. Kiterjesztésük az 1.0-s IDE verzió megjelenése óta .ino, előtte .pde volt. Az új verzióval megnyithatóak a régebbi kiterjesztésű sketchek is, de megnyitás után el kell őket menteni az új formátumban. Az **értesítési területen** kapunk visszajelzést a mentésről, a feltöltésről illetve ezek esetleges hibáiról. A **konzolban** az IDE és a fordítóprogram hibaüzeneteit és egyéb üzeneteit láthatjuk részletesen. A bal alsó sarokban megjelenő szám a sort jelöli, ahol a sketchben állunk, míg a jobb alsó sarokban láthatjuk, hogy milyen Arduino

¹jelenleg az USB porton

modell van és melyik porton csatlakoztatva a számítógéphez. Az **eszköztár gombjaival** tudunk programokat létrehozni, menteni, megnyitni, lefordítani és feltölteni az Arduinora, valamint itt tudjuk megnyitni a **soros port monitort**, melyben a soros portra érkező információkat olvashatjuk.

3.3.5. „Üzembe helyezés”, az IDE telepítése, kezdő lépések

Először is szükség van egy Arduino Uno-ra és egy USB A-B (nyomtató) kábelre. Ha ezeket beszereztük, töltsük le az operációs rendszerünknek megfelelő fejlesztői környezet legújabb verzióját a hivatalos honlapról:

<http://arduino.cc/en/Main/Software>

Windows operációs rendszeren egy tömörített, .zip állományt kapunk, Macintosh OSX alatt egy .dmg-t, míg Linux alatt az adott disztribúció¹ csomagkezelője segítségével történik a telepítés. Linuxon történő telepítéshez leírás itt található:

<http://www.arduino.cc/playground/Learning/Linux>.

Az alábbiakban csak az előbbi két operációs rendszerre való telepítést mutatjuk be:

- Windows

Csomagoljuk ki a letöltött .zip állományt, majd csatlakoztassuk az Arduinot a számítógéphez az USB kábel segítségével. Ezután telepítenünk kell a hozzá tartozó drivert. Ezt meg tudjuk tenni a Vezérlőpult/Eszközkezelő megnyitásával. A Portok (COM és LPT) alatt megjelent „Arduino UNO (COMxx)”-on (xx helyén a megfelelő COM port száma látható) jobb egér klikk, majd az „Illesztőprogram frissítése” ponton bal egér klikk lenyomása után válasszuk az „Illesztőprogram keresése a számítógépen” opciót és tallózzuk be az előbb kicsomagolt állomány Drivers könyvtárában található ArduinoUNO.inf fájlt. Ezután a Windows befejezi a driver telepítését. A fejlesztői környezet maga nem igényel telepítést, így indítsuk el azt.

- Macintosh OSX

A letöltött .dmg állományt nyissuk meg, majd másoljuk az Arduino.app-ot az Applications könyvtárba. Lévéen, hogy Uno modellt használunk, itt már nincs szükség driver telepítésére. Csatlakoztassuk az Arduinot a számítógéphez az USB kábel segítségével és indítsuk el a fejlesztői környezetet. Itt szintén nincs szükség külön ennek a telepítésére.

A program elindítása után írhatjuk is a saját kódunkat, vagy választhatunk a meglévő példák közül. Ezek kiválóak tanulásra, tesztelésre. Használhatjuk például a File/Examples/Basics/Blink (a magyar változatban File/Minták/Basics/Blink) nevű sketch-

¹ olyan terjesztés, amely egy Linux alapot használó operációs rendszert és néhány hozzá való programot tartalmaz

et. Betöltése után a Tools/Board (Eszközök/Alappanel) menüpont alatt a nekünk megfelelő Arduino modellre (például Uno) kattintsunk. A Tools/Serial Port (Eszközök/Soros port) alatt tudjuk kiválasztani, hogy melyik porton csatlakozik a számítógépünkhöz. Az eszköztáron található Upload (Feltöltés) gombbal pedig töltsük fel a programot. Ha mindent jól csináltunk, akkor a 13-as digitális porthoz kapcsolt integrált LED¹ (Az Arduino lapkán lévő LED) villogni fog a lapkán, futni fog a feltöltött program.

A következő fejezetben egy nagyon hasonló kód lesz bemutatva, ahol elemezve is van. A különbség a Blink példa és a 3.1. kódrészlet között, hogy előbbinél a 13-as digitális porton villogtatjuk a LED-et.

3.3.6. Egy konkrét példa programozás oktatására

A következő példával egyszerre taníthatunk, készségeket fejleszthetünk és több tudományterülethez kapcsolódó ismeretek körét bővíthetjük: fizika, elektronika, programozás és alkalmazói ismeret. Egy LED-et illetve egy LED sort fogunk vezérelni Arduino segítségével. Megtervezzük és megépítjük az egyszerű áramkört (fizika + elektronika), felprogramozzuk az Arduinot (programozási ismeret) és mindezek közben különböző szoftverek használatával ismerkedünk meg (alkalmazói ismeret).

LED

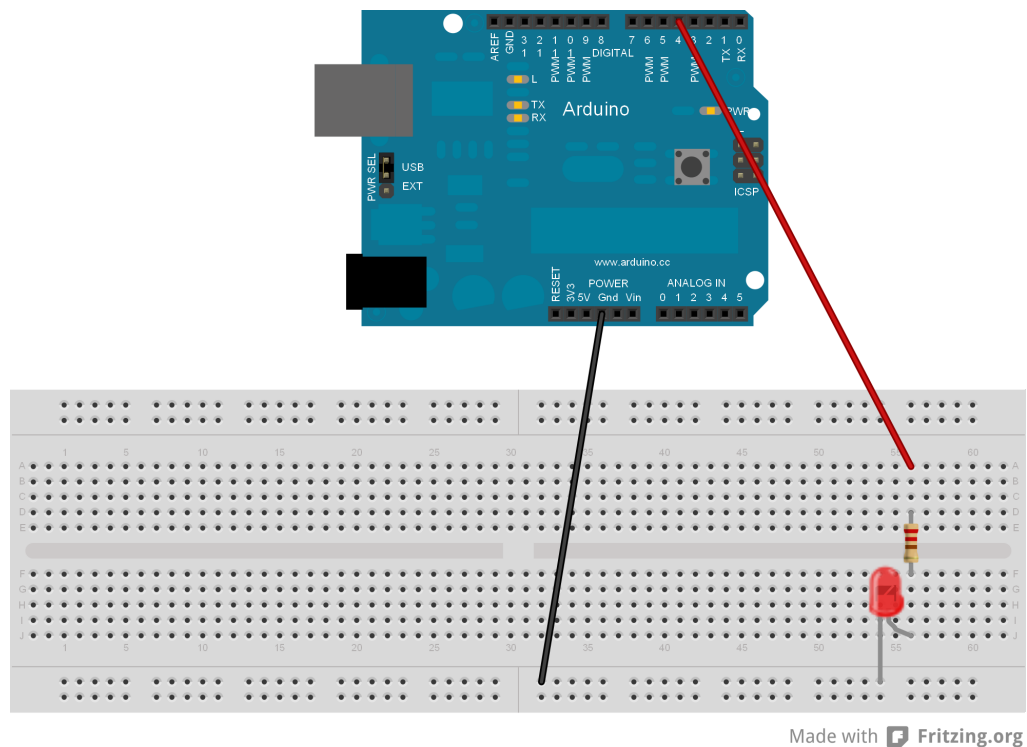
Mikor programozni tanul az ember, függetlenül attól, hogy milyen nyelvet használ, általában egy „Hello World!” kiírató alkalmazást készít el a rendszerrel való ismerkedés során. Ennek az Arduino-világbeli megfelelője egy LED villogtatása. (3.14. ábra)

Szükséges eszközök:

- Arduino Uno
- próbapanel²
- 1 darab LED
- 1 darab 220Ω-os ellenállás
- 2 darab vezeték
- USB A-B kábel

¹ fénykibocsátó dióda, olyan, mint egy kis lámpa

² olyan, mint egy műanyag lapka, amelyre forrasztás nélkül helyezhetők el a különféle elektronikai elemek



3.14. ábra. Kapcsolási rajz egy LED Arduinoval történő vezérléséhez

```

1 #define LED 4
2
3 void setup()
4 {
5     pinMode(LED, OUTPUT);
6 }
7
8 void loop()
9 {
10    digitalWrite(LED, HIGH);
11    delay(1000);
12    digitalWrite(LED, LOW);
13    delay(1000);
14 }

```

3.1. Listing. LED kapcsolgatás

A 3.1. program első sorában a 4-es számkonstanshoz egy szimbolikus nevet¹ rendelünk (a LED nevet), ezáltal növeljük a programkód rugalmasságát és olvashatóságát. A `setup()` metódusban a `pinMode()`-dal állítjuk kimenetre (OUTPUT) a 4-es digitális portot (a névvel hivatkozunk a 4-es számra), ugyanis ezen keresztül fogjuk a LED-et vezérelni. A `loop()`-ban pedig nem történik más, mint hogy fel-le kapcsolgatjuk a LED-et 1 másodperces időközönként, ezt a `delay(1000)` metódussal tudjuk elvégezni. Paramétere (az 1000-es érték) utal az 1 másodperces késleltetésre (`delay`).

Ha ezt az előző programot látványosabbá szeretnénk tenni, hogy például ne csak egy LED-et, hanem egy egész LED sort vezéreljünk amolyan Knight Rider Kitt-féle oda-vissza pásztázó módon, akkor ennek segítségével akár a for ciklust² és a tömbök³ kezelését is megtaníthatjuk.

LED sor

Ebben a példában nem egy LED-et, hanem egy 6 LED-ből álló LED sort fogunk vezérelni. (3.15. ábra) A LED-ek balról jobbra majd jobbról balra fognak egymás után sorra kigyulladni, majd kialudni. Szükséges eszközök:

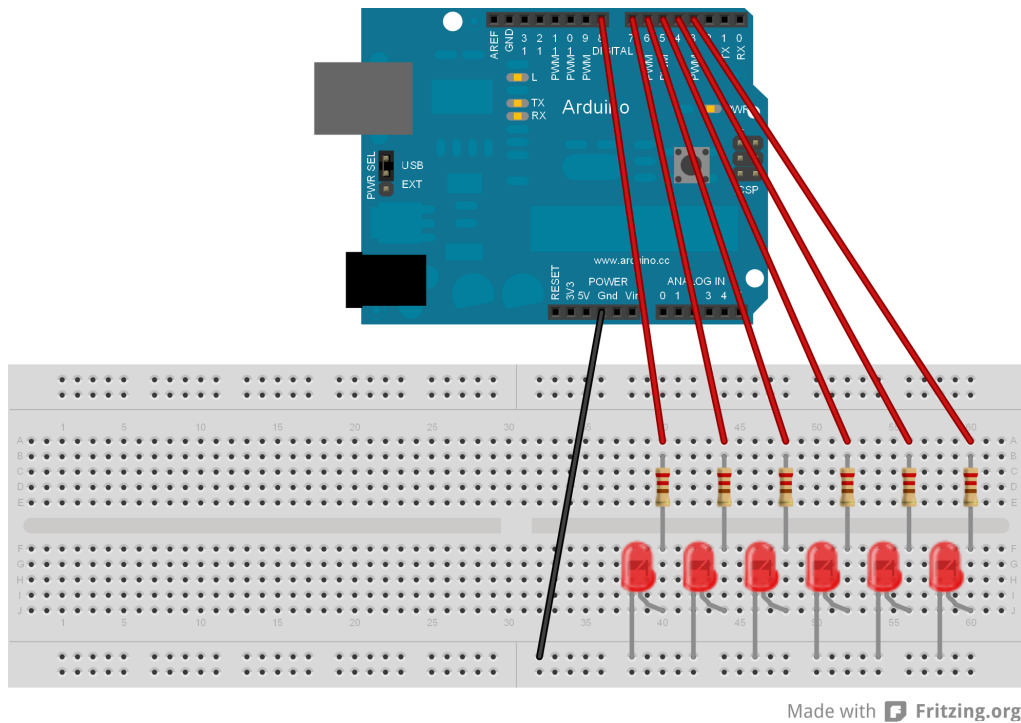
- Arduino Uno
- próbapanel
- 6 darab LED
- 6 darab 220Ω-os ellenállás
- 7 darab vezeték
- USB A-B kábel

```
1 #define DELAY 100
2 int LED[] = {3, 4, 5, 6, 7, 8};
3
4 void setup()
5 {
6     for(int i = 0; i < 6; i++)
7     {
8         pinMode(LED[i], OUTPUT);
9     }
```

¹ fontos a hatékony/megfelelő programozási stílus kialakítása szempontjából

² programvezérlési szerkezet, olyan, mint a LOGO-ban az ismétel parancs

³ adatszerkezet a programozásban, egy adatsorozat tárolható el benne



3.15. ábra. Kapcsolási rajz LED sor vezérléséhez
Arduinoval

```

10 }
11 void loop()
12 {
13     for(int i = 0; i < 6; i++)
14     {
15         digitalWrite(LED[i], HIGH);
16         delay(DELAY);
17         digitalWrite(LED[i], LOW);
18         delay(DELAY);
19     }
20     for(int i = 5; i ≥ 0; i--)
21     {
22         digitalWrite(LED[i], HIGH);
23         delay(DELAY);
24         digitalWrite(LED[i], LOW);
25         delay(DELAY);
26     }
27 }

```

3.2. Listing. LED sor kapcsolgatás

A 3.2. program első sorában a 100 számkonstanshoz egy szimbolikus nevet (DELAY) rendelünk, rajta keresztül hivatkozunk a 100-as értékre. A harmadik sorban egy egész számokat tároló tömböt deklarálunk¹ LED néven, ez azoknak a digitális portoknak az indexét tárolja, melyekre a LED-eket kötöttük. A `setup()` metódusban az előbbi tömb elmein egy ciklussal végiglépkezdve kimenetre állítjuk a portokat. A `loop()` metódusban egy-egy for ciklussal egyik irányba, majd a másik irányba lépkezdve a LED-eken fel-le kapcsolgatjuk azokat 0,1 másodpercenként.

3.3.7. Az Arduino előnyei, hátrányai, lehetséges helye a közoktatásban

A LEGO robot készletével ellentétben anyagi oka nincs annak, hogy a magyar közoktatás ne használja fel az Arduinot, mint oktatási eszközt, ugyanis a hardver olcsó és minden nyílt forrású, akár egyénileg is megépíthető. Nyilván egy komplett robot létrehozásához jóval több kreativitás és tudás, valamint egyedi alapanyagok szükségesek. Valamivel komolyabb előismereteket és absztrakciós szintet is igényel. A középiskolás korosztály az, amelyiknek ideális, ugyanakkor természetesen a tanár esetében szükséges ismereteinek bővítése ezen a területen.

¹ A deklarációból derül ki, hogy mennyi helyet kell lefoglalni az értékeknek a memóriában

4. fejezet

RoboLOGO projekt

4.1. A teknősök történelme

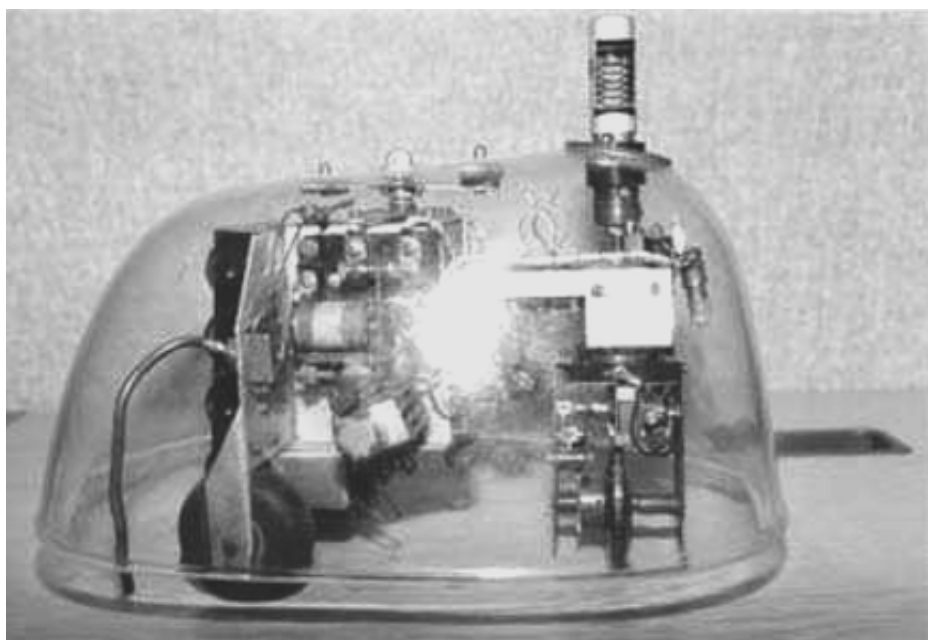
Az 1960-as években személyi számítógépek még nem léteztek. A Digital Equipment Corporation elkészítette a PDP 1-et (Programmed Data Processors - programozott adatfeldolgozók), melynek még akkoriban nem volt nagy képernyője. A tanulók rá voltak kényszerülve, hogy távíróval kommunikáljanak a PDP-vel. Ekkor jött a képbe egy LOGO nyelven működő robot (4.1. ábra) megépítése. Ez a robot Seymour Papert találománya, melyet végül Tom Callahan valósított meg az MIT Laborban. Jellemzője volt a lassúság és a robusztus felépítés.

Seymour teknőcnek nevezte el Grey Walter munkái nyomán, hiszen ő hozta létre a világ első autonóm (önálló, emberi irányítást nélkülöző) robotjait (4.2. ábra), melyeket szintén teknőcöknek nevezett el.

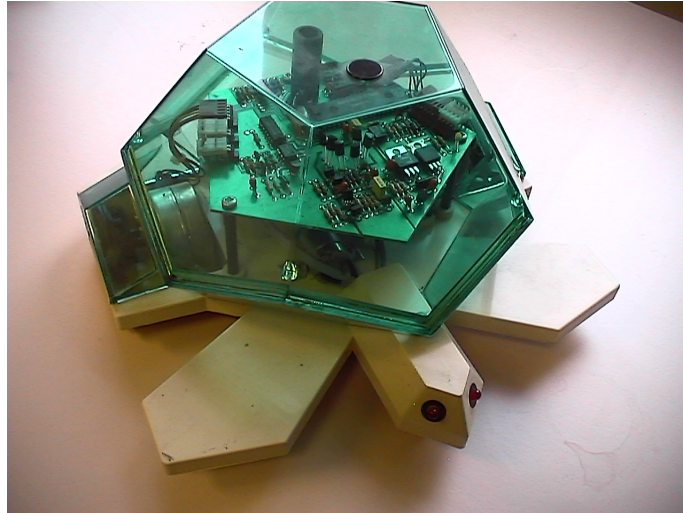
1985-ben megalakult a Valiant Technology nevű cég [24], melynek első és legfontosabb terméke az ún. Valiant Turtle (4.3. ábra) volt. Ez érdemelte ki az eddigi legsikeresebb teknőc robot címet. 1983-tól 2011-ig lehetett kereskedelmi forgalomban megtalálni. Jellemzője, hogy a mai LEGO Mindstorms NXT-hez hasonlóan nagyon drága volt (körülbelül 100.000 Ft) és viszonylag lassú mozgású is. A magyar oktatásban nem jelent meg. A teknőcökről bővebben [25].



4.1. ábra. Seymour Papert és Tom Callahan LOGO robotja



4.2. ábra. Grey Walter-féle autonóm teknőc



4.3. ábra. Valiant Turtle

4.2. Egy oktatási célú mobil robot koncepció bemutatása

Mind a jobban, mind a kevésbé érdeklődő tanulók problémájára megoldást jelenthet egy olyan eszköz, illetve annak működtetése, ami a „jobbakat” további munkára ösztönzi, a tantárgy iránt kevésbé fogékony, motiválatlan tanulóknál pedig felkeltheti az érdeklődést. Ahhoz, hogy a korábban említett - már a magyar informatikaoktatásban is használt, de teljes körben elterjedni nem képes - eszközök „hátrányaitól” ne szenvedjünk, robotunk bármelyik iskola és bárki számára megfizethető kell, hogy legyen. Használata nem igényelhet a tanár részéről plusz képesítést, képzettséget, a gyerekek részéről pedig plusz tanulást.

A fentebb megfogalmazottak alapján született a következő ötlet: okostelefonnal (is) vezérelhető, programozható, LOGO nyelvet „beszélő” robot. Az általunk tervezett, Arduino vezérlésű rajzoló robot - a gyerekek LOGO nyelven megírt programjai által - a fizikai valóságban teszi ugyanazt, mint amit a teknőc virtuálisan a számítógép képernyőjén. A robotot a diákok a tanórán elsajátított algoritmizálási és LOGO nyelvi ismereteik birtokában, egy könnyen kezelhető, okostelefonokra fejlesztett és optimalizált érintőképernyős felületen tudják vezérelni. A robot működtetése függetlenné válhat az iskolától, a tanítási órától, az asztali számítógépektől és a diákok már-már játszva tanulnak, észre sem veszik, hogy az informatika tantárgy tananyagát alkalmazzák, gyakorolják. A korábban felsorolt próbálkozások hátrányait, hiányosságait nem hordozná magával ez az eszköz: a robot alacsony költséggel előállítható, okostelefonnal rendelkeznek a tanárok is és a LOGO ismereteknek pedig már mind a birtokában vannak.

Nincs szükség új módszertan kidolgozására, a LOGO módszertana egy az egyben alkalmazható.

A következőkben a projektet alkotó, a csapatok által megvalósított alprojekteket mutatom be. Ahogy korábban említettem a laborban 2-3 fős csapatok dolgoztak különböző projekteken. Mivel ez egy nagyobb lélegzetvételű feladat, így minden csapat kivette a részét a munkából. A robot megtervezése és megépítése, az Androidos alkalmazás fejlesztése illetve egy „kis” projekt koordinálás képezte a munka ránk eső részét. Egy csapat feladata volt, hogy az Android alkalmazás mellé fejlesszenek egy Imagine LOGO-szerű, de modernebb asztali alkalmazást és fejlesztői környezetet. Ehhez szorosan kapcsolódva egy másik csapat fejlesztette a LOGO-C#¹ fordítót és a teknőcgrafika szimulátort. Egy harmadik csapat pedig az Arduino programját illetve azt a fordító programot írta, mely a LOGO nyelvet lefordítja az Arduino „nyelvére”, a C nyelvre. A teljes RoboLOGO csapat a 4.4. ábrán látható.



4.4. ábra. RoboLOGO csapat

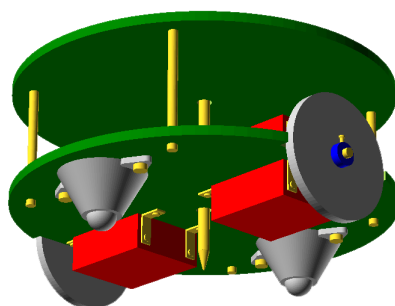
4.3. A robot felépítése

A robot a 4.7, a 4.8 és a 4.9. ábrán látható. Működését egy **Arduino Uno** vezérli. Azért választottuk az Uno modellt, mert ez a legolcsóbb és bőven megfelelő képessé-

¹ A C# is egy programozási nyelv



4.5. ábra. A Robotika labor



4.6. ábra. A robot modellje

gekkal rendelkezik a cél megvalósításához. Mozcását és pozíciónálását **motorvezérlő áramkörrel**, **2 léptető motorral**, **2 bolygókerékkel** és **2 alumínium esztergált kerékkel** valósítjuk meg. A „teknős” toll funkciójának megvalósítására két módszert is teszteltünk:

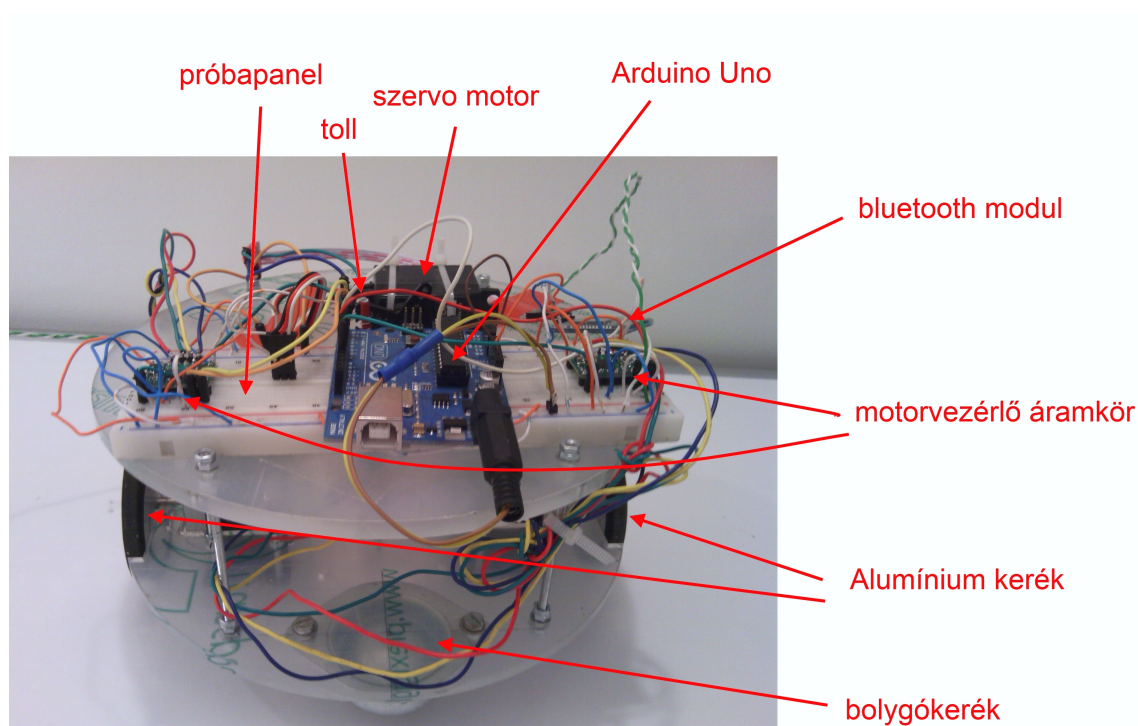
- a tollat a roboton egy elektromágnes segítségével tudjuk felemelni vagy letenni, attól függően, hogy szeretnénk-e hogy rajzoljon, miközben éppen halad vagy sem
- egy **szervo motor** segítségével mozgatjuk fel-le a **tollat**

Végülis a legjobb ötletnek az utóbbi bizonyult. Áramellátásáról lítium akkumulátorok és egy töltő áramkör fog gondoskodni, így az akkumulátorok helyes töltésére figyelni kell. A robotnak legalább egy tanítási órát, de inkább többet ki kell bírnia folyamatos használat mellett. A prototípus jelenleg még tápfeszültségről üzemel. A robot az utasításokat vezeték nélküli kapcsolaton keresztül, egy **bluetooth adapter** segítségével

kapja az okostelefonon futó alkalmazástól.

Az Arduinon futó programot mi írjuk meg a robot számára, azzal nincs különösebb teendő a későbbiekben, csupán akkor, ha újabb (javított vagy bővített funkcionalitású) verziót adunk ki belőle. Ekkor frissíteni kell, tehát fel kell tölteni az Arduinora, ami pár egyszerű lépésen át kivitelezhető bárki számára. A kapott utasításokat először interpretálja, majd végrehajtja az Arduinon futó program.

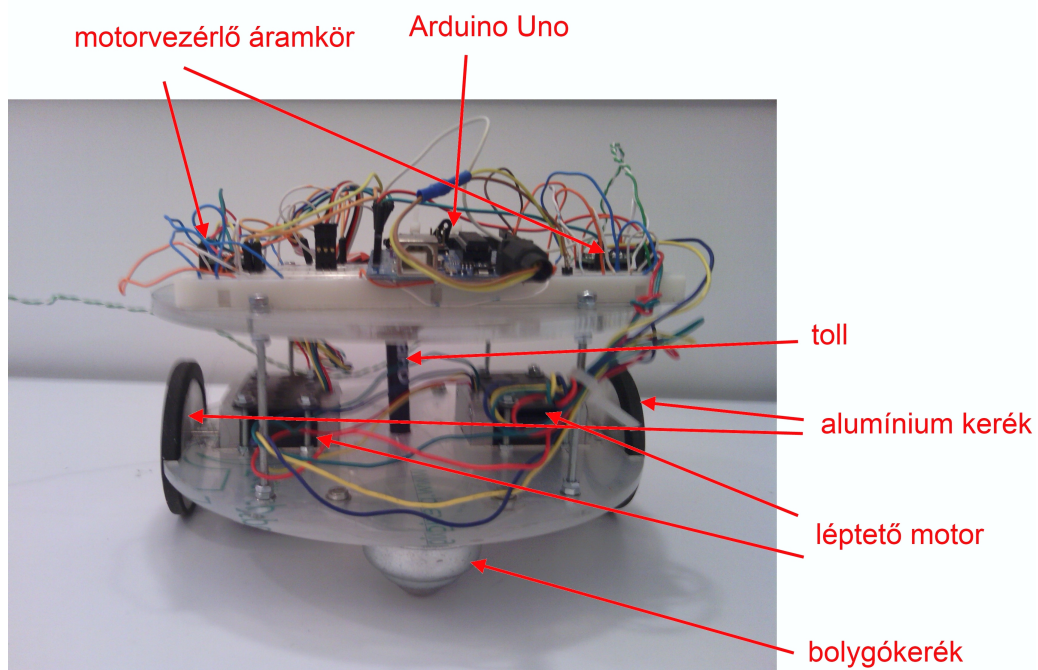
A robotot bárki meg tudja építeni, legyen az egy érdeklődő diák vagy egy szigorú költségvetésből gazdálkodó iskola tanára, de akár tőlünk is megrendelheti a későbbiek folyamán, amennyiben termék lesz belőle. A szükséges alkotórészek mindenki számára elérhetőek és megfizethetőek szerény anyagi ráfordítással. A jelenlegi prototípusunk megépíthető 30.000 Ft-ból, de a végtermék ennek töredéke lesz, ugyanis itt nem célhardvereszközöket használtunk (Pololu A4983 motorvezérlő, Arduino Uno, 1 példány-számban méretre vágott plexi robottest), ezeken még sokat lehet spórolni.



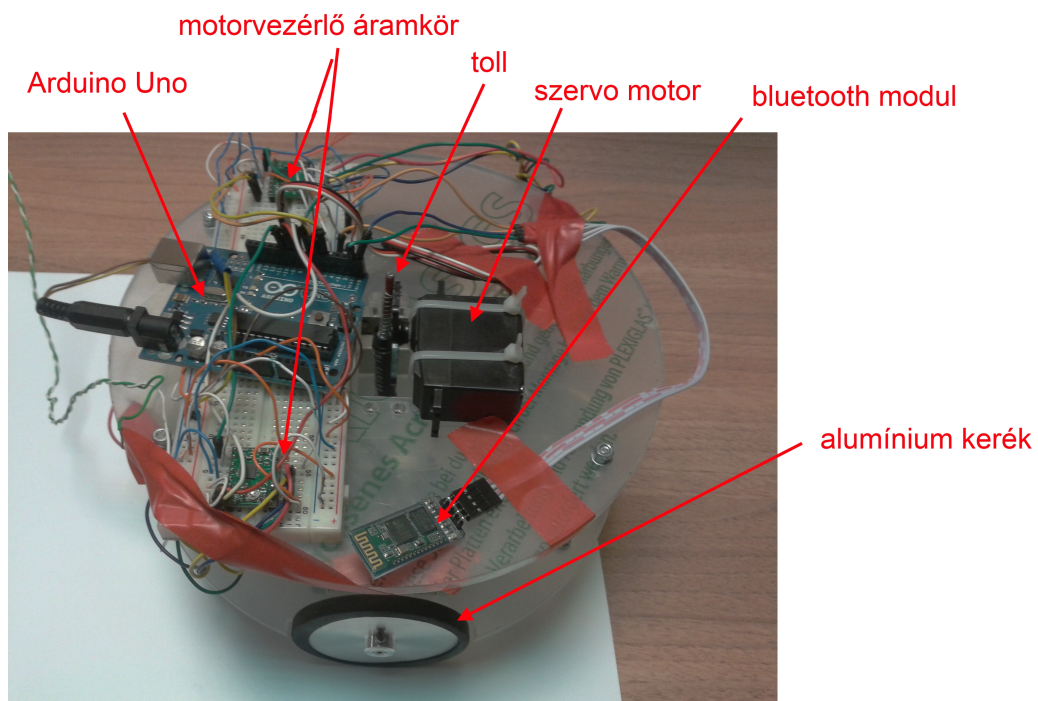
4.7. ábra. RoboLOGO robot prototípus

4.4. Mobil fejlesztői környezet - Android alkalmazás

Az Android kliens Java nyelven íródott. A nyelvről érdemes tudni, hogy legjobban éppen a hálózatalapú programokban működik. Az Android mobil operációs rendszerre



4.8. ábra. RoboLOGO robot prototípus



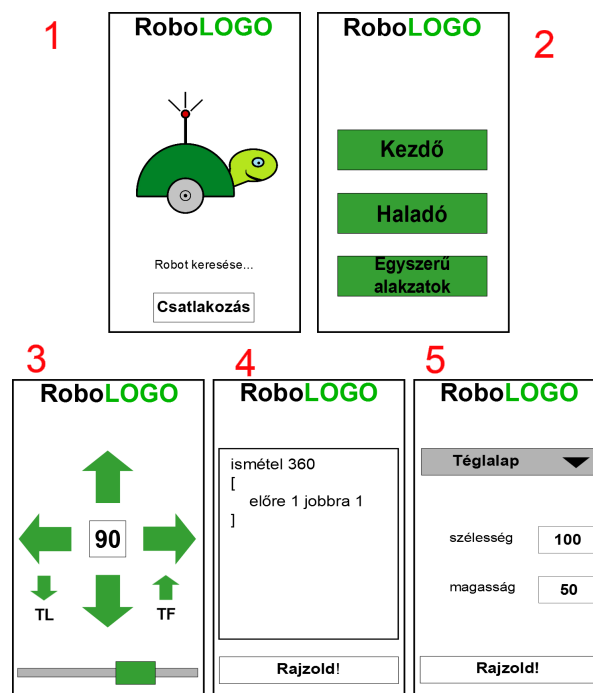
4.9. ábra. RoboLOGO robot prototípus

azért esett a választásunk, mert ez a legszélesebb körben elterjedt a célközönségünkben, azaz magyar diákok és tanárok között. Továbbá nyílt forráskódú és tetszés szerint fejleszthetünk rá alkalmazásokat. Széleskörű elterjedéséhez jelentősen hozzájárult a rend-

szer stabilitása és bővíthetősége, valamint a nagyfokú támogatás [26] és a számtalan ingyenesen letölthető program.

4.4.1. Felépítése

A RoboLOGO Android alkalmazás (4.10. ábra) elindítása után Bluetooth vezeték nélküli technológiával **csatlakozik az okostelefon a robothoz**, ezt láthatjuk az első képernyőképen (1). Sikeres csatlakozás után egy **funkció választó menü** (2) jelenik meg, mely 3 választási lehetőséget kínál. „**Kezdő**” menüpont (3) alatt a robotot előre, hátra tudjuk mozgatni illetve jobbra vagy balra forgatni a csúszkával beállított paraméterértéknyivel, vagy manuálisan értéket adva a középső szövegdobozba. A „**Haladó**”-ra kattintva (4) egy szokványos kódszerkesztőt kapunk, így összetett feladatokat is adhatunk a robotnak. Végül, de nem utolsó sorban paraméterezhető, előre definiált alakzatokat is rajzoltathatunk, ezt a funkciót az „**Egyszerű alakzatok**” (5) alatt érjük el.



4.10. ábra. RoboLOGO Android alkalmazás

4.4.2. Terméktámogatás

A mi kliensünk természetesen ingyenesen hozzáférhető lesz a Google Play digitális disztribúciós platformon, ami gyakorlatilag egy webbolt alkalmazásokkal az Android felhasználók számára. A letöltött alkalmazást a rendszer automatikusan fel is fogja telepíteni a telefonunkra, ezzel sem kell bajlódnunk. Mivel a program kis méretű, még

a viszonylag lassú Wifi hálózattal rendelkező kapcsolódási pontoknál is rövid idő alatt képes letöltődni.

Az alkalmazáshoz folyamatos terméktámogatást biztosítunk. Aktív netkapcsolat birtokában az Android automatikusan figyelmeztetni fogja a felhasználókat az újabb verzió meglétéről, így nem nekik kell rendszeresen figyelniük a Google Play oldalát. A program a jövőben bővíthet a felhasználók igényei szerint, ami nem megy majd az átláthatóság rovására. Az elsődleges cél mindig az egyértelműség, az egyszerű kezelhetőség lesz.

4.5. Fejlesztői környezet - asztali alkalmazás

.NET 4.5¹ alapokra építkező, C# nyelven írt, nyílt forráskódú, tetszőlegesen bővíthető, ingyenes WPF² alkalmazás, mely az ImagineLOGO koros, elavult kinézetű, zárt forráskódú, régóta nem fejlesztett asztali alkalmazást hivatott leváltani, emiatt sokkal előnyösebb tulajdonságokat takar. Amennyire csak lehet kompatibilis vele. A fejlesztői környezet magyar és angol nyelvet támogat, de egyszerűen bővíthető további nyelvekkel. A kezelőfelület, a szintaxis kiemelés és parancs értelmezés, valamint a hibaüzenetek nyelve is megválasztható.

Főbb jellemzői:

- Új, modern*
- ImagineLOGO kompatibilitás (amennyire csak lehetséges)
- Nyílt forráskód, szabadon bővíthető, módosítható*
- Többs nyelvű
- SQLite³ adatbázis háttér
- Saját animált technócgrafika*
- Bluetooth kapcsolat a robottal*
- Szintaxis kiemelés*
- Eljárások írása
- Állítható rajzlapméret*

A csillaggal megjelölt jellemzők újítások az Imagine LOGO-hoz képest.

¹ programfejlesztést elősegítő fejlesztői platform

² .NET alapú grafikus programok fejlesztését segítő technológia

³ ingyenes adatbázis-kezelő rendszer

4.5.1. Felépítése

Az asztali alkalmazás a következő elemekből épül fel:

- Fájl menü:

Projekt létrehozása, illetve előző projekt megnyitása, ahol előző, saját .rbsln, másnéven RoboLOGO Solution típusú projektfájlt tölthetünk be vagy újat hozhatunk létre.

Megadhatjuk a készítő(k) nevét, a szintaxis nyelvét is¹

A vászon szélességét, magasságát és a mentés helyét.

A program felületének a nyelvezetét is itt tudjuk átállítani.

- Szerkesztés menü:

Az előző, következő utasításra ugorhatunk, más programoknál megszokott módon. Másolhatjuk, kivághatjuk, beilleszthetjük a begépelte parancsokat, kódokat

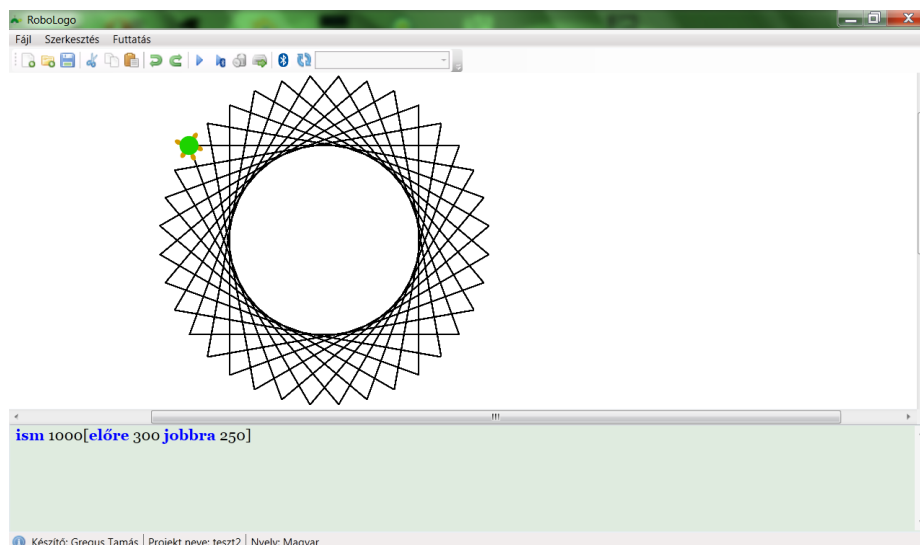
- Futtatás menü:

Lefuttathatjuk a parancsokat

Törölhetjük a rajzvászont

Új eljárásokat tárolhatunk le

A főbb menüpontok alatt az eszköztáron is megtalálhatjuk az opciókat, valamint itt tudunk a robot bluetooth moduljára csatlakozni. Megválaszthatjuk, hogy az utasításokat a szimulátorban (a programban), vagy a robottal hajtjuk végre.



4.11. ábra. RoboLOGO fejlesztői környezet

¹ hogy milyen nyelven írjuk a LOGO kódot

5. fejezet

Iskolai tapasztalatok a RoboLOGO-val

A robotot és a hozzá tartozó tanulási környezeteket volt alkalmunk bemutatni az Eszterházy Károly Főiskola Gyakorlóiskolájában is, ahol a diákok és a tanárok egyaránt érdeklődéssel fogadták ezt a koncepciót. Itt informatika órák keretében ismerkedhettek meg a tanulók a robottal.

Az órák első perceiben bemutattuk az Imagine LOGO-t is¹, hogy azok, akik ezt a tanulási környezetet nem ismerték korábban, képet kaphassanak a működéséről, felépítéséről, tulajdonságairól, a benne rejlő lehetőségekről. Ezután a tanulókat megismertettük az RoboLOGO asztali alkalmazásával.

Láthatták, ahogy a virtuális teknőc a szimulációs környezetben rajzol egy alakzatot. Ezt követően megismerhették, hogy okostelefonnal hogyan lehet irányítani a robotot. A diákok végig érdeklődéssel hallgatták ezeket az ismertetőket és alig várták, hogy működés közben is láthassák a robotot. Először mi irányítottuk, egyszerű utasításokat küldve a telefonról, majd az asztali kliensről. Nagyon érdekesnek találták, hogy akár a telefonjukról is vezérelhetik. Hamar akadtak önként jelentkezők, akik a mobil applikáció használatát könnyedén elsajátították, élvezettel utasították a robot-teknőcöt. Az órák végén egy rövid, lényegre törő online kérdőívet [27] töltöttünk ki a tanulókkal, akik készségesen válaszoltak kérdéseinkre.

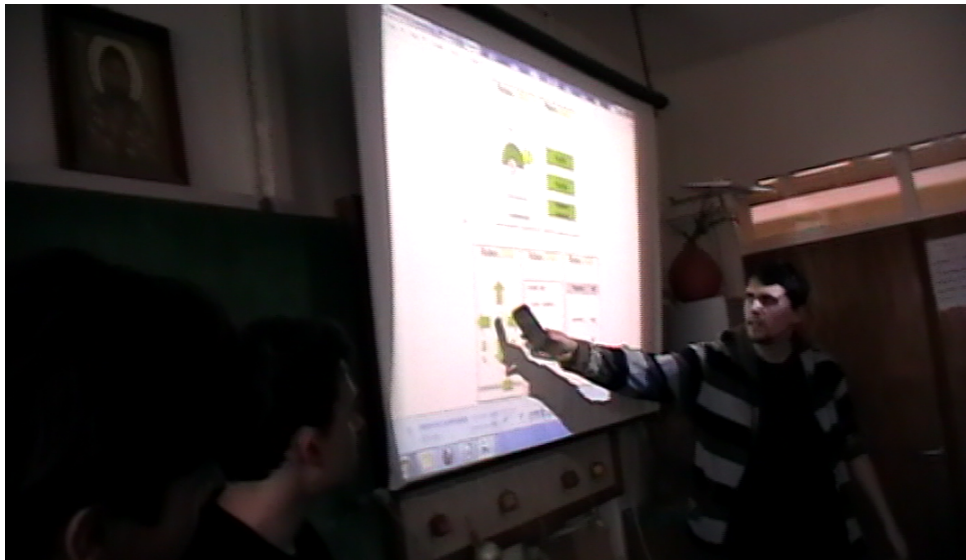
A kérdőívre adott válaszokból kapott eredményeket a következő dokumentum tartalmazza: http://robotika.ektf.hu/robologo_kerdoiv_eredmenyek.ods

¹erre azért volt szükség, mert a szűkös órakeret miatt kevésbé ismert

Pár diák véleménye a RoboLOGO-ról:

- „Egyszerű és jó!”
- „Jó volt, de nem teljesen nyerte el a tetszésemet.”
- „Szerintem praktikus és könnyen kezelhető. Jól néz ki, és szerintem bárki képes lenne használni.”
- „Nagyon tetszett. Nagyon könnyű volt. Szeretnék ilyen otthonra.”
- „Könnyen kezelhető, aranyos robot.”
- „Elég érdekes, szerintem nincs rá szükség. Talán kisebb gyerekeknek érdemes megvenni, mert nekik nagyon érdekes lehet és egyszerű kezelni.”

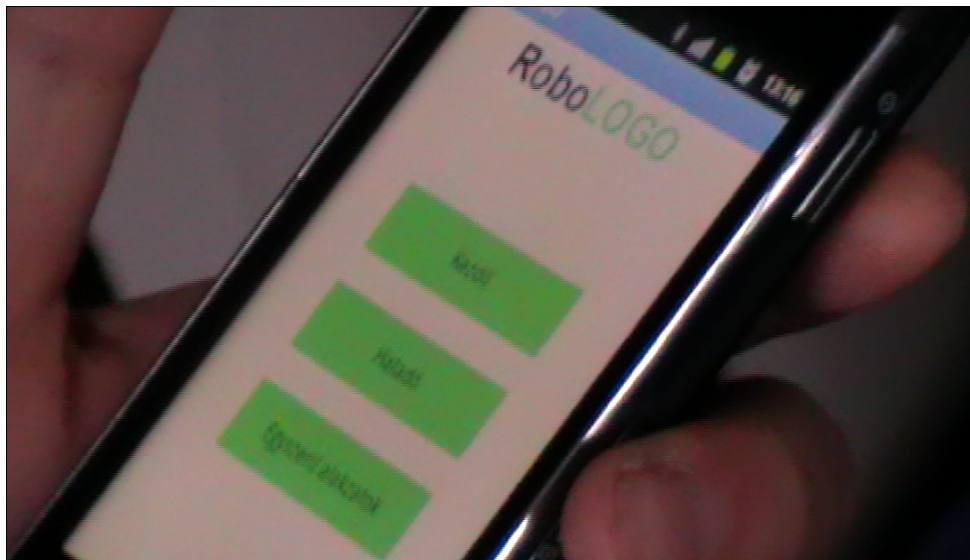
Az [5.1](#), [5.2](#), [5.3](#), [5.4](#), [5.5](#), [5.6](#). képek néhány tanítási órán készültek.



5.1. ábra. Épp a mobil alkalmazás bemutatása
zajlik



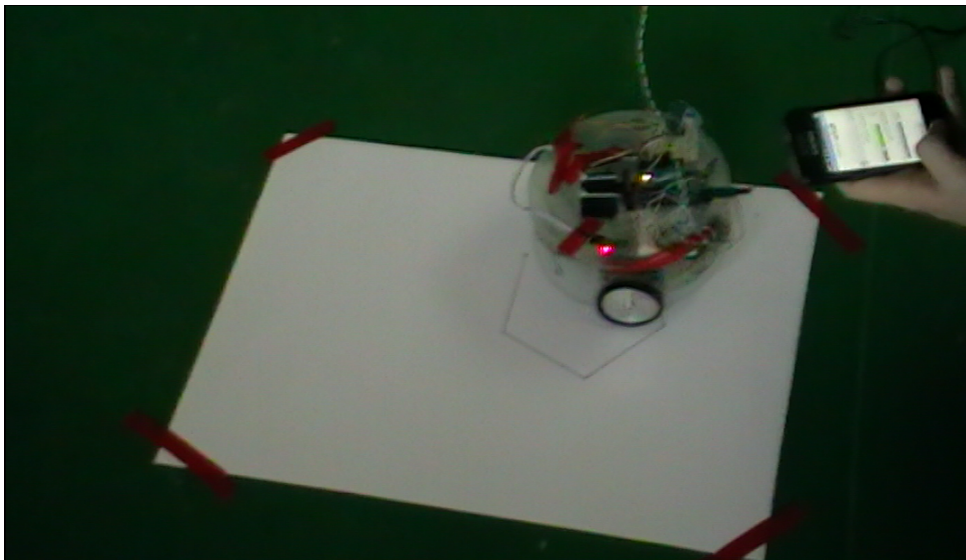
5.2. ábra. A diákok kipróbálják a mobil alkalmazást



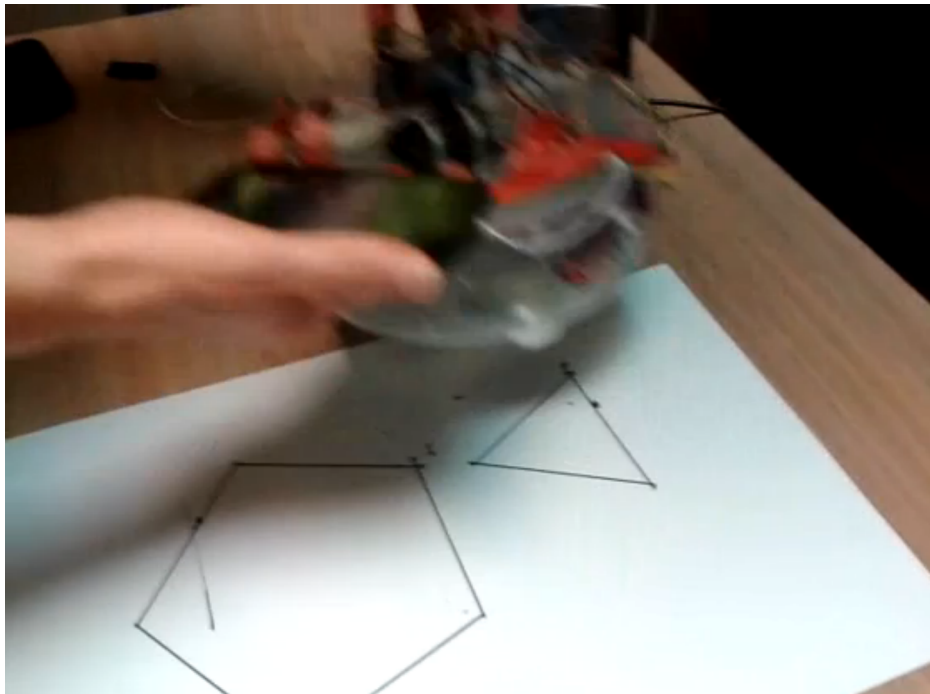
5.3. ábra. RoboLOGO mobil alkalmazás



5.4. ábra. A diákok érdeklődve figyelik a robotot



5.5. ábra. A robot rajzolás közben



5.6. ábra. Rajzolás után

6. fejezet

Jövőbeli tervek

Természetesen a munkánk itt nem ér véget. A projekt (alkalmazások és a robot) eddigi funkcióit folyamatosan bővítjük, hibáikat, hiányosságait javítjuk, pótoljuk.

Az Android alkalmazás mellé mindenképp készülni fog Windows Phone alkalmazás is. Az asztali alkalmazásunkat platformfüggetlenné fogjuk tenni, elérhető lesz Linuxra és OS X-re kiadott változata is. Ennek megvalósításához az asztali klienst Java programozási nyelven fogjuk újraírni. A fejlesztésekkel párhuzamosan újabb és újabb iskolákat fogunk meglátogatni a RoboLOGO-val és a tanulók visszajelzései alapján dolgozunk tovább a projekt színvonalának emelésén.

A robotot szenzorokkal felvértezve pontosabb eredményeket kaphatunk, illetve összetettebb feladatok elvégzésére is alkalmassá tehetjük. Például ultrahang és szín szenzorral figyelhetjük, hogy a robot elhagyta-e a rajzterületet.

Irodalomjegyzék

- [1] OFI dokumentumtár weboldal:
dokumentumtar.ofi.hu/index_NAT_informatika.html
- [2] Logo Foundation weboldal:
el.media.mit.edu/logo-foundation/logo/programming.html
- [3] Seymour Papert: Észrengés (A gyermeki gondolkodás titkos útjai), SZÁMALK, Bp., 1988
- [4] NJSzT Logo Országos Számítástechnikai Tanulmányi Verseny weboldal:
logo.inf.elte.hu
- [5] Sulinet SDT adatbázis weboldal:
<http://regi.sdt.sulinet.hu/Player/default.aspx?g=ffa68cfb-dea9-44bf-b064-47dd85c24c1c&cid=dd7876b7-8819-4873-82a3-56341e410c51>
tananyagok/műveltségi területek/informatika/elektronikus írásbeliség/Imagine
Súgó magyar nyelvű parancsokkal
- [6] Dancsó Tünde: Comenius Logo - Játék és Animáció, Kossuth Kiadó, 2000.
- [7] Addo Stuur - Turcsányiné Szabó M.: Comenius Logo - Játék és Programozás, Kossuth Kiadó, 1998.
- [8] Szentpéteri Király Tünde: Comenius Logo - Teknőc grafika, Kossuth Kiadó, 1999.
- [9] Turcsányiné Szabó M. - Zsakó L.: Comenius Logo Gyakorlatok, Kossuth Kiadó, 1997.
- [10] Abonyi-Tóth Andor, Holler János, Rozgonyi-Borus Ferenc: Képzeld el! Imagine - algoritmusok, játékok, ABAX Kiadó, 2008
- [11] Abonyi-Tóth Andor, Holler János, Rozgonyi-Borus Ferenc: Képzeld el! Imagine - teknőcgrafika, multimédia és játékok, ABAX Kiadó, 2007

- [12] LibreLogo kézikönyv:
[http://www.numbertext.org/logo/librelogo_kezikonyv_nyomtatasra.pdf](http://www.numbertext.org/logo/librelogo_kezikonyv nyomtatasra.pdf)
<http://www.numbertext.org/logo/logofuzet.pdf>
- [13] LEGO Mindstorms NXT weboldal:
mindstorms.lego.com/en-us/Default.aspx
- [14] Kiss Róbert, Badó Zsolt: Egyszerű robotika, Kecskemét, 2010.
- [15] Szabó Richárd: A LEGO Mindstorms NXT programozása weboldal:
<http://www.jataka.hu/rics/lego/index.html>
- [16] Kőrösi Gábor: Az NXC nyelv - Lego NXT robot programozása NXC nyelv segítségével, Zenta, Bolyai Tehetséggondozó Gimnázium, 2010.
- [17] Bricx Command Center weboldal: <http://bricxcc.sourceforge.net/>
- [18] RobotC weboldal: <http://www.robotc.net/>
- [19] Ságvári Endre Gyakorló Gimnázium LEGO Szumó weboldal:
<http://sagv.gyakg.u-szeged.hu/szumo/>
- [20] Bányai Júlia Gimnázium Robotika weboldal:
<http://www.banyai-kkt.sulinet.hu/robotika/html/versenyek.html>
- [21] Arduino weboldal: www.arduino.cc
- [22] Massimo Banzi: Getting Started with Arduino, O'Reilly Media / Make, 2008.
- [23] Brian W. Evans, Cseh Róbert: Arduino programozási kézikönyv, TavIR, Budapest, 2011.
- [24] Valiant weboldal:
<http://www.valiant-technology.com/uk/pages/whoarewe.php>
- [25] Régi robot teknőcök:
http://www.waitingforfriday.com/index.php/Valiant_Turtle
<http://roamerrobot.tumblr.com/post/23079345849/the-history-of-turtle-robots>
<http://cyberneticzoo.com/?p=1711>
<http://logothings.wikispaces.com>
- [26] Android fejlesztői weboldal: <http://developer.android.com/index.html>
- [27] RoboLOGO diák kérdőív:
<https://docs.google.com/spreadsheets/viewform?formkey=dGEtUjV5MXNrVjIwRW5ldmwzNGQ5dkE6MQ#gid=0>