EÖTVÖS LORÁND TUDOMÁNYEGYETEM
TERMÉSZETTUDOMÁNYI KAR

TANANYAGKÉSZÍTÉS A SCRATCH PROGRAMOZÁSI KÖRNYEZETHEZ

Témavezető:
Turcsányiné Szabó Márta
egyetemi docens
ELTE Informatikai Kar,
Média- és Oktatásinformatikai
Tanszék

Készítette:
Takács Valéria
Matematika–informatika-
tanári szak,
nappali tagozat

Budapest, 2009

Tartalomjegyzék

Bevezetés 4
Mi is az a Scratch? 5
A NAT és a Kerettanterv elemzése a tananyag szempontjából 12
NAT – kompetenciák 12
NAT – Informatika műveltségi terület 14
Kerettanterv 16
A Scratch az oktatásban 18
A tananyag bemutatása 20
A tananyag besorolása 20
A hipertext által segített oktatás 21
Cél: az algoritmikus gondolkodás fejlesztése játékos módon 21
Internetes tananyagok tervezése 22
Didaktikai tervezés 22
Modularitás 24
Rendszerszemlélet 25
A tervezés folyamata és nehézségei 25
A leckék felépítése 29
A tananyag tartalma 31
1. lecke – Ismerkedés a Scratch környezettel 31
2. lecke – Megmozdulnak a szereplők 31
3. lecke – Ismétlődések 33
4. lecke – Feltételek 34
5. lecke – Üzenetek 36
6. lecke – Rajzoljunk 37
7. lecke – Változók és véletlenszámok 38
8. lecke – Listák 39
Kiegészítő leckék 41
A portál 43
Módszertani útmutató 46
Előzetes ismeretek, célzott korosztály 46
A tananyag felhasználása az iskolai oktatásban 47
Az algoritmusok és adatok témakör tanítása a Scratch programozási környezet használatával 47

A Scratch programozási környezet bemutatása alternatívaként más programozási környezet mellett 52
A Scratch programozási környezet bemutatása szakköri foglalkozáson 56
Differenciálás 57
A tananyag felhasználása önálló tanulás során 59
Az általános iskola és az önálló tanulás 59
A tananyag és a tanulási stílusok 60
A tananyag tesztelése a gyakorlatban 64
Első kísérlet 65
Második kísérlet 70
Harmadik kísérlet 72
Fejlesztési lehetőségek 74
Irodalomjegyzék 76

Bevezetés

Szakdolgozatom célja a Scratch programozási környezethez általam készített tananyag bemutatása különböző szempontok alapján. A tananyag elektronikus formában érhető el a Scratch Magyarország portálon, amely a http://scratch.inf.elte.hu vagy a http://kihivas.inf.elte.hu/scratch címen érhető el.

A tananyag bevezetésként szolgál a Scratch környezet használatához és egyben a programozáshoz is, elsősorban 10-14 éves gyerekek számára. Az anyag nyolc fő leckéből áll, amelyek feldolgozzák az informatika vonatkozó területeinek az általános iskola felső tagozata számára előírt témáit. A Scratch környezet hazánkban még viszonylag ismeretlen, ezért ennek bemutatását is szükségesnek tartom. Röviden ismertetem születésének előzményeit, a készítése során alkalmazott elveket, célokat, és az elkészült környezet jellegzetességeit.

A szakdolgozatban a tananyagot különböző szempontok szerint vizsgálom, elemzem. Bemutatom, hogy a NAT által ismertetett kulcskompetenciák közül melyek megléte elengedhetetlen a tananyag sikeres elsajátításához. Ezen felül azt is, hogy a már meglévő kompetenciák fejlődését, illetve a még hiányzók kialakulását a tananyag mennyiben segíti elő. Fontos szempont az is, hogy a Kerettanterv által előírt informatikai területeknek mekkora részét és milyen módon fedi le az elkészült tananyag.

Lényegesnek tartom bemutatni a tananyagkészítés folyamatát is. Ennek először elméleti, majd gyakorlati megvalósulását is kifejtem. Bemutatom, hogy melyek az internetes tananyagokkal szemben támasztott követelmények, mik az előnyei és hátrányai egy ilyen oktatási eszköznek és hogy ezek hogyan jelennek meg a tárgyalt tananyagban. Részletes leírom az egyes leckék és kiegészítések tartalmát is.

Ennél a résznél technikai és esztétikai kérdésekre is kitérek, mivel a tananyagot nagyban meghatározza a megjelenése, megjelenítése. Röviden a magyar Scratch portált is bemutatom, hiszen ez ad helyet a tananyagnak és bizonyos funkciók, lehetőségek kimondottan a tananyagom számára, saját elképzeléseim alapján kerültek kivitelezésre a portál fejlesztője által.

A következő egység külön fejezetekben tárgyalja az anyag tanításához és tanulásához tartozó módszertani leírást tanárok és tanulók részére. A dolgozat végén gyerekekkel végzett első próbák eredményének ismertetése és elemzése található.

Mi is az a Scratch?

A mai, bizonytalanságon és állandó változáson alapuló társadalmunkban a kreatív gondolkodás képessége a siker és megelégedés kulcsa lehet mind a szakmai-, mind a magánéletben. A mai gyerekek számára tehát nagyon fontos a kreativitás, kreatív módon gondolkodás elsajátítása és fejlesztése, hogy az életük során adódó váratlan helyzetekben, problémák esetén innovatív megoldást találhassanak. A kreativitás fejlesztésének egyik megfelelő módja lehet saját játékok tervezése és elkészítése. Erre kiválóan alkalmas eszköz a Scratch. [12]

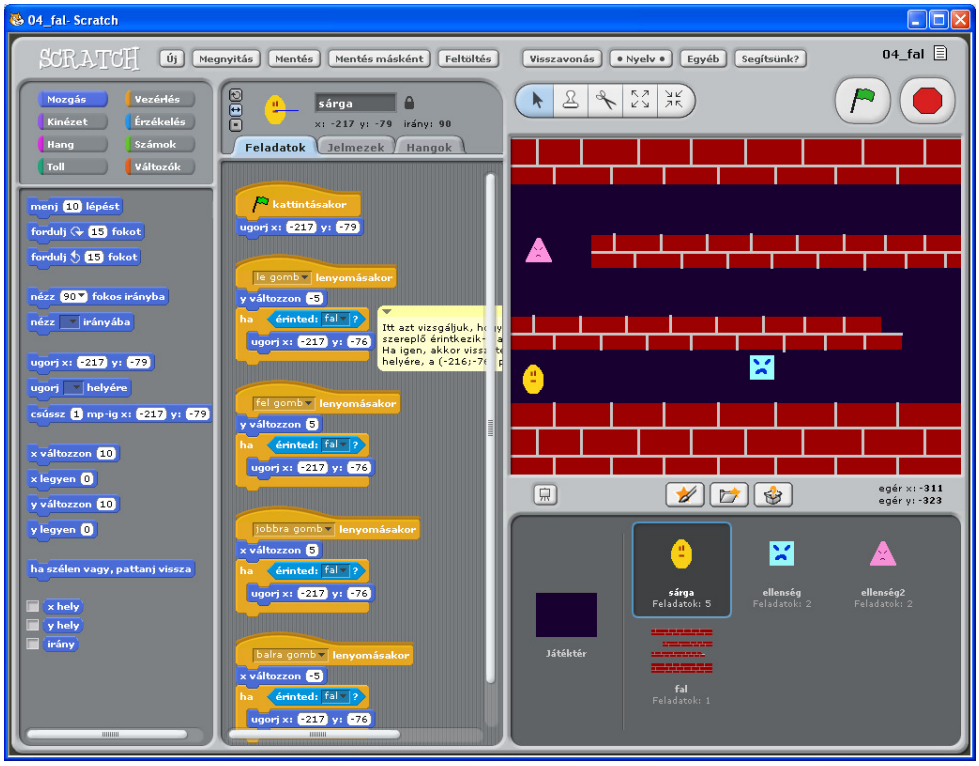
A Scratch egy kifejezetten gyerekek (8-18 éves korosztály) számára fejlesztett programozási környezet. Ingyenesen letölthető Windows és Mac OS X rendszerekre és kísérleti jelleggel már a különböző Linux disztribúciókra is. A tananyagom alapja a program 1.3.1-es verziója, amely több mint 40 nyelven, köztük magyarul is használható.

A Scratch egy Squeak-alapú interpretált, vizuális programozási nyelv. A dinamikus nyelvek csoportjába tartozik – a kód a program futása közben is változtatható. A fejlesztését 2004-ben kezdte a Lifelong Kindergarten csoport az MIT (Massachusetts Institute of Technology) egyetemen, és az első hivatalos verziója 2007-ben jelent meg. Azzal a céllal készült, hogy a programozással még csak ismerkedő gyerekek is könnyedén alkothassanak animációkat, játékokat. A fejlesztők ennek érdekében a gyerekek körében népszerű programok pozitívumait vették alapul a környezet elkészítésekor: az Etoys, a Logo Microworlds, az AgentSheets, a Boxer, az Alice2, sőt még a Macromedia Flash is szolgáltatott ötleteket a készítőknek a Scratch megalkotásakor.

A program az elkészülése óta eltelt két év alatt rendkívüli népszerűségre tett szert. Ez többek között az egyszerű kezelhetőségnek, a létrehozható látványos produktumnak és a közösségi élménynek köszönhető.

A parancsok a köznyelvi megfogalmazásokat követik, így a programozási gyakorlattal nem rendelkezők is könnyen tudják azokat használni (pl. menj 10 lépést; tűnj el). Emellett pedig a szintaktikai hibák lehetőségét (amelyek komoly gátat jelentenek a szöveg-alapú programozási nyelvek tanulásában) teljesen kizárták azzal, hogy az egyes parancsok különböző formájú színes elemeken helyezkednek el, és csak az összeillő darabok illeszthetők egymás után, vagy adott estben egymásba. Ennek

köszönhetően a Scratch-ben programot összeállító gyereknek a programozás mechanizmusára kevésbé kell koncentrálnia, inkább csak az adott probléma megoldására.



1. ábra

A Scratch-et elsősorban oktatási célokra fejlesztették és mára már világszerte sok országban használják az iskolai oktatásban. Az amerikai portálon külön fórumot hoztak létre a tanárok, oktatók számára, ahol azok megvitathatják a Scratch tanításával kapcsolatos elképzeléseiket és tapasztalataikat.

A portálra bármely regisztrált felhasználó feltöltheti az elkészült projektjeit közvetlenül a Scratch programból. A publikálhatóság, ezáltal a közösségi élmény megteremtése is a fejlesztők egyik fő célkitűzése volt. Így a projekt készítője megmutathatja a világnak, hogy mit alkotott. A feltöltött anyagokhoz pedig bárki hozzászólhat, kedvencnek jelölheti, galériákba rendezheti, és ami a legfontosabb: letöltheti. A projektek CC licenc (Creative Commons Attribution – Share Alike Licence) alatt állnak, ennek köszönhetően bárki, bármely projekt kódját megnézheti, letöltheti, és szabadon felhasználhatja. Ekkor a portálon természetesen megjelenik az is, hogy az átdolgozott projekt eredetije kitől származik. Ezeket az átdolgozásokat a Scratch világában remixeknek nevezik.

A Scratch fejlesztői nem véletlenül alakították a környezetet a fent említett elvek alapján. A fejlesztést hosszú megfigyelések előzték meg az MIT Media Laboratory által alapított Computer Clubhouses elnevezésű program keretein belül, amely kifejezetten nagyvárosi fiatalok számára szervez foglalkozásokat. Ezek célja, hogy szociálisan hátrányos helyzetű gyerekeknek „értelmes" iskola utáni időtöltést biztosítsanak. A „klubokba" járó gyerekek tevékenységeit megfigyelték, tanulmányozták, és az ebből levont következtetések, valamint a gyerekek által támasztott igények alapján fejlesztették a Scratch környezetet. Úgy gondolták, hogy ha a szociálisan hátrányos helyzetű fiataloknak tudnak készíteni egy olyan programot, amelyet azok szívesen használnak, s amely tanulásra motiválja őket, akkor ezt a programot bizonyára a szerencsésebb körülmények között élő gyerekek is szívesen használják majd. A megfigyelők a Computer Clubhouses fiatal tagjainak vizsgálata alapján a következő hat kritériumot támasztották egy ideális programozási környezettel szemben [13]:

1.) a gyerekek tartsák „menőnek" a környezetet és az általa elérhető aktivitásokat, feleljen meg az ízlésüknek és az érdeklődési körüknek.

Ez teljesen érthető elvárás, hiszen napjainkban már a felnőttek számára készült programok esetében is törekszenek arra, hogy a kezelőfelület minél látványosabb, esztétikusabb legyen, mivel akkor várhatósan sokkal többen és sokkal szívesebben fogják használni. Ez a gyerekeknek számára fejlesztett programok esetén még fontosabb. Fel kell kelteni az érdeklődésüket, amely célra például egy színes ábra jól megfelel. Sok fejlesztő elköveti azonban azt a hibát, hogy nem a célzott korosztálynak megfelelő képeket használ a program díszítésére, vagy a sok színes ábra mellett nem gondol arra, hogy az összhatás esztétikus legyen. Az ilyen programokat pedig nem biztos, hogy olyan szívesen használják a gyerekek, mint egy elvárásaiknak megfelelőt. Véleményem szerint a Scratch látványának tervezésekor és beépített képeinek megalkotásakor, illetve kiválasztásakor jó munkát végeztek a készítők. A képek között rajzolt ábrák, és fényképek egyaránt megtalálhatók. A rajzok is többféle stílusban készültek, így a legkülönbözőbb ízlésvilágú gyerekek is találhatnak olyan képeket, amelyeket szívesen beépítenek projektjükbe. Ha pedig mégsem, akkor a beépített festőablakban nekik tetsző rajzot készíthetnek, amelyet később szereplőként vagy háttérként használhatnak fel.

A Scratch ablak kinézete is olyan, amely különböző korosztályba tartozó gyerekek tetszésének is megfelel. A parancsok színes blokkjai játékossá (de nem infantilissá) teszik a program ablakának letisztult szürkéjét, a játéktér pedig teljesen egyéni módon díszíthető a már imént említett tetszőleges háttér és szereplők beillesztésével.

2.) lehessen azonnal látni a környezet előnyeit és felfedezni a benne rejlő lehetőségeket

Ez a pont gyakorlatilag a következő elvárásokat fogalmazza meg: a környezet legyen átlátható és könnyen kezelhető. Az átláthatóság sok mindenre vonatkozhat. Átlátható egy program, ha a lehetőségei logikus csoportosításban, könnyen elérhetően helyezkednek el, de átlátható akkor is, ha egyszerűen arról van szó, hogy a felhasználó nem érzi túlzsúfoltnak. Az egyszerű kezelhetőség részben összefügg ezzel: egy átláthatatlanul bonyolult és elágazó menürendszerrel rendelkező programot aligha nevezhetünk könnyen kezelhetőnek. Ez az elvárás azt is jelenti, hogy a program kezelése legyen nyilvánvaló, egy-egy funkció eléréséhez ne kelljen órákig böngésznünk a súgóját. A készítőknek ezen előzetes elvárást is sikerült teljesíteniük. A Scratch teljesen átlátható, amit a különböző színek használata még inkább elősegít. Többszintű menürendszerek nincsenek benne. Tulajdonképpen minden funkció elérhető gombok megnyomásával vagy lapfülek kiválasztásával; legfeljebb két kattintással. Az egyszerű kezelhetőség legjobb példája pedig az egyes összeillő elemekből „fogd és vidd” technikával történő programkészítés. A gyerekek számára ez a programozási mód azért is „ismerős”, vagy egyszerűen elsajátítható, mert szinte kivétel nélkül mindannyian játszottak LEGO kockákkal. A Scratch megalkotói pedig ezt az ismert játékot vették alapul a programblokkok összeépíthetőségének módjánál.

3.) már a legelső projektet is könnyen és gyorsan lehessen elkészíteni

A 21. században felnövő gyerekek már „felgyorsult világban élnek”. Hozzászoktak a szinte megállás nélkül feléjük áramló információkhoz és őket érő ingerekhez (televízió, számítógép, internet… stb.) Ilyen körülmények között egyre nehezebb felkelteni – de még inkább fenntartani – az érdeklődésüket bármi iránt is. Ezért fontos, hogy a program első megnyitása után szinte azonnal lássanak eredményt a gyerekek. Ez felkelti az érdeklődésüket, ugyanakkor motiválja őket további projektek készítésére és ennek érdekében a környezet alaposabb megismerésére. A Scratch-ben az első sikerek könnyű elérése garantált. Az alapértelmezett macska szereplőnek elég egy, akár két parancsból

álló feladatot adni ahhoz, hogy „életre keljen”. Például a zöld zászló kattintása után (azaz a program indításakor) mondja azt, hogy „Miau”. Ezt követően a Scratch-et használó gyerek valószínűleg kíváncsi lesz rá, hogy hogyan lehet másra is „rávenni” a macskát vagy hogy hogyan használhat más szereplőket, tehát további projekteket is fog készíteni az első után.

4.) a gyerek olyan „termékeket” készíthessenek a segítségével, amelyekkel dicsekedhet a barátai előtt

A gyerekek általában büszkék az alkotásaikra. A rajzaikat megmutatják az óvónőnek, tanárnak, szülőnek. Idősebb korukban már inkább a barátaik elismerését keresik. Szeretnének minél szebb, élvezetesebb projekteket készíteni. Az animációkat teletűzdelik multimédia elemekkel, a játékaikat szereplőkkel és minél rafináltabb megoldásokkal. De nem elég, ha csak a saját számítógépük monitorán látható az a játék. Szeretnék megosztani interneten, mobiltelefonokon és esetleg más, egyre divatosabb elektronikai eszközökön is.

A készítőknek mindkét elvárásnak sikerült megfelelni. A Scratch használatával olyan projektek készíthetők, amelyekre igazán büszke lehet a gazdájuk. A projektekbe képek és hangok illeszthetők. Emellett a sok gyerek által kedvelt Gimp vagy Photoshop egyes képmanipulációs eszközei is helyet kaptak a környezetben. Itt azonban a fent említett programoktól eltérő módon használhatók. Például különböző matematikai függvények segítségével megadható, hogy mikor milyen mértékben érvényesüljön egy hatás egy bizonyos képre. Így kis türelemmel és odafigyeléssel akár videoklipek vagy egyéb filmszerű projektek is készíthetők. Ezek megosztása a Scratch ablakából egyetlen kattintással történik. A portálra feltöltött programokat pedig az egész világ megcsodálhatja. Így elég a barátoknak egy linket küldeni e-mailben vagy valamilyen más kommunikációs eszközön és ők is bármikor megnézhetik az elkészült művet. És akkor a többi „scratch-előről” még nem is beszéltünk, akik a portálon böngészve bukkanhatnak rá az adott projektre.

5.) a környezet kínáljon sok különböző aktivitási lehetőséget

Ez szintén teljesen jogos elvárás. Hiába felel meg egy program az eddigi összes elvárásnak, ha csak egyféle dologra alkalmas. A túl specifikus programozási nyelvek és környezetek nem elégítik ki a gyerekek változatosság iránti igényét. Hamar megunják, ha kiaknázták a benne rejlő összes lehetőséget. Ha valamelyik gyereknek pedig nem

tetszik az a fajta tevékenység, amelyet a program kínál, akkor még rövid ideig sem fogja élvezni a használatát. A Scratch-ben ilyen problémák nem fordulhatnak elő. Az eddig leírtakból kiderül, hogy szinte minden gyerek találhat benne neki tetsző alkalmazási területet: készíthet animációt vagy játékot, illeszthet bele képet, hangot, rajzolhat szereplőt, hátteret és alkalmazhatja ezek tetszőleges kombinációját.

6.) a környezet fokozatosan mindig újabb ismeretekkel szolgáljon

Ez a pont tulajdonképpen az előző egy változata, azzal a különbséggel, hogy ebben megjelenik a fejlődés iránti igény is. Ha a gyerekek már tudják kezelni a környezetet és megtanulták a programozási alapokat, akkor felmerülhet bennük az igény változók, vagy bonyolultabb adatszerkezetek megismerésére és használatára. Erre van lehetőség a Scratch-ben, hiszen a projektekben lehet változókat, listákat létrehozni és használni. Azonban úgy gondolom, hogy a folyamatos újabb ismeretek iránti elvárást egy programozási környezet nem tudja önmagában teljesíteni. Szükség van ilyenkor egy informatikatanárra, más szakemberre, vagy egy szakértő kortársra, aki vezeti a gyereket, továbbá megérteti vele, hogy ha minden parancs használatát ismeri, az még nem jelenti azt, hogy a környezet összes lehetőségét kiaknázta.

A Scratch tehát a felsorolt elvárások mindegyikének – amennyire lehet –, maximálisan megfelel. Így alkalmas a legkülönbözőbb társadalmi helyzetű gyerekek figyelmének felkeltésére, motiválására, ennek következtében pedig az iskolai oktatásban való felhasználásra, vagy legalább a minél szélesebb körben való megismertetésre.

A Scratch Magyarországon már 2007 nyarán kezdett ismertté válni, miután Balaton Marcell Balázs és Bernát Péter lefordították a programot. Ebben az időszakban a magyar internetes sajtó is foglalkozott ezzel az új, egyszerűen kezelhető, gyerekeknek szánt programkörnyezettel. Nem sokkal ezután pedig a fordítók elkésztették a magyar portált is néhány egyetemi hallgató közreműködésével Turcsányiné Szabó Márta Telementorálás kurzusainak keretein belül. A 2008-as Kihívás versenyben a csapatok már Scratch-ben is elkészíthették programjaikat. Ehhez szükség volt valamilyen segédletre, hogy a gyerekek tudják használni a még tanáraik számára is ismeretlen új környezetet. Ekkor készült az első magyar nyelvű tananyag. Segítségével a Kihívás verseny résztvevői ötletes, használható projekteket készítettek. A viszonylagos siker ellenére más típusú leckékre volt szükség, mivel azok csak a kifejezetten informatikai érdeklődésűeket fogták meg, a többi gyerek számára túl hosszúak voltak a magyarázó

szövegek. Ennek hatására született az igény egy más típusú, más felépítésű, tömörebb és egységesebb tananyagra.

Ennek készítését azért mertem vállalni, mert az elmúlt évben alaposabban megismertem a Scratch környezetet. Részt vettem az első tananyag leckéinek szövegezésében, majd a Kihívás versenyre érkezett versenymunkák értékelésében is. Emellett Balaton Marcell Balázzsal közösen fordítottuk magyarra az amerikai Scratch portált. Az informatika tanítási gyakorlatom során pedig részben rálátást nyertem arra is, hogy a felső tagozatos diákoknak milyen tényleges informatikai ismereteik vannak, hogyan viszonyulnak a tantárgyhoz és a programozáshoz. Ezen tapasztalatokat igyekeztem felhasználni a tananyag elkészítése során.

A NAT és a Kerettanterv elemzése a tananyag szempontjából

A tankönyvek és tananyagok tartalmát erősen befolyásolja, hogy a Nemzeti Alaptanterv és a Kerettanterv mit ír elő egy-egy témáról egy adott korosztály számára. A Scratch tananyag elsősorban 10-14 éves gyerekek számára készült, természetesen az informatika műveltségi területhez kapcsolódóan. Fontos tehát, hogy az általános iskola felső tagozatára előírt elvárásoknak megfeleljen mind a NAT, mind a Kerettanterv szerint. A következőkben megvizsgálom elsősorban az algoritmusok és adatok, multimédia ismeretek, valamint a hálózati kommunikáció témakörökre koncentrálva, hogy mit ír elő ezen korosztályról és a témák tanításáról az említett két dokumentum.

NAT – kompetenciák

A 2003-ban bevezetett és 2007-ben módosított Nemzeti Alaptanterv az elsajátítandó tudás és képességek megszerzését a kulcskompetenciákhoz kötődően ismerteti. „*A kulcskompetenciák azok a kompetenciák, amelyekre minden egyénnek szüksége van személyes boldogulásához és fejlődéséhez, az aktív állampolgári léthez, a társadalmi beilleszkedéshez és a munkához*” [1].

A szakdolgozat témáját szolgáltató tananyag az informatika műveltségi területhez kapcsolódó, elsősorban önálló tanulásra szánt anyag. Tehát a hozzá köthető kompetenciák a következők:

Digitális kompetencia

„*A digitális kompetencia felöleli az információs társadalom technológiáinak magabiztos és kritikus használatát a munka, a kommunikáció és a szabadidő terén. Ez a következő készségeken, tevékenységeken alapul: információ felismerése, visszakeresése, értékelése, tárolása, előállítása, bemutatása és cseréje; továbbá kommunikáció és hálózati együttműködés az interneten keresztül.*” [1]

Az informatika tanításához, tanulásához természetesen elengedhetetlen a digitális kompetencia kialakítása és fejlesztése. Esetünkben a legszükségesebb képességek az informatikai eszközök használatához, az internethasználathoz és az infotechnológiához kapcsolódnak. A gyereknek kompetensnek kell lennie az informatikai eszközök használatában, ha egy olyan tananyagot szeretne elsajátítani, amely digitális formában,

interneten keresztül érhető el. Ehhez kapcsolódik az internethasználatra való képesség is. Az infotecnhnológia az algoritmizáláshoz, programozáshoz kapcsolódó területek összefoglalása. Ezek ismerete, az elsajátításukra való képesség a tananyag témájából adódóan fontos.

Matematikai kompetencia

„*A matematikai kompetencia a matematikai gondolkodás fejlesztésének és alkalmazásának képessége. (…) A matematikai kompetencia (…) felöleli a matematikai gondolkodásmódhoz kapcsolódó képességek alakulását, használatát, a matematikai modellek alkalmazását (képletek, modellek, struktúrák, grafikonok/táblázatok).*" [1]

A matematikai kompetencia szorosan kapcsolódik az informatikához, azon belül is az infotechnológiához. A struktúrákban gondolkodás és a modellalkotás képessége vezet el az algoritmikus gondolkodás kialakulásához, amely elengedhetetlen algoritmusok megértéséhez, készítéséhez és alkalmazásához.

Ezen felül bizonyos algoritmusok megalkotásához nélkülözhetetlen a matematikai gondolkodásmód és egyes matematikai algoritmusok, képletek, geometriai alakzatok tulajdonságainak ismerete. Az algoritmusok és adatok témakör kezdetén inkább a geometriai (irányok, szögek, távolságok), később pedig elsősorban az algebrai látásmódra, ismeretekre van szükség (relációk, változók, műveletek).

Hatékony, önálló tanulás

„*A hatékony, önálló tanulás azt jelenti, hogy az egyén képes kitartóan tanulni, saját tanulását megszervezni (…). Felismeri szükségleteit és lehetőségeit, ismeri a tanulás folyamatát. (…) A motiváció és a magabiztosság e kompetencia elengedhetetlen eleme.*" [1]

Az önálló tanulás azért került a szükséges kompetenciák közé e felsorolásban, mivel a szakdolgozat témája önálló feldolgozásra szánt tananyag. Egy ilyen anyag elsajátításához pedig szükség van arra, hogy a tanuló állandó tanári, szülői segítség nélkül is képes legyen tanulni. Ennek egyik lényeges feltétele, hogy a gyerek kialakult, hatékony tanulási stratégiával rendelkezzen. Az ilyen stratégiával rendelkező gyerekek egyedül is képesek meglátni a tananyag összefüggéseit, magabiztosak, bíznak az anyag elsajátításának sikerében és erre megfelelően motiváltak.

Esztétikai-művészeti tudatosság és kifejezőképesség

„Az esztétikai-művészeti tudatosság és kifejezőképesség magában foglalja az esztétikai megismerés, illetve elképzelések, élmények és érzések kreatív kifejezése fontosságának elismerését mind a tradicionális művészetek nyelvein, illetve a média segítségével” [1]

Bár e kompetencia nem kapcsolódik szorosan az infotechnológiához, az algoritmusok és adatok témakörhöz, mégis úgy gondolom, hogy az informatika tanítása során is fontos, hogy a tanár által bemutatott anyagok és a gyerek által létrehozott produktum esztétikus, szép legyen. Törekedni kell arra, hogy a gyerek is érezze ennek szükségességét. Egyes tanulókat pedig motiválhat a feladatmegoldásra, ha munkájának végeredménye valóban szép ábrákat, képeket, szereplőket tartalmaz.

NAT – Informatika műveltségi terület

A kulcskompetenciák tárgyalását követően a NAT műveltségi területekre bontva is felsorolja a kifejlesztendő vagy elsajátítandó kompetenciákat. Ez már természetesen hosszabb leírás, az egyes műveltségi területek témáihoz részletesen meghatározza az összes elvárt tudáselemet.

A következőkben az informatika műveltségi terület azon részeinek összefoglalása olvasható, három korcsoportra külön-külön meghatározva, amelyek valamilyen módon kapcsolódnak az elkészült tananyaghoz. Ezek a következők: informatikai eszközök használata, informatika-alkalmazói ismeretek, infotechnológia, infokommunikáció, valamint médiainformatika.

1-4. évfolyam

A gyerekek ebben az életkorban általában még csak ismerkednek a számítógéppel, az informatikával. A legfontosabb, hogy elsajátítsák az informatikai eszközök használatának módját. Emellett azonban már ismerkedhetnek különböző alkalmazásokkal is. Elvárható tőlük egyszerűbb rajzok, animációk készítése, és ezzel együtt a készítéshez szükséges programok alapvető funkcióinak használata. Ezen kívül fel kell ismerniük és végre kell hajtaniuk egyszerű algoritmusokat is, továbbá képesnek kell lenniük értelmezni az algoritmusokban használt adatokat.

A Scratch már e korosztály számára is biztosít lehetőségeket a NAT által előírt képességek megszerzésére. A festőablakban egyszerűen készíthetnek rajzokat, amelyek

segítségével később néhány könnyen megérthető parancs segítségével animációkat, vagy nagyon egyszerű játékokat is összerakhatnak. Ezek előállításához szükség van egy bizonyos fokú algoritmus-ismeretre, tehát ebben is jártasságot szerezhetnek a tanulók.

5-6. évfolyam

Ennek a korosztálynak már tudatosan kell használnia az informatikai eszközöket. Önállóan kell megoldaniuk informatikai problémákat, tehát képesnek kell lenniük a probléma megoldására alkalmas program kiválasztására. Meg kell fogalmazniuk egy adott feladat megoldásához tartozó algoritmust, valamint meg kell oldaniuk feladatokat valamely egyszerű fejlesztő rendszerrel. Ezen felül képesnek kell lenniük weboldalak betöltésére, internetes portálok használatára.

A Scratch a 11-12 évesektől elvárt szintű algoritmusok megvalósítására is alkalmas. Ezen kívül közvetetten a portálok használatát is megtanulhatják a környezet használatával, hiszen onnan közvetlenül feltölthetők az amerikai Scratch portálra az elkészült projektek. A projektjeik megtekintése, rendszerezése, mások munkáinak kipróbálása, értékelése során a gyerekek megtanulhatják a portálok használatának elveit és módját.

7-8. évfolyam

Ebben az életkorban a tanulóknak már képesnek kell lenniük tájékozódni a különböző informatikai környezetekben. Multimédiás dokumentumokat – például bonyolultabb animációkat – kell létrehozniuk. Összetettebb feladatokhoz kell algoritmust készíteniük és az elkészült algoritmust valamely fejlesztő rendszerrel megvalósítaniuk.

A Scratch-ben a bonyolultabb animációk készítéséhez alaposabb programozási ismeretek, algoritmizáló készség is szükséges. Tehát a tanulók e környezet használatával egyszerre fejleszthetik tudásukat a multimédia, valamint az algoritmusok és adatok témakörön belül. A környezet természetesen alkalmas ezen felül olyan algoritmusok megvalósítására is, amelyek a 13-14 éves korosztálytól várhatók el.

Kerettanterv

A kerettanterv konkrétabban megfogalmazza, hogy tantárgyanként melyek azok a témák, és az azokon belüli altémák, amelyeket egy-egy évfolyamon meg kell ismerniük a gyerekeknek, valamint előírja a továbbhaladás feltételeit is [7]. Ezeket szintén a tananyag szempontjából vizsgálom, évfolyamonkénti bontásban.

6. évfolyam

- Algoritmusok és adatok
    - A tanulóknak fel kell ismerniük és meg kell tudniuk fogalmazni a hétköznapokban alkalmazott algoritmusokat. Meg kell ismerkedniük a térbeli tájékozódási képességet fejlesztő, egyszerű Logo-algoritmusokkal. Ezek használatát, létrehozásuk módját is ismerniük kell. Elvárás továbbá a szekvenciális vezérlést, valamint számlálós ciklusokat tartalmazó programok futtatása és működésük értelmezése.
- Szöveg, kép és zene
    - A gyerekeknek számítógépen egyszerű rajzokat kell készíteniük. Ehhez ki kell választaniuk a megfelelő rajzeszközt.

Ebben az évfolyamban tehát a tanulók már használják valamelyik Logo rendszert és megismerkednek a szekvenciális vezérléssel, valamint a ciklusokkal. Ezen kívül tudnak egyszerűbb rajzokat készíteni, azaz hátteret vagy szereplőt rajzolni egy általuk készített meséhez vagy esetleg játékhoz.

Ezeket a Scratch környezetben is megtehetik. A festőablakban lehetőség van rajzok készítésére. Ez egy kisebb gyerekek számára is egyértelműen kezelhető rajzolóprogram, amelyben minden olyan funkció megtalálható, amelyre szereplők vagy hátterek rajzolásához szükség lehet. A tananyag pedig megismerteti a szekvenciális vezérlés és a számlálós ciklusok használatát és az abban rejlő lehetőségeket.

7. évfolyam

- Algoritmusok és adatok
    - A gyerekeknek ismerniük és értelmezniük kell a szekvenciális és a ciklusos vezérlést. Ezek kódolását és kipróbálását is végre kell

hajtaniuk. Tudniuk kell továbbá grafikával és szöveggel kapcsolatos programokat készíteniük.

- Kommunikáció a hálózaton
  - Ismerniük és használniuk kell legalább egy levelezőprogramot. Képesnek kell lenniük elektronikus levelek küldésére és fogadására, állományok levélhez csatolására. Ismerniük kell az e-mail cím létrehozásának módját, és ezt alkalmazniuk is kell. Tudniuk kell használni egy böngészőt, valamint annak segítségével hasznos weboldalakat önállóan felkeresni.

A tanulók már szekvenciákat és ciklusokat is tudnak használni, tehát mesék (animációk) mellett már interaktív képregényt vagy egyszerű játékprogramot is készíthetnek. Ezeket fel tudják tölteni valamely internetes portálra, vagy egy e-mailhez tudják csatolni. Tehát az elkészült programot meg tudják osztani másokkal is.

A tananyag ezen ismeretek elsajátítását is segíti, egyesek pedig szükségesek a tanulásához, például egy internetes portál (a magyar Scratch portál) önálló felkeresése, hiszen ott található a tananyag.

A Scratch környezet nagyon jól alkalmazható animáció-készítés során. A tananyag részletesen bemutatja ezek elkészítésének lehetőségeit, több leckén keresztül tárgyalja a mesekészítéshez szükséges elemek, parancsok használatának módját. Ugyanezen eszközök segítségével egyszerűbb interaktív programok, játékok is készíthetők, tehát a tananyag és a Scratch környezet segítségével a gyerekek elsajátíthatják a hetedik évfolyamosokkal szemben támasztott követelményeket

8. évfolyam

- Algoritmusok és adatok
  - A tanulóknak az eddigiek mellett meg kell ismerniük a feltételes vezérlés használatát. Képesnek kell lenniük számlálós és feltételes ciklust tartalmazó programok értelmezésére, kódolására, kipróbálására. Algoritmizálás segítségével kell megoldaniuk egyszerű matematikai és logikai feladatokat. Tudniuk kell különbséget tenni az eltérő számtípusok között és használni azokat.

Ebben az évfolyamban a gyerekek már bonyolultabb algoritmusokat képesek készíteni és alkalmazni, tehát már bonyolultabb játékokat is tudnak készíteni.

A tananyag bemutatja a játékkészítéshez szükséges parancsokat, valamint néhány fortélyt is, amelyek jól alkalmazhatók különböző játékok tervezése és kódolása esetén. Ezen kívül példát ad számokat használó programok készítésére is, tehát ezen évfolyam számára készült tananyagok előírásaival szemben is megállja a helyét.

A Scratch az oktatásban

A Kerettanterv, valamint a Nemzeti Alaptanterv követelményeit, valamint a Scratch környezet tulajdonságait, lehetőségeit összevetve elmondható, hogy a két dokumentum által az általános iskolások számára előírt készségek és ismeretek döntő többsége e környezet és a hozzá kapcsolódó internetes portálok használatával is megszerezhető, illetve fejleszthető. Néhány elvárt ismeret azonban csak közvetetten tanítható meg a Scratch használatán keresztül.

A Scratch-ben megtalálhatók, illetve megvalósíthatók a következők:

- szekvencia
- egyágú (ha) és kétágú (ha… különben) elágazás
- számlálós (ismételd *n*-szer), végtelen (mindig) és feltételes (mindig ha, ismételd eddig) ciklus
- többszálúság
- eseményvezérlés
- változók
- listák

A fent említett végtelen ciklusnak sok programozási nyelvben nincs értelme, a Scratch-ben viszont szinte elengedhetetlen az eseményvezérlés megvalósítása miatt. Már a program indulásától kezdve egy mindig ciklusba ágyazott feltételekkel folyamatosan ellenőrizhető például, hogy bizonyos billentyűket éppen lenyomtak-e.

Nincs lehetőség azonban a következők megvalósítására:

- rekurzió
- eljárás
- függvény

Az informatika tanárok az eljárásokat hiányolhatják leginkább a környezetből. A többi programozási nyelvhez hasonló módon valóban nem lehet eljárásokat létrehozni, nincs paraméterátadás sem, azonban eljáráshoz hasonló szerkezet, „kvázi-eljárás” létrehozható, mégpedig a Scratch egy speciális lehetősége, az üzenetküldés segítségével. Az üzenetek elsősorban a szereplők közötti kommunikációt segítik (de egy szereplő a saját maga által küldött üzenetre is reagálhat). Egy egyszerű eljárás a következőképpen valósítható meg ebben a környezetben:

```
Eljárás Növeld (változó n: egész)
    n:=n+1
Eljárás vége
```

A szereplő:

```
küldj üzenetet: Növeld
```

B szereplő:

```
Növeld üzenet érkezésekor
n változzon 1
```

Tehát konkrét eljárások nélkül is átadható az eljárásokban gondolkodás szemlélete a gyerekeknek, a 10-14 éves korosztálytól elvárt ismeretanyag így a Scratch környezet segítségével elsajátítható.

A módszertanról szóló fejezetben a Scratch oktatásban való felhasználásának módjai részletesen kifejtésre kerülnek.

A tananyag bemutatása

A tananyag besorolása

Az utóbbi években egyre többször hallhatjuk az e-learning kifejezést. Divat lett használni a szót és az általa jelzett tanulási formát egyaránt. Azonban ez a szóhasználat nem mindig helyes. Az e-learning a következő három, jól elkülöníthető összetevőből áll:

- számítógéppel segített tanulás: olyan oktatási forma, melyben felhasználják a multimédiás számítógép adta lehetőségeket, sőt esetenként köré szervezik az egész tanulási folyamatot.
- távoktatás: olyan sajátos oktatási/képzési forma, melyben a tanuló a képzési idő nagyobb részében egyedül, önállóan tanul, kisebb részében pedig konzultációkon vesz részt, ahol személyes kapcsolat során, közvetlen irányítás mellett mélyíti önállóan szerzett ismereteit, gyakorol és fejleszti képességeit tanárai (azaz tutorai) segítségével.
- internetes, web alapú tanulás: elsősorban az internetes keresésen (a tanuló szinte korlátlan információmennyiséghez hozzáférhet) és kommunikáción alapuló oktatási forma. [8]

E hármas együttes alkalmazásából áll össze az e-learning. Számítógépes adatbázisokon keresztül érhetők el a tananyagok, amelyek a tanulás hatékonyságának növelése érdekében sok képet, ábrát, multimédiás kiegészítést valamint oktató- és mérőprogramokat tartalmaznak. Ezen felül az e-learning rendszerekben a tanulóknak lehetősége van egymás között vagy a tutoraikkal megbeszélni az anyag kérdéses pontjait.

A szakdolgozat témájaként szolgáló tananyag több pontot is teljesít a fent említettek közül, ennek ellenére nem nevezhető e-learning tananyagnak. Ha megvizsgáljuk az e-learning összetevőinek leírását, akkor megállapíthatjuk, hogy a tananyag leginkább a számítógéppel segített tanulás kategóriájába tartozik. Más elemek is megjelennek benne (pl. mentorálás a fórumon keresztül), de azok kevésbé meghatározók.

A hipertext által segített oktatás

A tananyag csak oktatófüzet formában is elkészülhetett volna (ilyen változat is létezik, ez letölthető a honlapról), mivel az interaktív elemek nem töltenek be nélkülözhetetlen szerepet a megértésben. Mégis inkább a webes megjelenítési formát választottam. Ennek egyik oka az volt, hogy a tananyag így sokkal szélesebb réteghez juthat el, mint papíron. A másik pedig az, hogy a gyerekek a tankönyveket, tanulást segítő füzeteket az iskolai oktatáshoz kötik, sokaknak kellemetlen élményeik, negatív érzéseik lehetnek velük kapcsolatban. Ezzel szemben egy internetes tananyag tanulmányozása közben talán nem érzik olyan erősen, hogy az abban leírtakat „kötelező megtanulni", így esetleg szívesebben foglalkoznak vele. Ez persze részben abból adódik, hogy napjainkban jellemzően még mindig a hagyományos, papír alapú tankönyvekkel találkoznak az iskolai oktatásban. Így egy digitális tananyag az újdonság varázsával is hat a tanulókra.

A tananyag tehát hipertext formátumú. A hipertext a hagyományos szöveghez képest annyiban több, hogy egyes speciális elemei (linkek) a szöveg egy más részét vagy más szöveget jelenítenek meg. Ez a felépítés – az asszociatív indexelés – sokkal inkább megfelel az emberi gondolkodásnak, mint a könyveké [4]. Ráadásul így akár csak linkek használatával is megoldható, hogy a tanulók megértsék a tananyag szerkezetét, lássák az összefüggéseket bizonyos anyagrészek között.

A tananyagomban a linkeknek két fajtája van: némelyek egy másik leckére visznek, máshol pedig bizonyos fogalmak, parancsok magyarázatát az adott szó mellett felbukkanó buborékból tudhatjuk meg. A könyvekben ezek a magyarázatok lábjegyzetben vagy a könyv végi fogalomtárban szerepelnének. Az információértékük azonos, azonban ilyen formában interaktivitás-érzést keltenek – annak ellenére, hogy ez csak látszólagos interaktivitás (az egeret a fogalom fölé húzva jelenik meg a jegyzet).

Cél: az algoritmikus gondolkodás fejlesztése játékos módon

Napjainkban nagy hangsúlyt kap az oktatásban a gondolkodásfejlesztés. Ez a tananyag céljai között is szerepel, ám speciálisabb formában: az anyag célja az algoritmikus gondolkodás fejlesztése. Ennek egyik nyilvánvaló oka, hogy az informatikában gyakran van szükség ilyen típusú gondolkodásmódra a különböző feladatok megoldása során. A másik ok – és talán ez a fontosabb, hiszen túlmutat egy

tantárgy vagy az egész iskolai oktatás keretein –, hogy a gyakorlati életben is szükségünk van arra, hogy algoritmusokban gondolkozzunk. így találjuk meg a rendet a megoldásra váró problémában és magában a világban is.

Fontos, hogy „a gyerekek ne műveletekben, hanem eljárásokban, megoldásmódokban, tervekben, műveletek folyamatában gondolkozzanak. Ezáltal valódi, azaz rugalmas, más helyzetekben is alkalmazható, nagyobb transzferhatású tudás jöhet létre.” [15]

Az, hogy a tananyag – illetve annak alkalmazásával a pedagógus – konkrétan hogyan segítheti elő az algoritmikus gondolkodás kialakulását, illetve fejlődését, a módszertanról szóló fejezetben részletes kifejtésre kerül.

Internetes tananyagok tervezése

A tananyagtervezés általában három fő részből áll. Ezek a didaktikai tervezés, modularitás, valamint a rendszerszemlélet megtervezése. Az utóbbiak inkább oktatási keretrendszerek elvárásai, de az egyéni tananyagok esetén is hasznos lehet az alkalmazásuk.

Didaktikai tervezés

„A didaktikai tervezés a tananyag, a tanulási program és a tanulási környezet optimális hatásegyüttesének kialakítása” [8]. A didaktikai tervezés további részfolyamatokból áll:

A tananyag tartalmának meghatározása

Természetesen a legfontosabb lépés annak eldöntése, hogy mi az, amiről a tananyag szól, milyen témákat érintsen, milyen fogalmakat magyarázzon el. Azonban nem elég csupán a tananyag tartalmát megtervezni, azzal együtt a szerkezetét is fel kell vázolni (legalább gondolatban). A tartalom és a szerkezet ugyanis sok esetben meghatározzák egymást. Ezeket érdemes felváltva, „felülről lefelé haladva” tervezni. Tehát először az egész anyag tartalmát kell átgondolni, majd azt, hogy ez milyen szerkezeti egységekből fog állni. Ezután az egyes szerkezeti egységek tartalma tervezhető, azt követően pedig az, hogy ezeket milyen kisebb részekre lehet bontani. Ez így folytatódhat tovább egészen a legkisebb egységekig.

A szöveg didaktikai feldolgozása

Ebben a lépésben kell eldöntenünk, hogy a tananyag milyen fejlesztési céllal készült, valamint hogy ezt a célt milyen eszközökkel kívánjuk elérni. El kell döntenünk, hogy milyen eszközöket akarunk használni kompetenciafejlesztésre, illetve ismeretbővítésre és ezek konkrétan hogyan jelennek meg a tananyagban.

Ebben a pontban említem a szöveg formai tervezését is. A tanulási folyamat eredményességét az is befolyásolja, hogy milyen a megtanulandó anyag külső megjelenése. Ez gyerekek esetén számít igazán. A nekik készült anyagokban különösen átgondoltan kell megtervezni a szöveg tipográfiáját (megfelelő betűtípus és –fokozat, margók nagysága, behúzások mértéke, kiemelések, színek). A legfontosabb szempontok az olvashatóság és az egységesség.

Hipertextstruktúra kialakítása

Az előzőekben bemutatásra került a hipertext által segített oktatás, melynek legfontosabb eleme a link. A hipertextstruktúra kialakítása azt jelenti, hogy gondosan megtervezzük, mely szavakat alakítjuk linkké, és hogy az egyes linkek hová mutassanak. Az így kialakított szerkezet nagyban befolyásolja az egész tananyag érthetőségét, koherenciáját és még azt is, hogy milyen tanulási stílusú diákok számára lesz a legmegfelelőbb. A linkek használata tehát alapos átgondolást igényel a tananyag tervezőjéről. Arra pedig különösen figyelni kell (főként kisebb gyerekeknek szánt tananyag esetén), hogy a linkek ne alkossanak „útvesztőt”, ne lehessen eltévedni a sok elágazásban. Inkább legyen kevesebb, de átlátható szerkezetű út a tananyagban.

Multimédia-elemek integrációja

Az internetes tananyagok nagy előnye a tankönyvekhez képest, hogy képek, hangok, videók, animációk beillesztésével elősegíthetjük a megértést és sokkal látványosabbá tehetjük a megjelenését. Sajnos sok tananyagfejlesztő csak az utóbbi szempontot veszi figyelembe a multimédia elemek használata során. Ez azonban nem segíti elő (egyes esetekben még hátráltathatja is) a tananyagban leírt tudásanyag elsajátítását. „Az interaktív kommunikációs betéteknek, az auditív és vizuális elemeknek a tananyag szerves, esszenciális összetevőiként kell megjelenniük” [8]. Használatuk csak ekkor indokolt.

Néha azonban nagyon nehéz eldönteni, hogy melyek azok az elemek, amelyek valóban támogatják a tananyag elsajátítását, és melyek azok, amelyek csak „dekorációként” szolgálnak. Ez főként interaktív animációk esetén jelenthet problémát (akár még az animáció készítőjének számára is). A megfelelő interaktív elemeket tartalmazó tananyag elősegíti a gyerek önálló tanulását, viszont a mindössze látványos, ám tanulást nem támogató interaktivitás csak akadályozza azt.

Tanulástámogató visszacsatolások beillesztése

Itt következik a mentorálás megtervezése. A tananyag tervezőjének végig kell gondolnia, hogy szinkron vagy aszinkron kapcsolattartást kell-e létrehozni a tananyag tanulói és az ő mentoraik között. Ettől függ, hogy chat-szobát, illetve fórumot vagy esetleg blogot hoz-e létre.

Mérés és értékelés megtervezése

Ennek a pontnak a témája nem jelenik meg minden tananyagban, azonban nagyon hasznos lehet például e-learning anyagok esetén. Az értékelés kétféle lehet: egyéni vagy csoportos. Egyéni értékelés esetén a tanuló a saját régebbi eredményeihez képest kap visszajelzést a teljesítményéről, csoportos értékelés esetén pedig egy csoportátlaghoz (vagy inkább elváráshoz) viszonyítva. Az első értékelési forma felkeltheti, vagy fenntarthatja a tanuló motivációját, míg a másik a mentornak, oktatónak szolgálhat hasznos információkkal a diák haladásáról.

Modularitás

„A modularitás a komplexitás kezelésére és a sokféleség iránti igény kielégítésére irányuló rendszerszervező törekvés” [8]. A modularitás valójában azt az igényt fejezi ki, hogy a tananyag könnyen illeszthető legyen az e-learning oktatási keretrendszerekbe. Ehhez az anyagot a lehető legkisebb egységekre kell bontani (tanulási objektumok), amelyek teljesítik például a SCORM szabványt. A keretrendszerbe nem illeszkedő tananyagok esetén ennek természetesen nem kell teljesülnie, a modularitásra kisebb mértékben azonban törekedhetünk. Az elemi részekre bontás célja az újrafelhasználhatóság. Ez hasznos lehet például fogalmak magyarázata esetén, ahol megoldható – a Sulinet Digitális Tudásbázis tananyagaiban is alkalmazott módon –, hogy az egérkurzort egy adott fogalom neve fölé húzva megjelenjen a definíció

szövege. Itt az egyszer elkészített egységet (a szöveges magyarázatot) minden alkalommal felhasználjuk, ahol a fogalom neve szerepel a tananyagban.

Rendszerszemlélet

A rendszerszemlélet a tanítás és tanulás egészére való figyelemfókuszálás [8]. Ez szintén elsősorban az oktatási keretrendszerekhez kapcsolódó fogalom, de a modularitáshoz hasonlóan ennek gondolataiból is átvehetünk néhányat. A rendszerszemlélettel gondolkodás egyrészt azt a törekvést jelenti, hogy a tananyag egyes részei (képek, szövegek, médiaelemek) megfelelően illeszkedjenek egymáshoz, és a kijelölt oktatási/fejlesztési célhoz. Másrészt ilyen jellegű gondolkodást igényel annak megvizsgálása is, hogy a tananyag hogyan illeszkedik bele a tanulási környezetbe, illetve a hagyományos oktatási rendszerbe.

Látható tehát, hogy az e fejezetben kifejtett hármas (didaktikai tervezés, modularitás, rendszerszemlélet) szem előtt tartásával és alapos átgondolásával minden szempontból megfelelő elektronikus tananyagok készíthetők.

A tervezés folyamata és nehézségei

A tananyagkészítés során az elsődleges célom az volt, hogy nagyon hamar – akár már a harmadik lecke után használható, élvezetes játékot vagy animációt lehessen készíteni a megszerzett ismeretek alapján. Ennek hatására a gyerekek talán nem veszítik el az érdeklődésüket a tananyag későbbi részei iránt, és a sikerélmény új lendületet adhat nekik a további – kissé nehezebb, összetettebb – leckék anyagának elsajátításához.

A sikerélmény egyébként is a Scratch népszerűségének egyik titka. Néhány parancs "behúzása" után azonnal látszik az eredmény, a készítéssel egy időben futtatható is a program. Kisebb gyerekek, vagy gyakorlatlanabb felhasználók számára már ez is pozitív élményeket adhat. S mivel az elkészült program néhány kattintással, könnyedén publikálható az amerikai Scratch portálon, akár több ezren megnézhetik, kipróbálhatják, letölthetik és véleményezhetik. A portál főoldalán pedig különböző szempontok (például a közösség által legkedveltebb, vagy a legtöbbször átdolgozott) szerint ajánlják a felhasználók figyelmébe a szempontonkénti három-három aktuálisan legjobb projektet. A másokkal való megosztás hatására egyszerűbb projekt esetén is erősebb a

készítőben az "alkotás-élmény". Továbbá a versenyszellem, a legjobb háromba kerülés esélye is kreatív, minőségi munkára ösztönzi.

A tananyag szerkezetének tervezésekor az is fontos szempont volt, hogy a teljes anyag rövid idő alatt átolvasható legyen. Félő, hogy nagyobb számú lecke esetén a mennyiség már ránézésre elveszi a gyerekek kedvét a végigolvasástól. Ennek okán a tananyag gerincét mindössze 8 lecke alkotja. Ezek az egységek a lehető legtömörebben közlik az új ismeretanyagot. A hosszabb kifejtést igénylő, vagy több helyen visszatérő témákról külön kiegészítő leckék szólnak.

A tananyag így lineáris szerkezetű lett, egyik lecke után a másikra lép, de egy leckén belül mégsem vezeti teljesen "kézen fogva" a tanulót. A kiegészítő leckék is oldják ezt a "szigorúságot", a gyerek csak akkor olvassa el őket, ha akarja, ezek esetében nincs kötelező haladási irány. A fogalmak és beágyazott parancsok rövid magyarázatainak felbukkanó buborékos megjelenítése szintén ezt a célt szolgálja. Ezek az elemek nem nélkülözhetetlenek a tananyag megértéséhez, de az eltérő előzetes ismeretek miatt egyes gyerekeknek szüksége lehet rájuk. Ezek csak akkor jelennek meg, ha a tanuló az őket jelző szó fölé húzza az egérkurzort.

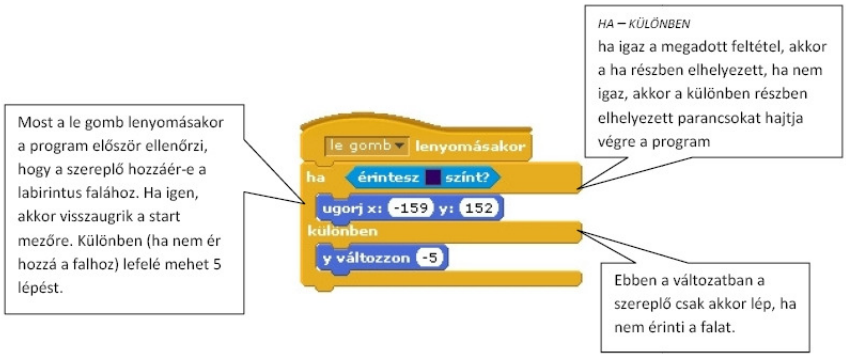
A leckék külső megjelenítésének tervezése is alapos átgondolást igényelt. Arra törekedtem, hogy minél kevésbé legyen száraz, tankönyvszerű a megjelenése, mivel az ijesztően hathat a gyerekekre. Olyan tananyag készítése volt a cél, amely „hívogatja”, bátorítja a gyereket ahelyett, hogy elijesztené. Éppen ezért a képregényszerű megjelenés mellett döntöttem. Ez alatt természetesen nem a klasszikus, sötét képregények képi világát értem, hanem azoknak csak bizonyos jellemző elemeit: sok kép, ezekhez szorosan kötődő, tömör, lényegre törő szöveg.

Tehát kevés szöveget használtam, az anyagban inkább a képeken van a hangsúly, amelyek gazdag színeikkel már első pillantásra magukra vonzzák a tekintetet. Csak a tananyag megértését és a projekt elkészítését elősegítő képeket illesztettem a leckékbe, dekorációként alkalmazott, fölösleges illusztrációkat nem.

A leckék színeinek kiválasztása két irányból is meghatározott volt. Egyrészt a képek döntő többsége Scratch parancsokból álló blokkokat ábrázol, tehát a Scratch parancsok színeit használja. Másrészt a tananyagnak otthont adó magyar Scratch portál is kialakult színvilággal rendelkezik, amely körülveszi a leckéket. A fehér alapon alkalmazott élénk narancssárga, zöld, és kék színeknek igen kellemes, gyerekkönyveket idéző az

összhatása. Nekem már csak arra kellett figyelnem, hogy a kevés, nem parancsokat tartalmazó ábra és a leckékhez készült projektek színei olyanok legyenek, amelyek illeszkednek a már adott környezetbe. Ez többnyire sikerült, így a leckék látványvilága egységes; a leckék a portállal összeillő egészet alkotnak.

A folyószöveg tömörsége miatt más magyarázó módszert kellett találni az egyes képelemekhez. Itt mutatkozik meg leginkább a képregényekhez való hasonlatosság. A képek kérdéses, vagy nagyobb figyelmet érdemlő részeire szövegbuborékokban elhelyezett rövid leírások adják meg a magyarázatot, vagy hívják fel a figyelmet. Erre mutat egy példát a következő ábra:

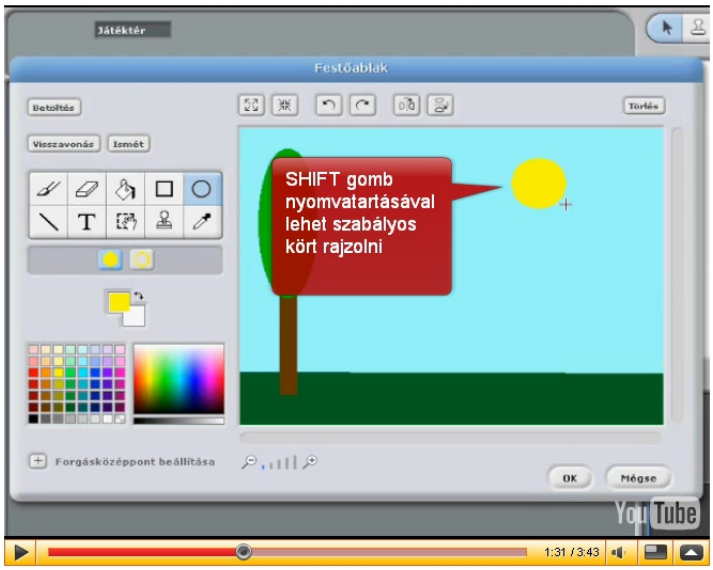


2. ábra

A szövegbuborékok könnyen olvashatók, vezetik a szemet, azonban éppen az olvashatóság szem előtt tartása végett a szöveg mennyiségét néha korlátoznom kellett. Az olvashatóság egyébként is fontos szempont egy tananyag esetében – különösen akkor, ha az gyerekek számára készült. Ki kell választani a megfelelő betűtípust, betűfokozatot, sorközt. Köztudott alapszabály, hogy a képernyőre szánt szöveg ne tartalmazzon talpas betűket, mivel azok csak a papírról való olvasást segítik, a monitoron megjelenő szöveg olvasását akadályozzák. A tananyag esetén természetesen alkalmaztam ezt az elvet. A portálon megjelenő szöveg valóban talp nélküli betűtípussal íródott, a letölthető (és nyomtatásra szánt) PDF fájlokban pedig talpas betűtípust használtam. Azonban a buborékok szövege a nyomtatható változatban is talp nélküli betűtípussal szerepel. Ennek az oka, hogy a képregényes stílushoz ez illeszkedik leginkább, és itt egyébként is csak néhány soros folyószövegből álló magyarázatokat kell elolvasni, amely még nem megterhelő.

Esett már szó a leckékbe illesztett képekről. Ezeken kívül azonban más szemléltetési eszköz használatára is van lehetőség a tananyagban. Nevezetesen videót lehet illeszteni

az egyes leckékhez. Ilyen videóból többet is tartalmaz a tananyag. Ezek egyszerű, a CamStudio szoftverrel rögzített képernyővideók, amelyek a környezet egy funkciójának használatát vagy egy projekt elkészítésének folyamatát mutatják be. Hallható magyarázat nem tartozik hozzájuk, a videók zenei aláfestéssel rendelkeznek. A szöveges magyarázatok a leckékhez hasonló módon itt is szövegbuborékokban jelennek meg.



3. ábra

A videók feltöltése a You Tube videómegosztó portálra történt, melyen lehetőség van a feltöltött anyagok feliratozására szövegdobozok és különböző alakú szövegbuborékok segítségével. A videók tehát itt találhatók, a Scratch portálon csak beágyazva szerepelnek.

A kizárólag képeket és feliratokat tartalmazó oktatási anyagokkal (prezentáció, animáció, videó) nehezebb a tanulás, mint azokkal, amelyek hangokat is tartalmaznak. Ennek oka, hogy a képek és feliratok ugyanazt az érzékelési csatornát terhelik. Tehát szerencsésebb megoldás lett volna, ha a videók képei mellett hangalámondásos magyarázat is szerepel. Azért választottam mégis ezt az utat, mert a legtöbb iskolában nincs hangszóró csatlakoztatva a számítógépekhez. Egy 2005-ös kutatás tanulságai szerint az általános iskolás gyerekek közül sokaknak – több mint kétharmaduknak – nincs otthon saját számítógépe. Ez az azóta eltelt négy évben számottevően nem változott, „*mivel a számítógépek elterjedése lassú, és a társadalmi rétegződéssel szoros összefüggésben lejátszódó folyamat*” [14]. Tehát a tananyag önálló elsajátítása is az iskolai gépteremben történhet, így a magyarázat nem lenne hallható. A buborékokban

megjelenő szöveget azonban így is el lehet olvasni, tehát a gyerekek mindenképpen megkapják a szükséges információkat esetünkben például a festőablak használatával kapcsolatban.

A leckék felépítése

A tananyag egységességének kialakításakor fontos volt, hogy minden lecke azonos szerkezetű legyen. Ez a szerkezet a következő:

- bevezetés – miről szól a lecke (esetleg milyen előzetes ismeretekre épít és milyen új ismereteket tartalmaz)
- a leckéhez tartozó játék/animáció bemutatása
- az új fogalom magyarázata (ha van ilyen)
- a játék/animáció elkészítése
- a projekt elmentése

Ezek az elemek mind fontos szerepet töltenek be az egyes leckékben. A bevezetés annak ellenére, hogy csak egy-két mondatból áll, igen lényeges része a leckének: segíti a ráhangolódást, előrevetíti, hogy miről fog szólni. Az azt következő beágyazott, webböngészőben működő projekt is hasonló célt szolgál. Bemutatja, hogy a lecke végigkövetésével milyen produktum kapható. Ha ez megnyeri a gyerek tetszését, felkelti az érdeklődését, akkor nagyobb kedvvel kezdi olvasni a leckét. Ellenkező esetben az is előfordulhat, hogy a számára nem szimpatikus játék elkészítésére nem kíváncsi, el sem olvassa a hozzá tartozó leírást. Ez azért jelenthet problémát, mert a leckék egymásra épülnek és a tömörségük miatt az egyszer elmagyarázott fogalmak más leckékben ismertként szerepelnek, ott már nem tartozik hozzájuk külön leírás. A leckék „átugrásának” elkerülésére éppen ezért törekedtem arra, hogy minél érdekesebb projektek tartozzanak az egyes leckékhez. Természetesen nem lehet minden gyerek ízlésének megfelelni, de igyekeztem a korosztályhoz közel álló képeket, háttereket felhasználni és „divatos” játékokat készíteni. Ehhez sokat kipróbáltam az amerikai Scratch portálra feltöltött projektek közül. Bár az egyes projektek készítőinek életkorát nem tüntetik fel, a projektek népszerűségéből lehet következtetni a 10-14 évesek ízlésére, ugyanis a portált ez a korosztály látogatja a legnagyobb számban.

A következő pont a játék elkészítése. Ez előtt bizonyos leckéknél előfordulhat egy magyarázó rész bonyolultabb, nehezebb vagy csak definiálást igénylő témák esetén (pl.

üzenetküldés, változók). A játék elkészítésének leírása a felhasznált hátterek, szereplők, változók, listák felsorolásával és jellemzőik, feladataik leírásával kezdődik. Ez azért fontos, mert a gyerek így már a lecke elején tudja, hogy mi a lecke célja és esetleg azon is elkezd gondolkozni, hogy ez hogyan valósítható meg, mennyit tud majd felhasználni az eddig megismert parancsokból, és hol lesz szüksége az újakra. Az új parancsok rendszerint egy feladat (parancsokból álló programblokk) részeként jelennek meg a leckében. Egy parancs első előfordulásánál szövegbuborékban elolvasható, hogy mire használható és általában egy másikban az, hogy ebben a konkrét esetben hogyan viselkedik, miért éppen ezt kellett beépíteni a feladatba. Esetenként a leckében egy problémára több megoldási módszer is olvasható. Tehát a projektek elkészítéséhez a leckék nem adnak kötelező leírást. Sok esetben van a tanulónak választási lehetősége különböző alternatívák között. Ezzel csökken a kényszerítés-érzet: a gyerek nem érzi úgy, hogy a tananyag megköti a kezeit. Ehelyett megadja a kellő szabadságot új vagy más megoldások kipróbálásához és használatához.

A leckék záró mondata – az első lecke kivételével (mivel ott nincs elkészült projekt) – minden esetben az alkotás mentésére szólítja fel az olvasót. Ennek külön jelentősége van, hiszen így nem vész el a befektetett energia, a gyerek később is előveheti a projektet, változtathatja, kísérletezhet vele, megmutathatja a családjának, barátainak. Ez azért került oda, mivel a Scratch csak új projekt indításakor kérdezi meg, hogy az előzőt elmentjük-e, a programból való kilépéskor nem. Így amikor a gyerek lecke végeztével bezárja a Scratch ablakát, könnyen megfeledkezhet az általa előállított projekt mentéséről.

A leckékhez általában tartozik egy alternatíva is, egy úgynevezett variáció. Ez olyan projekt, amely a leckében megismert elemekből épül fel. A variációkhoz nem tartozik külön magyarázó szöveg, csak néhány megjegyzésdoboz, így a gyereknek a saját tudására támaszkodva kell összeállítania. Ezen projektek célja egyrészt a gyakoroltatás, rögzítés. Más részről pedig hogy ne engedje, hogy a tanuló egy-egy leckét egyetlen projekthez kössön, ne gondolja azt, hogy csak a példaprojekthez hasonló játékot lehet készíteni a megismert parancsok, módszerek felhasználásával. Látnia kell azt, hogy bátran szabadjára engedheti a fantáziáját.

A tananyag tartalma

A következőkben bemutatásra kerülnek az egyes leckék külön-külön. Mivel a formai szempontokat már tárgyaltam, most csak a tartalmukat vizsgálom. Közben bemutatom a Scratch környezet egyes fontosabb elemeit, valamint azt, hogy mit érdemes tudni róluk.

1. lecke – Ismerkedés a Scratch környezettel

Az első lecke mind felépítésében, mind tartalmában különbözik kissé a többitől. A Scratch kezelésének kezdeti lépéseit mutatja be, tehát egyfajta alapozó jelleggel bír. Először magának a programablaknak az alapfogalmait mutatja be, mivel azok olyan fontosak, hogy a későbbiekben szinte minden leckében előkerülnek (játéktér, parancslista, programozási tér, stb.).

Ezt követően a játéktér és a szereplők bemutatása következik. Ezek szintén nagyon fontos elemek. A Scratch projektek szereplők nélkül szinte elképzelhetetlenek. A projektkészítés első lépései között szokott szerepelni a szereplők kiválasztása vagy megrajzolása. A rajzolásukról ez a lecke nem sok szót ejt, ezzel a témával a Festőablakról szóló kiegészítő lecke foglalkozik. A szereplők jelmezei szintén fontos szerepet töltenek be az animációkban, játékokban. Ezek váltogatásával érhető el például egy szereplő mozgásának animálása. A jelmezek tulajdonképpen a szereplő mozgásainak, állapotainak pillanatfelvételei. A leckében ezért bemutatásra kerül a jelmezek betöltése és használata.

A játéktér speciális szereplőként fogható fel. Az ő jelmezei is fontos szerepet töltenek be a projektekben, azok ugyanis a hátterek. Ezen kívül – a speciális jellegéből adódóan – a játéktérnek szokták adni a minden szereplőt érintő feladatokat (pl. inicializálás), így ennek bemutatása sem maradhatott ki a kezdő leckéből.

2. lecke – Megmozdulnak a szereplők

A Scratch-ben történő projektkészítés leírása, tanítása valójában a második leckében kezdődik. Itt már tényleges parancsokkal is megismerkedhet a tanuló, nem csak a lecke alapozásában használatos lépésekkel. A használhatóság, látványos eredmény elérése érdekében a szereplők mozgatásának módját ez a korai lecke mutatja be. Az elkészült projekt játéknak tekinthető. Valójában nem tartalmaz semmilyen akciót, mindössze egy papagájt lehet röptetni a játéktérben, a gyerek az elkészítése után mégis úgy érezheti,

hogy egy igazi játékot készített, majdnem olyat, mint amilyenekkel a különböző internetes oldalakon, portálokon találkozhat.

A szereplők mozgatása többféleképpen megvalósítható Scratch-ben. A legegyszerűbb – és talán a legtermészetesebb is –, amikor a fel nyíl megnyomásakor a szereplő előre megy, azaz a pillanatnyi irányában halad tovább megadott lépésnyit. A balra és jobbra nyilak hatására pedig elfordul a megfelelő irányba a megadott szöggel. Természetesen mindig az adott problémától függ a mozgások megvalósítása, nem lehet azt állítani, hogy valamelyik szereplőmozgatási mód minden esetben használható. Az imént bemutatott mozgatást oldalnézet esetén olyan szereplők esetén érdemes alkalmazni, amelyek mozgása nem annyira kötött (pl. úsznak vagy repülnek), illetve felülnézetes játékokban, például ilyen nézetű autóversenyek, illetve egyes mászkálós játékok esetén.

A mozgások megvalósítása úgynevezett feladatokkal történik. A feladat egy sapkához csatolt parancsok összessége. A sapkák is parancsok, egy-egy bizonyos esemény bekövetkezésekor lefuttatják a hozzájuk csatolt többi parancsot. Ilyen esemény lehet a zöld zászló megnyomása (azaz a játék indítása), vagy egy billentyű leütése. Ennek segítségével a papagáj irányítása mindössze három feladattal megoldható. A feladatok pedig önmagukért beszélnek: „balra gomb lenyomásakor | fordulj 15 fokot”, „fel gomb lenyomásakor | menj 10 lépést”.

A lecke a Mozgás mellett a Kinézet parancscsoportot is érinti. Itt kerül alkalmazásra az előzőekben már említett „válts jelmezt” parancs is. A gyerek itt a gyakorlatban is láthatja, hogy a jelmezváltás élettel tölti meg a szereplőt. A papagáj így nem mozdulatlanul tartja a szárnyait repülés közben (ami teljesen természetellenes lenne), hanem csapkod is velük.

A projekt elkészítése egy videón is végigkövethető. Ez azért készült, hogy a gyerekek láthassák is egy projekt készítésének lépéseit. Ez főként az olyan mozzanatoknál szükséges, amelyek szöveges leírása kissé nehézkes vagy hosszú, illetve amelyeket a későbbiekben is rendszeresen kell majd használniuk a gyerekeknek. Például hogy hogyan kell a parancsokat a Parancskészletből a Programozási térbe húzni, vagy hogyan törölhető egyszerűen egy szereplő a Játéktérből.

3. lecke – Ismétlődések

A következő lecke tartalma híven követi a programozás tanításának szokásos menetét, amelyben a szekvenciális algoritmusok után a ciklusok következnek. Azonban nem csupán a szokásokhoz való igazodás volt az oka annak, hogy éppen itt kerül bemutatásra ez a téma. A tananyag tervezéséről szóló fejezetben említett egyik törekvés az volt, hogy a gyerekek minél hamarabb tudjanak mesét készíteni a Scratch segítségével. A mesekészítés azonban tulajdonképpen animáció-készítésnek felel meg, tehát az a cél, hogy a gyerekek nagyon hamar tudjanak animációkat létrehozni. A ciklusok megismerésével ez teljesül is. A gyerekek ekkor már ismerik a háttér és a szereplők kezelését (létrehozás, törlés, stb.), a jelmezek betöltését és váltását, a Mozgás parancskészlet alapvető elemeit. Ha ehhez megismerik az ismétlődések megalkotásának módját és a Kinézet parancskészlet egyes elemeit, akkor tetszőleges animációt, mesét, animált képeslapot tudnak majd készíteni.

A ciklusoknak két típusát ismerteti a lecke: a számlálós és a végtelen (mindig) ciklust. Az előbbi megadott számú alkalommal ismétli a magjában elhelyezett parancsokat, az utóbbi pedig folyamatosan, megállás nélkül. A kétféle ciklus közötti különbséget két szereplő különböző viselkedésével szemlélteti a leckéhez tartozó projekt. A Nap *mindig* süt (azaz váltogatja a jelmezeit), a Kutya viszont csak egy kis ideig sétál, majd ugat egyet és megáll. A szereplők feladatainál természetesen különböző típusú ciklusokat használunk. Az első esetben a Nap egy kis várakozás után (a „várj” parancs szintén fontos eleme lehet az animációknak, megoldható vele egyes események időzítése) jelmezt vált és ezt ismétli a végtelenségig – vagy a projekt futásának leállításáig. A másik szereplő ezzel szemben csak tízszer ismétli meg a benne elhelyezetteket, amelyek lényegi része a „menj 10 lépést” parancs. Az ilyen típusú (számlálós) ciklus után illeszthetünk további parancsokat, a ciklus végeztével azok is le fognak futni – ebben az esetben a Kutya csak azt követően ugat, hogy befejezte a sétát. Nem mondható el ugyanez a „mindig” ciklus után illesztett parancsokról. Mivel ez a ciklus végtelen sokszor hajtódik végre, soha nem lépünk ki belőle, tehát fölösleges lenne utána parancsokat illeszteni, azok nem hajtódnának végre. A Scratch szintaktikus felépítésének köszönhetően erre nincs is lehetőség, hiszen hiányzik a puzzle-szerű csatlakozópont a „mindig” parancs aljáról. A lecke igyekszik ezt a fontos különbséget

tudatosítani a gyerekekben, de ha tanári közreműködéssel tanulják a Scratch-ben való programozást, akkor érdemes ezt a tanárnak külön is kiemelnie.

Egy korábbi fejezetben már említésre került, hogy a végtelen ciklus sok programozási nyelvvel ellentétben a Scratch-ben gyakran használt eszköz. A fenti példából látszik, hogy animációk készítésére kiválóan alkalmas. Ezen kívül ennek segítségével könnyen megvalósítható az eseményvezérlés, amely a projektek nagy részében jelentős szerepet kap.

A leckéhez tartozó variáció egy animált balatoni képeslap. A gyerekek ennek hatására remélhetőleg kedvet kapnak saját animált képeslap készítéséhez. Az külön motivációt jelenthet számukra, hogy a magyar Scratch portálon létezik képeslapküldő funkció, melynek felhasználásával az elkészült Scratch projektek elektronikus képeslapként bárkinek elküldhetők. Így az animált képeslapok ténylegesen „úgy viselkedhetnek”, mint az interneten fellelhető elektronikus képeslapok, tehát nem csak nevükben lehetnek azok.

4. lecke – Feltételek

A ciklusok után logikus lépés az elágazások bevezetése. Segítségükkel tovább tökéletesíthetők az animációk, ráadásul már igazán izgalmas játékokat is lehet készíteni a felhasználásukkal. A leckében a játékkészítés kerül előtérbe – egy egyszerű labirintusos játékon keresztül mutatja be az elágazások működését. A játék előnye, hogy a háttérben elhelyezett labirintus megváltoztatásával egészen bonyolulttá is tehető, így a készítője büszke lehet rá, hogy az internetes flash-játékokhoz hasonló, különböző elrendezésű és nehézségi fokozatú labirintusos játékokat hozhat létre.

A ciklusokhoz hasonlóan az elágazásoknak is két változatát mutatja be a lecke: a „ha”, valamint a „ha… különben” parancsokat. A leckéhez készült játék lényege, hogy a labirintus falának érintése nélkül kell végigmenni a pályán. Ha a főszereplő érinti a falat, akkor visszatér a kiindulási mezőre. Ez a visszatérés úgy történik, hogy a kurzormozgató gombok megnyomásakor a szereplő lép, majd a „ha” parancsban lévő „érintesz színt” feltétel ellenőrzi, hogy hozzáért-e a falhoz (amely egyszínű sötétlila, ennek köszönhetően vizsgálható az érintkezés). Ha igen, akkor visszaküldi a start mezőre.

A „ha…különben” parancs működését egy másik szereplő feladatán keresztül mutatja be a lecke. A labirintus végén várakozó figura saját középpontja körül forog abban az esetben, ha nem érintkezik a főszereplővel, különben egy szöveget mond: „Megtaláltál!”. Ennek ekvivalens változatát is bemutatja az anyag: ha a feltételt tagadjuk és a ha és különben ágon lévő parancsokat megcseréljük, akkor ugyanazt az eredményt kapjuk. Azaz a figura a „Megtaláltál!” szöveget mondja, ha érintkezik a főszereplővel, különben pedig a saját tengelye körül forog. Tehát itt jelenik meg először igazi interakció a szereplők között.

A lecke nem csak a feltételekkel ismerteti meg a tanulót, a Scratch koordinátarendszerének felhasználásába is betekintést enged: bemutatja a szereplők koordinátáinak változtatásán alapuló mozgatását, illetve a konkrét koordinátákkal megadott helyre ugrást. Ez a parancs („ugorj”) általában az inicializálás során kap nagy szerepet, ugyanis a szereplők kezdőpozíciója ennek felhasználásával állítható be.

A projektben egy másik fontos típusú elem is felhasználásra kerül. Ez a már említett parancs az érzékelést vizsgálja, és ez az első olyan megismert elem, amelyik igazán jól szemlélteti a Scratch puzzle játékokhoz való hasonlatosságát. A „ha” parancsban a feltétel helye egy hatszög. Az ebbe illeszthető elemek (pl. az érintkezésvizsgálat vagy a tagadás) szintén hatszög alakúak. Így a gyerekek láthatják, hogy a parancsba milyen elemeket illeszthetnek feltételként. Ez más parancsoknál is hasonlóan működik: a kapszula alakú változók például csak olyan helyre illeszthetők, amely szintén kapszula formájúak.

A projekt még egy újdonságot mutat a gyerekeknek. Itt találkozhatnak először a ciklusba ágyazott elágazással. Ha eddig úgy gondolták, hogy ez a két fajta parancstípus (ciklus és elágazás) csak egymástól függetlenül használható, akkor itt jó példát láthatnak ennek ellenkezőjére.

A leckéhez készült variáció a klasszikus videojátékokra (pl. Super Mario) hajazó mászkálós játék. Itt szintén a fal érintését és a szereplők egymás közötti érintkezését kell vizsgálni. Ez az előző projekthez hasonlóan oldható meg, tehát egyáltalán nem nehéz a megvalósítása.

5. lecke – Üzenetek

Ez a lecke egy igen egyszerű projekt elkészítésén keresztül ismerteti az üzenetek működését. Az üzenetek a Scratch-ben a szereplők között kommunikációt segítik, de felfoghatók kvázi eljáráshívásként is.

A projektnek három szereplője van, két lány és egy fiú. A lányok a rájuk való kattintáskor üzenetet küldenek a fiú szereplőnek, hogy nézzen rájuk. A fiú az üzenet fogadásakor a megfelelő irányba fordul és megtesz néhány lépést. Ez nem egy izgalmas, szórakoztató játék, azonban alkalmas arra, hogy érthetően, életszerűen bemutassa a gyerekeknek az üzenetküldés és –fogadás működési mechanizmusát. Ezért az iskolaudvart ábrázoló háttérrel és a nagyobb gyerekek számára már esetleg ismerős jelenettel próbáltam elérni, hogy az izgalmas fordulatok hiányának ellenére mégis érdeklődjenek a projekt, és így az elkészítése iránt.

Az üzenetek sokféle módon felhasználhatók a Scratch-ben, többszereplős játékok esetén szinte nélkülözhetetlenek. Ezen felül a játék bizonyos pontjain végrehajtandó feladatok ütemezésére is kiválóan megfelelnek, például egy szereplő valamilyen esemény hatására küldhet „vége” üzenetet (az üzenetek tetszőlegesen elnevezhetők). Ennek hatására leállhat az óra, megjelenhetnek a képernyőn a gyűjtött pontok, megállhat a többi szereplő, stb. Éppen ezt a felhasználási módot mutatja meg ennek a leckének a második projektje. Ez valójában nem új játék, hanem az előző lecke során elkészült labirintus folytatása – illetve módosítása. Egy régebben készített játék továbbalakításának az volt az oka, hogy a gyerek ne gondolja azt, hogy ha egy projektet a lecke befejezettnek mond, akkor abba már nem lehet további szereplőket vagy parancsokat beleilleszteni. A gyerek érezze azt, hogy egy játékon mindig alakíthat még, tudásának gyarapodásával új elemekkel, más megoldási módokkal gazdagíthatja, teheti élvezetesebbé, használhatóbbá.

Ezt tudatosítja a labirintusos projekt átdolgozása is. Egyetlen üzenetküldést illesztünk bele, amelynek hatására – amikor a szereplő a labirintus végére ér – megjelenik a „Vége” felirat. A felirat megjelenítésével és eltűntetésével újabb parancsok ismerhetők meg, ezúttal a Kinézet parancscsoport elemei közül. Ezek a szereplő láthatóságát befolyásolják. A Scratch-ben a szereplők az egész játékidő alatt léteznek, ezért ha nem akarjuk látni őket, akkor a „tűnj el” parancsot kell használnunk. A felirat inicializálása ebben az esetben annyiból áll, hogy a játék indításakor váljon

láthatatlanná, mivel akkor a másik szereplő még biztosan nem ért a labirintus végére. Amikor azonban ez bekövetkezik, akkor küld egy üzenetet („Vége”). Ennek az üzenetnek az érkezésekor a felirat megjelenik. Tehát az üzenetküldés valóban felfogható egyfajta eljáráshívásként. Az eljárásoknak azonban itt nincs nevük és nincsenek paramétereik a többi programozási nyelvben megszokott értelemben, ennek ellenére eljárásokhoz hasonló programblokkok készíthetők ebben a környezetben is.

6. lecke – Rajzoljunk

A Scratch megalkotásakor felhasználták a Logo bizonyos elemeit is, például a tollat fel, tollat le és a hozzájuk kapcsolódó parancsokat (tollméret, tollszín változtatása, rajzok törlése). A hatodik lecke ezekkel ismerteti meg a gyerekeket (vagy ha már használtak Logo-t, akkor ezek használatát mutatja be Scratch-ben), de nem az oktatásban megszokott módon. A Logo-ban ugyanis a rajzok készítésén keresztül tanítják meg a ciklusokat, elágazásokat, stb. Itt azonban a gyerekek már ismerik ezeket a fogalmakat és a használatukat. Ezért az eddigiekhez hasonlóan egy komplett projekt készítését mutatja be a lecke. A projekt egy egyszerű rajzolóprogram, amelyben az egér segítségével lehet rajzokat készíteni. Ezen kívül lehetőség van a toll színének és vastagságának megváltoztatására, valamint a rajzok törlésére.

A gyerekek tehát megismerik a Toll parancskészlet elemeit és használatukat. A rajzolást úgy valósítjuk meg, hogy folyamatosan figyeljük, le van-e nyomva az egérmutató. Ha igen, akkor a tollhegy nevű, egyetlen pixelből álló szereplő a kurzor helyére ugrik és leteszi a tollát, tehát elkezd rajzolni. Ha azonban az egér nincs lenyomva, akkor felemeli a tollat, azaz nem rajzol tovább.

A lecke végén egymásba ágyazott elágazások is megjelennek. Erre azért van szükség, hogy a játéktér bal oldalán elhelyezett panelre – amelyen a színpaletta, illetve a tollméret változtatására és a törlésre szolgáló gombok vannak – ne lehessen rajzolni. Ez kétféleképpen is megoldható lenne. Az egyik esetben a panelt szereplőként kellene elhelyezni a játéktérben. A szereplőkre tollal nem lehet rajzolni, így megoldódott a probléma. A másik megoldás talán egy kicsit bonyolultabb, ám jól szemlélteti az elágazások egymásba illeszthetőségét. Ennek során egy plusz feltételre van szükségünk az előbbi elágazásban. Nevezetesen arra, hogy a tollhegy csak akkor rajzoljon, ha az x

koordinátája nagyobb, mint a panel jobb szélének ezen pozíciója, azaz ha az egérmutató nem a panel, hanem a rajzlapnak szánt terület fölött helyezkedik el.

A tollméret és tollszín változtatását az előző leckében megismert üzenetekkel lehet megoldani, mivel a gombok megnyomásának hatására nem az ő saját tolluk jellemzői változnak, hanem a tollhegyé. Üzenetet kell tehát küldeniük neki, hogy változtassa meg a tollának valamelyik jellemzőjét. Ez a lecke már nem fűz magyarázatot az üzenetküldés működéséhez, azonban itt is csak egyszerű üzenetekről van szó, amelyek használata az előző lecke alapos átolvasása és megértése után nem jelenthet problémát a gyerekek számára.

A leckéhez tartozó variáció az Imagine Logo-ra emlékeztet. Egy teknőst lehet irányítani a játéktér bal felső részében elhelyezett E, H, J, B (előre, hátra, jobbra, balra) gombokra kattintva. Mozgás közben a teknős tolla lent van, tehát rajzol.

A leckében bemutatott mindkét program tekinthető csupán alapnak is, melyet a gyerekek számtalan funkcióval egészíthetnek ki az eddigi ismereteiket felhasználva. Mindkét esetben szándékosan használtam viszonylag kevés lehetőséget, hogy a tanulók később saját fantáziájukra és tudásukra támaszkodva fejleszthessék tovább a rajzolóprogramjaikat.

7. lecke – Változók és véletlenszámok

A leckék sorában ez az első, amely igazán összetett projekt készítését mutatja be amellett, hogy olyan témákat dolgoz fel, amelyek a gyerekek számára nehezebben elsajátíthatók. Ezért két részre osztottam. Az elsőben a projektnek csak a változókat használó részeit készítjük el, a másodikban részben pedig a véletlenszámokhoz köthetőket.

A játékban egy tátogó fejjel kell véletlenszerűen mozgó színes labdákat elkapni. A változók használata itt a pontszámolásban jelenik meg. Azt számoljuk, hogy hány színes labdát kapott el a fej, illetve pontlevonással büntetjük, ha hozzáér a fekete golyóhoz. A leckében található egy rövid magyarázat a változók fogalmáról, valamint egy ábra, amely bemutatja, hogy hogyan történik egy konkrét változó (pontszám) értékének változása; mi látszik ebből a képernyőn és mi zajlik közben a háttérben.

A leckében ezután bemutatásra kerülnek a változók kezelését szolgáló parancsok. A legfontosabb a változó létrehozása. Ezt részletesen leírja a lecke, mivel természetesen

ekkor kell meghatározni, hogy egy változó egyetlen szereplőhöz (lokális), vagy az összeshez (globális) tartozik-e. A két változótípus közötti különbségről a leckében külön magyarázat szól.

Ezek után következik az inicializálás, melynek során a változó értékét nullázni kell. Ez egyértelműnek tűnhet, azonban a gyerekek nem feltétlenül gondolnak erre, fel kell hívni rá a figyelmüket. A pontszám változtatását végző parancsokat a labdák feladataiba illesztjük. Ezek elkészítése (a labdák mozgásának megvalósítása) a lecke második felében történik.

Ha a fej találkozik egy színes labdával, akkor megeszi és kap egy pontot. A labda feladata tehát, hogy a fejjel való érintkezéskor tűnjön el, a pontszámot pedig növelje eggyel (ezt megteheti, mivel a pontszám közös változó). A fekete golyó esetén is hasonlóan kellene eljárni, azonban itt az érintkezésvizsgálat tulajdonságai miatt változtatásokra van szükség. A fekete golyó levon egyet a pontszámból, abban a pillanatban, amikor érintkezik a fejjel, majd nem tűnik el, hanem folytatja útját. Az érintkezést azonban folyamatosan vizsgálja a Scratch, azaz ha a két szereplő látszólag csak egyszer ütközött, akkor is előfordulhat, hogy a pontszám csökkentése többször is végrehajtódik. Ez a színes golyók esetén azért nem fordulhatott elő, mert azok az érintkezés pillanatában láthatatlanná váltak, láthatatlan szereplőkre pedig nem működik az érintkezésvizsgálat. Ez a probléma könnyen megoldható egy rövid várakozás beiktatásával, mialatt a fekete golyó távolabb kerül a fejtől.

A lecke második fele a véletlenszámokról szól. Rájuk azért van szükség, hogy a golyók mozgása ne legyen kiszámítható. Ennek érdekében a kezdőpozíciójuk, egy lépésük hossza és egy elfordulásuk nagysága mind véletlenül generált számok. A véletlenszámok fogalmát ekkor már ismerhetik a matematikai tanulmányaikból a gyerekek, de a leckében is tartozik hozzájuk egy rövid magyarázat, amire a fiatalabbaknak esetleg szüksége lehet.

8. lecke – Listák

Az utolsó bevezető lecke talán túl is lépi a bevezető szintet. Mégis úgy gondolom, hogy van helye ebben a tananyagban, mert a lista fogalma a hétköznapi életben is ismert, még ha kissé mást is jelent ez a szó általában, mint az informatikában. Nem áll

távol a gyerekek gondolkodásától, így már akár 10-14 éves korukban is megtanulhatják a jelentését, használatát.

A lecke természetesen a lista fogalmának magyarázatával kezdődik egy mindennapi életből vett példa, a bevásárló lista segítségével. A projekt így lehetne bevásárlással kapcsolatos is, azonban nem akartam, hogy a gyerekek megragadjanak az informatikai lista és a hétköznapi lista fogalmának azonosításánál, szerettem volna, ha egy gyökeresen eltérő témájú listát használnak.

A projekt egy európai fővárosok helyét gyakoroltató program. Három listát is tartalmaz: a fővárosok neve, az x koordinátáik és az y koordinátáik mind külön listába kerültek. A listákkal való első találkozáskor soknak tűnhet ez a szám, de mivel kezelésük sokban hasonlít a változókéhoz, a gyerekek könnyen megtanulhatják a használatukat, így több lista kezelése sem jelenthet problémát számukra egy projekten belül.

A lecke a lista fogalma után a listakezelő parancsokat mutatja be. Itt a Scratch összes ilyen parancsa bemutatásra kerül. Az eddigi parancscsoportok esetében azért nem volt erre szükség, mert azok könnyen kezelhető, egyértelmű parancsok. Ezzel szemben a listakezelés már összetettebb, a hozzá kapcsolódó parancsok olyan bonyolultságúak, amelyeket részletesen ismertetni kell; jellemzően két-három paraméter is állítható egy parancson, az általában előforduló legfeljebb egy helyett.

Ezt követően a listák létrehozási módjának leírása következik. Ez a változókhoz hasonlóan megy végbe, a listaelemek beillesztése azonban kétféleképpen is történhet. Az egyik esetben „manuálisan” töltjük fel a listát, az elemeket kis szövegdobozokba írva adhatjuk a listához. A másik esetben mindez parancsok segítségével, futási idő alatt történik. A gyerekeknek az első változat valószínűleg jobban tetszik, ez a beviteli mód ismerős is lehet nekik a táblázatkezelésből (természetesen ez csak akkor igaz, ha már tanulták). A másik esetben – főleg ha hosszú listát kell beilleszteni – már ránézésre ijesztő lehet a programkód, amelyben egy olyan feladat is van, amelyik nagyon sok listába beillesztő parancsot tartalmaz.

A projektben szinte az összes eddig megismert parancs felhasználásra kerül. A játéknak három szereplője van: az irányítótorony, amely véletlenszerűen választ célt a városok közül a repülőnek; a repülő, amely akkor kapja a következő célt, ha bizonyos pontossággal eltalálta a megadott város helyét; és végül a város, amely a rosszul eltalált

célváros helyén villan fel, hogy segítsen a repülőnek. A projekt működése üzenetküldésen alapul.

Kiegészítő leckék

Festőablak

Ennek a leckének egy másik változata is megtalálható a portálon, az áttekintések között. Abban egy háttér megrajzolásán keresztül részletesen kifejtve megtalálható minden rajzeszköz használata. Ez a kiegészítés azért készült el mégis, mert ez a lehető legrövidebben jellemzi az egyes rajzeszközöket vagy funkciókat, így ha valaki csak röviden szeretne olvasni az eszközökről, vagy esetleg csak keresi valamelyiknek a leírását, akkor itt könnyebben megtalálja. A tömör bemutatáson felül egy videó is tartozik a leckéhez, amelyen egy háttér megrajzolása követhető végig.

Forgási stílusok

A szereplők forgási stílusát sok esetben át kell állítani. Ez a tananyag leckéinek többségében is így történik, ezért kapott a téma egy külön kiegészítő leckét. A forgási stílus a szereplő irányának változtathatóságát jelenti. A Scratch három esetet ismer: a teljesen körbeforduló, a csak jobbra-balra néző, illetve a soha el nem forduló stílust. Ezeket persze különböző esetekben érdemes használni. Az elsőt olyankor, amikor a szereplő kötetlenül mozog, például úszik vagy repül, illetve ha a játéktérben felülnézet szerepel. A második forgatási stílust olyan szereplőkre kell beállítani, amelyek a szélről visszapattanáskor nem állhatnak fejre. A harmadik forgási stílus a legritkább, olyan játékok esetén alkalmazható leginkább, amelyben a szereplők egyébként sem fordulnak el.

A téma fontossága miatt itt is található egy videó, amely egy szereplő mozgását mutatja be különböző forgási stílusú beállítások mellett.

Kinézet

Ez egy különleges kiegészítő lecke, ugyanis a parancskészlet egy csoportjának minden eleméről szól néhány szót. Ez fölöslegesnek tűnhet, mivel az áttekintésekben is el lehet olvasni az egyes parancsok leírását. Azonban mivel ezek a parancsok a legtöbb projektben felhasználásra kerülnek (ez nem csak a tananyaghoz készített projektekben,

hanem általában is igaz), hasznos lehet, ha a bevezető leckékkel egy oldalon megtalálható a rövid összefoglalásuk. Így a gyerekeknek nem kell a portálon barangolniuk, elég a tananyag „címlapjára” lépniük és máris megtalálják a keresett parancs leírását. Azok a gyerekek, akik csak az első néhány leckét olvassák el, mivel csak animáció készítés a céljuk, nem is találkozhatnak a kinézet csoport későbbi leckékben bemutatott parancsaival. Pedig ezek a parancsok az animáció készítésben igen jól felhasználhatók, látványos eredményt lehet velük elérni. Ennek az összefoglalásnak köszönhetően ezek a gyerekek is megismerhetik a kinézet csoport parancsait.

Az óra

Ez a kiegészítő lecke valójában a hetedik, Változók és véletlenszámok című lecke folytatása. Azért került mégis a kiegészítő leckék közé, mert témája más projektekben is felhasználható és így az előző pont érvei ebben az esetben is működnek. Azaz ha valaki nem olvassa el a változókról szóló leckét, azért még ismerkedhessen meg az óra kezelésével. Bár az ismerkedéshez valamennyire szükség van a változókra, mivel az óra is kezelhető néhány esetben a változókhoz hasonló módon.

A Scratch órája a program indításától kezdve folyamatosan számolja az időt másodpercekben és tizedmásodpercekben. Ezt a számot az óra nevű változó tárolja. amely az Érzékelés parancscsoportban található. Ez a változó felhasználható különféle feltételvizsgálatokban (pl. „ha óra > 30”), vagy egyszerűen információnyújtásra a játékosnak a gyorsaságáról. A hozzá kapcsolódó egyetlen parancs a „nullázd az órát”. Ez értelemszerűen nullára állítja az óra értékét. Általában az inicializálás során használják.

Hangok

A hangokról szóló leírás akár a leckék közé is kerülhetett volna, mivel témája és felépítése hasonlít az első, a szereplőket és háttereket bemutató leckéhez. Mivel azonban a hangok nem nélkülözhetetlen elemei a Scratch-ben készült projekteknek, ezért használatuk leírása csak kiegészítésként szerepel a tananyagban.

A hangok használata két jól elkülönített részre bontható. Az egyik esetben a jelmezekhez hasonló módon rendelhetünk hangokat az egyes szereplőkhöz. Ezek tulajdonképpen wav vagy mp3 fájlok használatát jelentik. A hangot vagy dallamot

betölthetjük a Scratch beépített hangjai közül (vagy a számítógép más mappáiból), illetve fel is vehetjük mikrofonnal a környezet saját hangrögzítőjének segítségével. A hang bekerül a szereplő saját hangjai közé. Az itt található hangokat a projekten belül parancs segítségével lehet lejátszani.

A másik esetben hangonként összerakott dallamot lehet lejátszatni. Ez leginkább a midi-nek felel meg. Ebben az esetben 128 különböző hangszer vagy hanghatás, valamint ezen felül közel 50-féle dob közül választhatjuk ki, hogy egy adott magasságú hang hogyan szólaljon meg. A hang időtartama is meghatározható, megadhatjuk, hogy hány ütemen keresztül szóljon. A tempó pedig egy külön paranccsal változtatható.

A tananyag tartalmát áttekintve látható, hogy valóban bevezetőként szolgálhat a Scratch tanításában. Bonyolult szerkezetek, igazán összetett, átláthatatlan projektek nem szerepelnek benne, mivel azzal a céllal készült, hogy a programozásban járatlanoknak megmutassa, hogy mennyire könnyen készíthetnek saját projekteket, tehát egyrészt jó alapozásként és másrészt egyfajta kedvcsinálóként szolgáljon.

A portál

A Scratch Magyarország portál 2007 tavaszán indult. Létrehozói a Scratch programozási környezet két fordítója. Turcsányiné Szabó Márta támogatta az ötletet, így az oldal az egyetem egyik szerverén kaphatott helyet, a tartalmi fejlesztést pedig a Telementorálás kurzus keretein belül kezdtük el.

Az oldalon a Scratch-csel kapcsolatos hírek (legújabb verziók, fordítások elkészülte, versenyek), a környezet használatát segítő áttekintések, projekteket tartalmazó galériák és a telementorálást segítő fórum, illetve kommentelési lehetőségek mellett most már az új bevezető tananyag is megtalálható, a főoldal Bevezető leckék menüpontja alatt.

A tananyag kezdőoldalán két nézet választható. Az ikon nézetben csak a leckék címe és egy képkocka jelenik meg az azokban elkészíthető projektekből, vagy ha ilyen nincs, akkor egy témához kapcsolódó kedvcsináló kép. A leckék sorszáma mellett apró ikonok teszik lehetővé a kapcsolódó letöltéseket: a lecke nyomtatható verzióját PDF formátumban, a példaprojekteket, valamint azok szereplőit és háttereit. Ugyanez megtalálható az úgynevezett tematikus nézetben is, melyben ezen felül a leckék tartalmához kapcsolódó plusz információkat is kaphatunk. Ez lehet rövid leírás a lecke tartalmáról, vagy néhány kép az új ismeretet jelentő legfontosabb parancsokról.

4. ábra

Az ikonokra kattintva érhetők el a leckék. Ezek nemcsak tartalmukban és struktúrájukban, hanem a megjelenítésükben is teljesen egységesek. Egy lecke címsora alatt mindig megtalálható a navigációs sor. Ezekre azért van szükség, mert egy lecke több oldalra tagolódik. Ezen oldalak címe szerepel itt link formájában. Ezt követi a letölthető anyagok sora. Itt letölthetők ugyanazon kapcsolódó anyagok, melyek a bevezető leckék főoldaláról is. Ám amíg ott csak ikon formájában jelentek meg – az egeret föléjük húzva látható csak az ikonhoz tartozó tartalom címe –, addig itt az ikonok mellett jól olvashatóan látszik is a címük.

Ezután következik a lecke maga. A szövegében előfordulhatnak olyan szavak, amelyekhez külön magyarázat tartozik. Ezek a már korábban említett felbukkanó buborékokban jelennek meg. Ezt a funkciót külön kérésemre a tananyag kedvéért fejlesztették. A buborékok létrehozása és kezelése igen egyszerű, egy úgynevezett buborékszerkesztőben megadhatók olyan fogalmak, amelyekhez magyarázatot szeretnénk fűzni. Így csak egyetlen helyen, egyetlen egyszer kell megírni a magyarázat szövegét, amelyet akármelyik leckében, tetszőleges számú alkalommal felhasználható. Így az ilyen fogalmakhoz tartozó magyarázatoknak teljesíteniük kell az újrafelhasználhatóság elvét, azaz a meghatározásnak önállóan, a környezetétől függetlenül is értelmesnek kell lennie.

A buborékok nemcsak fogalmak meghatározását szolgálják. Az oldalon található egy összefoglaló gyűjtemény, amely a Scratch programozási környezet minden egyes

parancsát magába foglalja és mindegyikről rövid leírást ad. Ezt is a buborékszerkesztővel valósították meg. Így ha egy parancs képét szúrjuk egy leckébe, akkor az egérkurzort fölé húzva megjelenik a parancs magyarázata, ugyanúgy, mint a fogalmak esetén.

A szöveget követően található a lecke következő oldalára, vagy – utolsó oldal esetén – a következő leckére vezető link. Ez utóbbi alatt egy kis ikon mutatja az ezután elkészíthető projektet, mintegy kedvcsinálóként.

A lecke alatt hozzászólások írására (és természetesen olvasására) van lehetőség. Gyakorlatilag minden lecke egy külön fórumtopik is egyben. A felhasználók és a mentorok itt beszélhetik meg az aktuális leckével kapcsolatos kérdéseiket, észrevételeiket, megállapításaikat. Ez egyébként nem csak a leckékre igaz, a hírek kommentálása is hasonlóképpen lehetséges. Ezen kívül külön fórum áll rendelkezésre az általánosabb, vagy a leckékhez, hírekhez nem kapcsolódó problémák megvitatására.

Az oldalon lehet regisztrálni. Hozzászólást csak regisztrált felhasználók írhatnak. Az oldal készítői a regisztrációt követően különböző adminisztrátori jogosultságokat adhatnak a felhasználóknak. Ezek közé tartozik például a rejtett tartalmak elérése, a hírek, leckék szerkesztése, a buborékszerkesztő használata, valamint a fórumhozzászólások moderálása. Mivel magam is rendelkezem adminisztrátori jogokkal, nem kell minden alkalommal a fejlesztőkre várnom, ha változtatni szeretnék valamely leckén, vagy új képeket, letölthető anyagokat rendelnék hozzájuk. Ez nagyban megkönnyítette a tananyagfejlesztést.

A portál tehát minden szempontból megfelel a tananyag befogadására. Az oldal megjelenése színes, a honlap struktúrája logikusan kialakított, jól átlátható, tehát egy általános iskolás korú gyerek könnyen eligazodhat rajta. Ezen kívül a leckék oldalának látványvilága, illetve a szövegbe ágyazott segítségek (linkek, felbukkanó buborékok) is olyan megvalósításúak, amely közel áll a gyerekek által ismert modern megjelenési formákhoz.

Módszertani útmutató

Előzetes ismeretek, célzott korosztály

A leckék elsősorban felső tagozatos általános iskolások számára készültek, akikről feltételezzük, hogy rendelkeznek bizonyos előismeretekkel az informatika terén. Ezek nagyrészt megegyeznek a korábban bemutatott, NAT által támasztott követelményeknek.

A tananyag megértéséhez és elsajátításához szükséges készségek, ismeretek:

- értő olvasás – a leckék képek és videók mellett viszonylag sok szöveges leírást, magyarázatot is tartalmaznak, így elengedhetetlen, hogy az ezeket olvasó gyerek rendelkezzen az értő olvasás képességével. Tanári vagy szülői segítséggel a kevésbé jól olvasó gyerekek is elsajátíthatják a tananyagban leírtakat, de teljesen önálló tanuláshoz a szövegértés nélkülözhetetlen. A leckék szövegének hosszát a korosztály olvasási sajátosságai mellett az is meghatározta, hogy napjainkban sajnos egyre kevesebb gyerek rendelkezik az értő olvasás képességével.
- informatikai eszközök használata – mivel a tananyag nem papíron, hanem a számítógép monitorán jelenik meg egy böngésző ablakában, a gyereknek magabiztosan kell tudnia kezelni a hardver- és szoftvereszközöket. Meg kell nyitnia egy ablakot, kezelnie kell a görgetősávot, el kell indítania a Scratch programot és projekteket kell készítenie benne a leírás alapján. Ezekhez a műveletekhez pedig használnia kell az egeret, valamint a billentyűzetet. Szükség lehet a nyomtató használatára is, hiszen a leckék PDF formátumban is letölthetők, és kinyomtathatók.
- operációs rendszer kezelése – a projektek mentése és megnyitása miatt a gyereknek tudnia kell lépkedni a mentés/megnyitás ablak fastruktúrájában. A program indításához tájékozódnia kell az asztalon vagy a Start menüben (Windows operációs rendszer esetén), a használatához pedig ismernie kell az ablakok részeit.
- internethasználat – az informatikai eszközök használata pontban említett böngészőt használnia is kell a gyereknek ahhoz, hogy elérje a tananyagot.

Be kell írnia a webcímet a címsávba, navigálnia kell a különböző linkek között. Ismernie kell a fórum fogalmát és tudnia kell használni, valamint keresni az oldalakon, ha több információra van szüksége a tananyaggal kapcsolatban.

Ezen ismeretek birtokában a gyerekek akár önállóan is sikeresen elsajátíthatják a tananyagot.

A tananyag felhasználása az iskolai oktatásban

A Scratch tanítását és a tananyag tanórán való felhasználását (vagy felhasználhatóságát) több különböző tényező befolyásolja. Milyen érdeklődésű és képességű gyerekek járnak az osztályba? Hány órát szán a tanár a program használatára? Az algoritmusok és adatok témakört a Scratch segítségével szeretné tanítani, vagy marad valamelyik Logo változat mellett és csak „érdekességképpen" mutatja be a gyerekeknek a másik programot? A kérdésekre adható válaszok alapján három eset képzelhető el:

- A tanár az algoritmusok és adatok témakört teljes egészében a Scratch használatával tanítja meg.
- A témakört más programozási környezetben tanítja, azonban megmutatja a Scratch programot is néhány órában.
- A Scratch-et szakköri foglalkozáson mutatja meg az érdeklődő diákoknak.

A következőkben mindhárom változat kifejtésre kerül.

Az algoritmusok és adatok témakör tanítása a Scratch programozási környezet használatával

A magyarországi informatikatanárok döntő többsége Logo környezetet használ e témakör tanítására, mivel a Kerettanterv is „Logo-algoritmusok" ismeretét várja el az általános iskolásoktól. Ez érthető is, hiszen a Logo kifejezetten alkalmas arra, hogy megszerettesse a gyerekekkel a programozást. „*A Logo egy olyan pedagógiai környezet, »mikrovilág« amelyben a gyermekek maguk tehetnek felfedezéseket, miközben szinte észrevétlenül, minden kényszer és »bemagolás« nélkül számos új ismeretet sajátítanak el*" [16]. Eddig ráadásul nem is létezett igazán jó, magyar nyelven is működő alternatívája.

A Scratch ismertebbé válásával azonban elképzelhető, hogy egyes tanárok a Logo helyett ezt a környezetet használnák a témakör tanítására. A Scratch-et is gyerekek számára fejlesztették és a Logo-hoz hasonlóan ebben a környezetben is megvalósítható a programozás játékos, gyerekbarát oktatása. De meg lehet-e tanítani ugyanazt Scratch-ben, mint Logo-ban? Úgy gondolom, hogy igen. Ugyanis általános iskolában nem „A Logo-t", vagy „A Scratch-et" kell tanítani, hanem egy bizonyos szemlélet, egy gondolkodási stílus kialakítására kell törekedni. Ezt pedig mindkét környezetben megtehetjük – más módszerekkel, de ugyanolyan jól.

A Logo tanítását általában teknőcgrafikával kezdik. A Scratch-ben is van lehetőség ilyen típusú megközelítésre, azonban úgy gondolom, hogy érdemes más utat választani. Ha egy tanár ragaszkodik az eddig megszokott felépítéshez, akkor jobb, ha a Logo mellett marad, mivel az alkalmasabb rá. Amennyiben hajlandó másfajta megközelítést választani a programozás tanításához, akkor a Scratch megfelelő eszköz lehet számára.

A Scratch használata két részre osztható: mesekészítésre és játékkészítésre. Ezek különböző gyerekek számára lehetnek élvezetesek. A játékkészítés túl összetett fiatalabbaknak, ezért azt inkább a 7-8. évfolyamba járóknak érdemes tanítani. A kisebbek (5-6. évfolyam) – vagy a programozási tapasztalattal nem rendelkezők – érdeklődését könnyebb felkelteni egyszerűen elkészíthető, de látványos programokkal, amelyek inkább animált meséknek tekinthetők.

A mesekészítés tehát animáció-készítésnek felel meg. A gyerekek így egyszerre ismerkedhetnek a programozással és a multimédiával. Közben az egyszerű szekvenciális vezérléstől eljuthatnak a ciklusok használatáig. Megtanulnak használni egy új rajzolóprogramot – a Scratch beépített festőablakát – az eddig általában megismert Paint után, valamint hang illesztésének módját az elkészített animációjukhoz.

A következő táblázat egy ajánlott tanmenetet tartalmaz 5-6. osztályosoknak az Algoritmusok és adatok témakör tanításához.

óra	óra címe	óra anyaga	követelmény
1.	Ismerkedés a Scratch környezettel	A program kezelésének és legfontosabb jellemzőinek bemutatása.	A gyerek tudja megnyitni és kezelni a programot. Ismerje a hozzá kapcsolódó fogalmakat, elnevezéseket (szereplő, játéktér… stb.) Tudjon új projektet létrehozni; szereplőt betölteni, rajzolni, törölni; hátteret beilleszteni. Tudja elmenteni a munkáját.
2.	A szereplők mozgatása	Egyszerű mozgások és mozgatások megvalósítása. Szereplők jelmezeinek beállítása.	Tudjon szereplőket mozgatni, többféle mozgatást megvalósítani. Értse meg a jelmezváltás folyamatát.
3.	Ismétlődések	A ciklusok működésének bemutatása. Mesekészítés számlálós és „mindig” ciklus használatával	Értse meg a ciklusok működését és tudja őket alkalmazni. Tudjon több szereplős projektet készíteni.
4-5.	Mesedíszítés	Párbeszédet és hangot tartalmazó mesék készítése.	Ismerje és tudja használni a kinézet parancscsoport parancsait. Tudjon hangokat illeszteni a projektbe. Tudjon mesét készíteni az eddig megismert eszközök felhasználásával.

1. táblázat

A játékkészítés már összetettebb folyamat. Ehhez szükség van arra, hogy a gyerek magabiztosan tudja kezelni a programot, helyesen tudja használni az eddig megismert parancsokat és szerkezeteket. A játékokban gyakran sok szereplő van, ők párhuzamosan hajtják végre feladataikat. A gyereknek tehát képesnek kell lenni arra, hogy értse és

tudja megtervezni ezt a párhuzamos működést. Tudja elképzelni azt, hogy a szereplők hogyan hatnak egymásra. Ehhez szükség van arra, hogy képes legyen algoritmusokat értelmezni és alkotni, valamint az algoritmusokat megvalósítani.

Egy ajánlott tanmenet 7-8. osztályosoknak az Algoritmusok és adatok témakörhöz.

óra	**óra címe**	**óra anyaga**	**követelmény**
1.	Algoritmuskészítés	Hétköznapi algoritmusok. Algoritmusok elemzése. Egyszerű algoritmusok készítése.	A gyerek tudjon egyszerű algoritmust készíteni és ábrázolni, tudjon folyamatábrát értelmezni
2.	Újraismerkedés a Scratch környezettel	Az eddig tanultak ismétlése. Projekt készítése az eddig tanultak felhasználásával.	Tudja alkalmazni az eddig tanultakat.
3.	Elágazások	Elágazások működésének bemutatása. Elágazást használó projekt készítése.	Értse meg az elágazások működését és tudja használni őket. Legyen képes az új ismereteket beilleszteni az eddigiek közé.
4.	Üzenetküldés	Az üzenetküldés fontosságának bemutatása. Az üzenetküldés használata.	Lássa be, miért fontos az üzenetküldés. Tudja használni az üzenetküldés adta lehetőségeket.
5.	A toll parancskészlet	A toll parancskészlet elemeinek bemutatása. „Teknőcgrafika”. Egyszerű rajzolóprogram készítése.	Tudja használni a parancskészlet elemeit. Tudjon egyszerű geometriai alakzatokat kirajzolni a képernyőre.

óra	óra címe	óra anyaga	követelmény
6-7.	Változók	A változó fogalmának ismertetése. Változók létrehozása, kezelése. Műveletek változókkal. Véletlenszámok.	Értse meg a változó fogalmát. Tudjon változókat létrehozni, használni és törölni. Képes legyen műveleteket végezni változókkal, valamint speciális változókat használni (pl. óra). Tudja, mi az a véletlenszám és mire érdemes használni.
8-9.	Listák	A lista fogalmának és a listakezelő parancsoknak ismertetése. Összetettebb projekt készítése listák használatával.	Értse meg a lista fogalmát. Tudjon listákat létrehozni, használni és törölni. Tudja használni a listakezelő parancsokat.

2. táblázat

Mindkét tanmenet az általam készített tananyagra épül. Az egyes órák témája általában egy-egy lecke anyagának felel meg. (De ezek természetesen csak ajánlott tanmenetek, a téma más felépítésben, szerkezetben is eredményesen tanítható.)

A témakör tanításának egyik célja az algoritmikus gondolkodás kialakítása, fejlesztése. Ennek érdekében az oktatási folyamatba be kell építeni ezen gondolkodásmód kialakításának elemeit. Ezek a következők [15]:

- Tudatos, tervező magatartás kialakítása: fontos, hogy a gondolkodási lépések tudatosak legyenek. a gondolatok a megfelelő sorrendben kövessék egymást. Ezt segítheti ábrák, képek rajzolása és a hangos gondolkodás. Időnként érdemes egy-egy feladatot, problémát (kisebbeknél többet, nagyobbaknál már kevesebb is elég lehet) részletesen megbeszélni, és logikusan felépített, jól átgondolt ábrát vagy folyamatábrát készíteni a megoldáshoz az osztállyal közösen.
- Idő az átgondolásra: fontos, hogy egy feladatot a gyerek végig tudjon gondolni, legyen ideje eltervezni a megoldást és a tervet még gondolatban ellenőrizni. Ez a pont magában foglalja a tervezés mellett a megoldást is.

Ha túl kevés a konkrétan a megoldásra szánt idő, akkor a gyerek nem tudja alaposan átgondolni a megoldás előzőleg megfogalmazott tervének gyakorlati megvalósításához szükséges lépéseket.

- Értékelés, tudatosítás: a feladat (probléma) megoldása után a tanuló idézze fel a megoldás menetét, lássa egységben a feladatot és a megoldási módszert. Fontos, hogy hibáinak következményeit is átlássa. A megszerzett tapasztalatokat ekkor tudja beépíteni majd későbbi problémák megoldásába. Az előzetesen megbeszélt, eltervezett feladatokat a megoldást követően is meg kell néha beszélni az osztállyal közösen. Ezáltal épülhet be a gyerekek „feladatmegoldó algoritmusába" a megoldott feladat utólagos átgondolásának lépése.

A Scratch programozási környezet bemutatása alternatívaként más programozási környezet mellett

Ha az osztályt jó képességű, érdeklődő diákok alkotják, akkor az Algoritmusok és adatok témakör befejezése után néhány órában másik programozási környezetet is bemutathat nekik a tanár. Például Logo után a Scratch-et. Természetesen ebben az esetben vigyázni kell arra, hogy a gyerekeket nehogy összezavarja a kétféle környezet.

Kevés órában nem lehet teljes egészében megtanítani a Scratch-ben való programozást, ilyenkor jobban kell támaszkodni a webes tananyagra.

Azonban a gyerekek ekkor már tudnak programokat készíteni, megismerkedtek az alapvető fogalmakkal és képesek alkalmazni őket – más nyelvben. A tanárnak így elsősorban az a dolga, hogy az új környezetet magát mutassa meg, valamint azokat a „különlegességeket", amelyek az előzőleg megismert programban nem, vagy máshogyan szerepeltek.

Ehhez megfelelő oktatásszervezési mód lehet a csoportmunka. Ez sok esetben különlegességet jelent a diákok számára, hiszen a tanárok egy jelentős része még mindig nem alkalmazza ezt a módot. Tehát valószínűleg aktívak lesznek az órán és szívesen is dolgoznak majd. Közben pedig fejlődik a kommunikációs és az együttműködő készségük. A következőkben egy 2-3 órából álló blokk tematikus terve szerepel.

Tantárgy: Informatika
Témakör: Algoritmusok és adatok
Téma: A Scratch programozási környezet
Fejlesztési célok:
Az algoritmikus gondolkodás fejlesztése
Együttműködő készség fejlesztése
Kommunikációs készség fejlesztése

Tartalom	**Az órai tevékenység**		**Megjegyzés**
Gondolatmenet, részfeladat Adatok, fogalmak	***Eszközei***	***Módszerei, tevékenységi formái***	
Motiváció A tanár ismerteti az órák témáját		frontális munka	
Előzetes ismeretek felelevenítése Mit tanultunk eddig? Szekvenciális vezérlés, ciklus, elágazás.		frontális munka megbeszélés	
A Scratch bemutatása A program kezelésének bemutatása. Egyszerű feladat megoldása közösen, majd egyénileg.	projektor számítógép	frontális munka megbeszélés egyéni munka	A tanár a mesekészítéshez szükséges alapvető elemeket mutatja be. Egyszerű animációs feladatot ad egyéni megoldásra
Összetett feladatok megoldása Ismerkedés a program lehetőségeivel. Összetett projektek készítése.	számítógép tananyag	csoportmunka	Csoportokat alakítanak. Minden csoport megkapja a tananyag egy leckéjét, melynek alapján elkészítik a leckéhez tartozó projektet (természetesen más leckéket is elolvashatnak segítségképpen).

Az eredmények bemutatása Ismerkedés új fogalmakkal, programozási eljárásokkal.	projektor számítógép	frontális munka előadás magyarázat	A csoportok megmutatják az osztálynak az elkészült munkákat. Elmagyarázzák a projekt készítésének leglényegesebb részleteit.
Rendszerezés Az új ismeretek rendszerezése, összefoglalása		frontális munka megbeszélés	
A megszerzett ismeretek felhasználása Komplex feladat megoldása.	számítógép	csoportmunka mozaikmódszer	Új csoportok alkotása (a tagok különböző csoportból „származnak”). A tanár olyan feladatot ad, melynek megoldásához minél többet kell felhasználni az eddig megismert elemekből.
Értékelés Az elkészült projektek bemutatása, értékelése.	projektor számítógép	frontális munka előadás vita	A csoportok bemutatják az elkészült projekteket.

3. táblázat

A csoportmunka megtervezése a tanárnak sok energiájába kerülhet. Első lépés a csoportok összeállítása. Ez történhet véletlenszerűen (pl. sorsolással), összeállíthatja a tanár valamilyen szempont szerint (pl. a tanulók tanulmányi szintje vagy az egymáshoz való viszonya alapján), de a gyerekekre is bízhatja. Esetünkben a tanulók képességei (vagy informatikai érdeklődése) szerinti csoportalakítás lehet a legelőnyösebb, hiszen a csoportoknak nem azonos nehézségi szintű projekteket kell megoldaniuk. Így az informatika iránt kevésbé érdeklődők (vagy az Algoritmusok és adatok témakört nehezebben elsajátítók) is kaphatnak olyan feladatot, amellyel meg tudnak birkózni, a programozást szeretők pedig olyat, amelynek megoldása kihívást jelenthet számukra. Ilyen típusú csapatösszeállítások esetén azonban figyelni kell arra, hogy a diákok ezt ne beskatulyázásként éljék meg.

A következő lépés az egyes csoportok feladatainak kiosztása és a megoldáshoz rendelkezésre álló időkeret meghatározása. Az időkeretet ne legyen se túl szűk – mert akkor a tanulók „kapkodni” fognak –, se túl bő, mert akkor pedig mással kezdenek foglalkozni. A feladatok meghatározásakor éppen ezért figyelni kell arra, hogy projektek elkészítéséhez szükséges idő ne térjen el nagyon egymástól. Ez úgy is megvalósítható, hogy az összetettebb projektek egyes részeit (pl. valamelyik szereplő feladatait) a pedagógus előre elkészíti és a csapatoknak ezt kell folytatniuk. Másik megoldás erre a problémára, hogy a könnyebben elkészíthető feladatot megoldó csoportok projektjeiben több (4-5) szereplő használatát várja el a tanár.

A csoportmunka felügyelete szintén nagy odafigyelést kíván meg a tanártól. Figyelemmel kell kísérnie az összes csoport munkáját. Ha valahol kérdés merül fel, azt meg kell válaszolnia szem előtt tartva azt, hogy ne a megoldást mondja el, hanem csak rávezesse a tanulókat. A csoportokban felmerülő konfliktusok megoldásában is szükség lehet a pedagógusra. Ha a konfliktus nem a feladat megoldásával kapcsolatos, hanem személyes, akkor vissza kell terelnie a csoportot az órai munkához. A feladat megoldásával kapcsolatos vitákba csak akkor kell beleavatkoznia, ha az hátráltatja a csoport munkáját.

A pedagógusok számára talán ebbe az új szerepkör való behelyezkedés a legnehezebb csoportmunka esetén. Itt nem ő az egyedüli irányító, nem tőle függ, hogy a tanulók megértik-e az anyagot. Ebben a helyzetben ő „csak” segítő. Ezt meg kell szoknia és az első (esetleg sikertelen – zűrzavarosabb) csoportmunkás órák után nem kell visszatérnie a frontális munka egyeduralmához. Ha megtanulta az „új feladatkörét” gyakrabban és bátrabban alkalmazhatja ezt az oktatásszervezési módot.

A csoportmunka befejeztével a csapatok frontális munkaformában bemutatják az elkészült projektjeiket az osztálynak. A bemutatás történhet egy gyerek által, de végezheti az egész csoport is. Az utóbbi esetben vigyázni kell arra, hogy a beszámoló érthető legyen, a csapattagok ne vágjanak egymás szavába, de ne is álljanak szótlanul arra várva, hogy valamelyik társuk elkezdje a bemutatást.

A csoportmunka értékelése szövegesen történik. Ebbe az egész osztály bevonható, közösen meg lehet beszélni a csoport teljesítményét. Az elkészült projekt értékelése mellett a csapat együttműködésének jellegére is ki kell térni. [11]

Látható, hogy a csoportmunka szervezése sok időt és energiát vesz igénybe, mégis azt gondolom, hogy a pozitív hatásai miatt időnként érdemes ezt az oktatásszervezési módot alkalmazni. „*A gyerekek megtanulnak figyelni, hallgatni egymásra, vitatkozni egymással; gyakorlatot szereznek a munkamegosztásban, az idővel való gazdálkodásban; tapasztalatot szereznek az önálló tanulás mások által alkalmazott eljárásairól.*” [11]

A Scratch programozási környezet bemutatása szakköri foglalkozáson

Informatika szakkörre többnyire az informatika iránt érdeklődő, a tárgyat jól teljesítő tanulók járnak. Ha az informatikatanár úgy dönt, hogy az osztály egészének Logo nyelven tanítja az Algoritmusok és adatok témakört és nem mutat meg nekik más környezetet, a szakkörre járó diákokkal akkor is megismertetheti a Scratch-et. Ezeket a gyerekeket valószínűleg érdekelni fogja az új programozási nyelv, és hajlandók lesznek akár önállóan is tanulni róla.

A tanárnak elég egy ízelítőt, rövid bemutatót tartania, ahol megismerteti a diákokat a Scratch használatának alapjaival, valamint megmutatnia, hogy milyen projektet készíthetők a környezetben. Ezt követően megkérheti a tanulókat, hogy önállóan olvassák el a leckéket – tanulják meg az alapokat (akár otthon, akár a szakkör keretein belül). Így később csak a kérdéses pontokat kell megbeszélniük. A továbbiakban pedig haladó szintű problémák megoldásával foglalkozhatnak – immár közösen.

Itt merül fel a tehetséggondozás kérdése. Szakköri foglalkozásokra általában érdeklődő, az adott tantárgyban sikeres gyerekek járnak. A tanulmányi versenyek résztvevői is közülük kerülnek ki (hiszen a verseny is a tehetséggondozás egy formája). Ennek következményeként nem valószínű, hogy sok informatikatanár mutatja majd meg a tananyagot a szakkörök ideje alatt. Ugyanis a programozási (vagy animáció-készítő) versenyek szinte mindegyikének felső általános iskolások számára rendezett fordulóiban a Logo nyelvet kell használni, így a tanárok természetesen ennek a nyelvnek a mélyebb, biztosabb megismertetésére törekszenek. Az ország eddigi egyetlen olyan informatikai versenye, ahová a résztvevők Scratch-ben készült programokkal is nevezhettek, a 2008-ban megrendezett Kihívás verseny volt. A környezet ekkor még szinte ismeretlen volt Magyarországon, a gyerekek csak egy internetes tananyagra és a saját felfedezőkedvükre támaszkodhattak, amikor a versenykiírást elolvasva azt választották,

hogy ebben készítik el versenyprogramjukat. Ennek ellenére tíznél több csapat küldött be Scratch projektet, amelyek szinte kivétel nélkül jól megvalósított, ötletes programok voltak.

Néhány Scratch tanulmányi verseny megrendezésével a környezet nagyobb népszerűségre tehetne szert, többen ismernék meg és használnák mind a diákok, mind az informatikatanárok közül.

Differenciálás

Ebben a módszertani fejezetben kell szólnom a differenciálásról, azaz „különbségtételről" is, mivel annak megvalósítása részben a tanár feladata. Egy 1999-es, az Oktatási Minisztérium munkatársai által folytatott kutatás eredményei azt mutatják, hogy informatikából maguk a diákok is igényt tartanak a differenciálásra. „*A legjellemzőbb véleménycsoportba tartozók a differenciált oktatás bevezetésének igényét fogalmazták meg. A tanulók igénylik, hogy az informatika terén eltérő tudással rendelkezők a tudásszintjüknek megfelelő csoportban sajátíthassák el a szaktárgyi ismereteket.*" [17]

Az egyes gyerekek ugyanis eltérő idő alatt és különböző nehézségek árán sajátíthatják el ugyanazt a tananyagot. Az is előfordulhat, hogy míg egy tanuló egy bizonyos szinten nem, vagy csak nagyon nehezen tud átlépni (például a változók működését és használatát nem érti meg), addig egy másik diák már „mindent tud", és unatkozik. Az ilyen esetekben a tanárnak olyan oktatási módot vagy tananyagot kell találnia, amely mindkét tanuló fejlődését egyaránt támogatja. „*A differenciálás célja, hogy az egyes tanulók egyéni szükségleteihez igazítsuk – a tanterv biztosította lehetőségeken belül – az elsajátítandó tananyag tartalmát és szerkezetét, valamint oktatási módszereinket.*" [6]

A Scratch-hez készült tananyag önmagában nem ad lehetőséget a differenciálásra, mivel egyetlen útvonalból áll, azon kell végigmennie a tanulóknak, nincsenek alternatív utak. Illetve a differenciálás egyetlen módja ebben az esetben a gyerek saját tempójához való igazodás. Ez a tananyag önálló tanulása esetén adott, hiszen ilyenkor a tanulónak nem kell másokhoz, vagy akár határidőkhöz igazodnia, a saját tempójában haladhat: többször átolvashat részeket, vagy éppen kihagyhatja a számára egyértelmű megoldások magyarázatát. További differenciálási módot jelenthetnek a felbukkanó buborékok.

Ezek az olyan tanulóknak adhatnak segítséget, akik nehezen tudják megjegyezni az egyes fogalmakhoz tartozó magyarázatokat, vagy a korábbi leckékben ismertetett parancsok funkcióit, vagy egyszerűen nem rendelkeznek a szükséges előismeretekkel. A buborékok csökkentik az előbbiek lemaradását, hiszen nem kell visszalapozniuk és újra végigolvasniuk a magyarázatokat, a felbukkanó szöveg segíti azok felidézését. Az utóbbiak pedig ugyanúgy be tudnak kapcsolódni a Scratch használatába, mint a kellő számú előismerettel rendelkező társaik, anélkül, hogy saját maguknak kelljen felkutatni a szükséges információkat.

Az iskolai oktatásban azonban már több lehetőség adódik a tananyag differenciálással történő feldolgozására. Itt is megjelenhet a gyerek saját tempójához való alkalmazkodás, ám ezt az egész osztállyal való munkával nehéz összeegyeztetni. A differenciált oktatás nehezen valósítható meg frontális munka alkalmazása során. A pedagógusnak ilyenkor inkább egyéni, páros, vagy csoportmunkát kell alkalmaznia oktatásszervezési módként.

Az egyéni munka során a fent kifejtetthez hasonlóan könnyen megvalósítható, hogy minden gyerek a saját tempójában haladjon. Ekkor azonban a tanárnak figyelnie kell arra, hogy minden tanuló számára (a leggyengébbtől a legtehetségesebbig) megfelelő segítséget nyújtson, illetve a szintjéhez mért feladatokat adjon. Ez a tanulók alapos ismeretét és megfelelő felkészülést követel az oktatótól. Az értékeléskor pedig arra kell törekednie, hogy az egyes diákok teljesítményét a saját szintjükhöz mérje. Ha egy gyerek nagyon nehezen sajátítja el témakört, nem látja át az algoritmusok működését, mégis összerak egy projektet – esetleg nagy erőfeszítések, hosszan tartó munka árán –, amelyben legfeljebb ciklusokat használ, akkor ne a legrosszabb jegyet kapja arra hivatkozva, hogy nem éri el az előírt szintet. A másik véglet esetén is figyelni kell erre: egy tehetséges diák percek alatt „összedobált” játéka ne érjen automatikusan jelest, a tanár serkentse a tanulót saját képességeinek kiaknázására. Érdemes ilyenkor szöveges értékelést (is) alkalmazni, amely árnyaltabb képet adhat a diákok teljesítményéről, fejlődéséről.

A páros és csoportmunka felhasználása a differenciásban hasonló. Mindkét esetben alkothatják a csoportokat vagy párokat közel azonos képességű diákok. Így az egyes csoportok kaphatnak különböző nehézségi szintű feladatokat. Például a lassabban haladó tanulók egy könnyebb lecke feldolgozását, a jobb képességűek pedig egy

összetett projekt megvalósítását. Az értékelésnél ebben az esetben szintén érdemes megfontolni az egyéni munkánál leírtakat.

A tananyag felhasználása önálló tanulás során

Az általános iskola és az önálló tanulás

Általános iskolások számára idegen tanulási forma lehet az önálló tanulás. Az iskolai oktatás során hozzászoktak ahhoz, hogy valaki mindig elmondja nekik a tananyagot, megtanítja nekik az új ismereteket. Az önálló tanulást csak önálló átismétlés/önálló memorizálás értelemben ismerik és gyakorolják. Ez érthető is, hiszen az iskolai tananyagon kívül más tudásra nem igazán van szükségük, vagy ha mégis, azt különóra vagy szakkör keretében szintén tanulják valakitől.

De mint ahogy az előző fejezetben olvasható, elképzelhető, hogy az informatikatanár nem tanítja a Scratch-et, a tanulót mégis érdekli ez a programozási környezet. Ekkor rákényszerül, hogy valóban önállóan, egy internetes tananyag segítségével sajátítsa el a program használatához szükséges tudnivalókat.

Az önálló tanulás azonban csak akkor lehet sikeres, ha a gyerek rendelkezik az ehhez szükséges képességekkel. Ezek korábban már kifejtésre kerültek; összefoglalva a következők: a gyereknek „*alkalmasnak kell lennie a tananyag elsajátítására, a tanulási program végrehajtására, azaz rendelkeznie kell az önálló tudáskonstrukció képességével*” [8]. Tehát teljesen önálló tanulás nem valósulhat meg olyan gyerekek esetében, akiknél a fenti képességek még nem alakultak ki. Nekik személyes (tanári, szülői), vagy távsegítségre (telementorálás) van szükségük.

A tanári segítségnyújtás az informatika óra keretein kívül történhet, hiszen ha a gyerek önállóan szeretné elsajátítani a tananyagot, akkor annak az lehet az oka, hogy a tanár nem tanítja az órán. Történhet a szakköri foglalkozáson (ez az eset a tanárokra vonatkozó módszertani fejezetben került kifejtésre), vagy az órák közötti szünetben. Ez azonban nem ideális időpont és időtartam a problémák megbeszélésére. Így ha a tanár nem szeretne az egyes tanulók lényegében iskolán kívüli tanulmányaira külön időt szánni, akkor a gyerek nem fog megfelelő segítséget kapni tőle. Ez pedig igen valószínű eset, ugyanis az informatika tárgy alacsony óraszámai miatt a kötelezően előírt tananyag problémáinak megbeszélésére sem jut mindig elég idő.

A személyes segítségadás másik forrása a szülő lehet. Ebben az esetben elméletileg nincs időkorlát, azonban a gyerek és a szülő közös munkáját más tényezők is gátolhatják. Például az, hogy a szülők nagy része még mindig nem tudja kezelni számítógépet, esetleg fél is azok használatától. Ezért előfordulhat, hogy el sem olvassák a tananyagot, mert „úgy sem értenek a számítógéphez”. Ilyenkor a gyerek egyáltalán nem számíthat a segítségükre.

Persze vannak olyan szülők is, akik rendszeres számítógép- és internethasználók. Viszont ők sem ismerik a Scratch programot, tehát vagy egyedül, vagy a gyerekükkel közösen nekik is végig kell olvasniuk a tananyagot ahhoz, hogy segíteni tudjanak. Erre pedig különböző okok miatt nem minden szülő akar vagy tud időt szánni, így a számítógép használó szülők gyerekei is maradhatnak segítség nélkül.

Ha a gyerek sem a tanárától, sem a szüleitől nem tud segítséget kérni, akkor még mindig fordulhat a magyar Scratch oldal készítőihez, illetve adminisztrátoraihoz. Ennek érdekében jött létre a magyar Scratch portálon a fórum. A tanulók tehetnek fel kérdéseket egy általános topikban is, de minden leckéhez tartozik egy saját topik, így a konkrét, leckéhez kapcsolódó kérdéseket feltehetik a lecke után közvetlenül. Ezekre a kérdésekre általában egy-két napon belül érkezik is válasz, mivel a Telementorálás tanegység egykori hallgatói (akik az előző tananyag szövegezését végezték) a mai napig aktívan „telementorálnak”.

Tehát minden gyerek kaphat segítséget valakitől, ha akadályba ütközik a tananyag elsajátítása során. A szerencsésebbek (mivel ők azonnal kapnak választ és a válasz kapcsán felmerülő újabb kérdéseiket is rögtön feltehetik) személyesen egy informatikatanártól, vagy esetleg a szüleiktől. A többiek pedig a magyar Scratch portál valamelyik mentorától.

A tananyag és a tanulási stílusok

A tananyag feldolgozásának módja tanulónként eltérő lehet. Mivel a gyerekek különböző tanulási stílusúak, nincs olyan ajánlott módszer, amely mindenki számára hatékony tanulást biztosít. A következőkben azt foglalom össze, hogy a tanulási stílus milyen szempontok szerint vizsgálható, valamint ez alapján megvizsgáljuk, hogy a tananyag milyen tanulási stílussal rendelkező gyerekek számára felel meg leginkább.

A tanulási stílusok különbözőségének okait először három szempont szerint vizsgáljuk Katona Nóra és Thomas Oakland kutatásai alapján [5]. Ezek: biológiai, kognitív és motivációs különbségek. A biológiai különbségek alatt esetünkben az agyfélteke-dominanciát kell érteni. Azok a gyerekek, akiknek a jobb agyféltekéje a domináns, könnyebben teljesítik a nyelvi és az analitikus gondolkodást igénylő feladatokat, míg a bal agyfélteke dominanciájú tanulók a vizuális és holisztikus látásmódot igénylő feladatokban jobbak. Ezek markáns különbségek, mégsem meghatározók, mivel a gyerekek többsége mindkét agyféltekéjét szinte ugyanolyan jól használja, tehát a biológiai különbségek ellenére ugyanolyan jól elsajátíthatják a tananyagot.

A kognitív különbségek sokkal inkább meghatározzák az egyén tanulási stílusát. Itt három szempont szerint különbözhetnek a tanulók. Érzékleti modalitás szempontjából egy gyerek lehet auditív (azaz a legkönnyebben akkor tanul meg egy anyagot, ha hallja), vizuális (aki elsősorban képek és írott szöveg alapján tanul), valamint taktilis-kinesztétikus (tapintás és mozgás útján tanul). A tanulókat be lehet sorolni e három kategória egyikébe, azonban kutatások kimutatták, hogy a tanulás akkor a leghatékonyabb, ha több érzékleti csatornát használunk egyidejűleg. A tananyag ennek a kritériumnak nem felel meg, elsősorban vizuális érzékleti modalitású gyerekek számára készült.

A következő kognitív dimenzió a mezőfüggőség-mezőfüggetlenség. A mezőfüggő tanulók inkább a részletekre összpontosítanak, nehezebben veszik észre a tágabb összefüggéseket. Ezzel szemben mezőfüggő tanulók éppen a szélesebb összefüggéseket veszik észre könnyedén és a részletek felismerése okozhat számukra problémát. Általában a természettudományos, informatikai érdeklődésű gyerekekre igaz a mezőfüggetlenség, valószínűleg a tananyagot is ők sajátítanák el könnyebben.

A harmadik kognitív dimenzió a tanunókat a tanulás menete szerint sorolja be a szukcesszív (akik apró lépésekben szeretnek előrehaladni), vagy a szimultán módon tanulók közé. Az utóbbiak azt szeretik, ha egyszerre sokféle információt kell figyelembe venniük. A tananyag e dimenzió mindkét pólusán lévő tanulóknak megfelelő lehet. Egyrészt kis lépésekben halad előre, egy-egy lecke nem tartalmaz túl sok új információt. Az utolsó három leckéhez készített projekt azonban már összetett, egyszerre több tényezőt is figyelembe kell venni az elkészítésüknél. A „tiszta” típusok egyébként is

ritkán fordulnak elő, így azt mondhatjuk, hogy ez a szempont nem határozza meg egyértelműen, hogy milyen módon tanuló gyerekek számára előnyösebb a tananyag.

A biológiai és kognitív különbségek mellett motivációs stílus szerint is vizsgálhatjuk a gyerekeket. Először a feladat nehézségének megítélése szerint. Ha egy gyerek egy feladatot túl könnyűnek, vagy irreálisan nehéznek ítél meg, akkor csökken a motivációs szintje. Az a tapasztalat, hogy akkor legmagasabb a teljesítménymotiváció, ha a gyerek közepesen nehéznek ítéli meg a feladatot (azaz a megoldás valószínűségét 0,3-0,7 közöttire becsüli). Természetesen két gyerek ugyanazt a feladatot nem feltétlenül ugyanolyan nehéznek ítéli. A tananyag készítésekor törekedtem arra, hogy a projektekhez különböző nehézségi szintű feladatok tartozzanak. Így remélhetőleg minden gyerek talál köztük (több) olyat, amelynek megoldásában motivált lesz.

A motivációs stílus külső-belső kontroll szerint is vizsgálható. A belső kontrollal rendelkező személyek viselkedésüket és a velük történteket úgy tekintik, mint ami a saját ellenőrzésük alatt áll. Úgy gondolják, hogy a környezetüket és a viselkedésüket is meg tudják változtatni (ha akarják). Az ő motivációs szintjük általában igen magas. A belső kontrolos gyerekek valószínűleg megtanulják a tananyagot. A külső kontrollos emberek éppen ellenkezőképpen gondolkoznak. Azt hiszik, hogy a viselkedésük és a környezetük teljesen független tőlük, a véletlen, a sors vagy egy külső hatalom irányítja. Nem bíznak önmagukban és azt hiszik, nem képesek változtatni a környezetük vagy saját maguk viselkedésén. Általában passzívak, félnek a kudarcoktól, ezért túl magas vagy túl alacsony szintet tűznek ki maguk elé. Az ilyen gyerekekben kevés a kezdeményezőkészség, ezért valószínűleg nem kezdenének bele önállóan a tananyag elsajátításába.

Az előzőekben bemutatott jellemzők befolyásolják tehát azt, hogy egy gyerek egyáltalán elkezd-e önállóan tanulni egy tananyagot, ha elkezdte, akkor mennyi nehézséget fog neki okozni. És e jellemzők ismeretében állíthat össze egy tananyag tervezője és írója olyan szerkezetű és tartalmú anyagot, amely minél több, különböző tanulási stílusú gyereknek megfelel.

A tanulási stílusok különbözőségei más szempontból, Jung temperamentumelmélete alapján is megfigyelhetők. Ez az elmélet három bipoláris dimenzió mentén helyezi el az embereket. Ezek a következők: introverzió – extraverzió (mi a forrása az egyén energiáinak és irányulásának), gyakorlati – képzeleti (ez gyakorlatilag a mezőfüggőség

– mezőfüggetlenség dimenziójának felel meg), gondolkodás – érzelem (ezek a döntéshozatallal kapcsolatos tulajdonságok – az egyén racionális vagy érzelmi alapon hoz-e döntést). Ezek teljes egészében lefedik az előbb tárgyalt tulajdonságokat, de azoktól eltérően mindegyikük megbízhatóan mérhető. A következő táblázat bemutatja az egyes dimenziók sarkpontjait képviselő emberek tanulási jellemzőit [5].

Az energia forrása és az energiaszerzés módja szerint	Introvertált	Extravertált
	Inkább saját belső világuk felé fordulnak. Jobban kedvelik a magányt és a kiscsoportos tevékenységet, önreflexió révén alakítják gondolataikat.	Az emberek és a külvilág számára nyitottak, mások jelenléte energetizáló számukra. gondolataikat elsősorban a másokkal való kommunikáció során alakítják, csiszolják.
A preferált ingerek szerint	Gyakorlati	Képzeleti
	A valós világ felé fordulnak, a személyes élmények kötik le őket. Jobban szeretik először gyakorlati síkon megérteni a dolgokat és az elméletet csak azután elsajátítani. A lépésről lépésre való tanulást kedvelik.	A fogalmak és elméletek érdeklik őket. Az elméleti, átfogó megértést szeretik a részletek megismerése előtt. A belátáson és megérzésen történő tanulást szeretik.
A döntéshozatal módja szerint	Gondolkodó	Érző
	Elemeznek és kritikusan gondolkoznak. Az igazság számukra fontos tényező a döntéshozatalban. Élvezik a versenyt és a vitát.	Személyes, szubjektív normákat alkalmaznak a döntéshozatalban. Élvezik az együttműködést, de a versenyt nem.

4. táblázat

A táblázat összefoglalása alapján azt mondhatjuk, hogy a tananyag elsősorban az introvertált, gyakorlati, gondolkodó gyerekek igényeinek és stílusának felel meg.

A tananyag tesztelése a gyakorlatban

A tananyag első teszteléséhez három gyerek segítségét kértem. Az anyag ajánlott korcsoportjába tartozó 12 éves fiú és 13 éves lány mellett egy 15 éves lány is részt vett a kipróbálásban. Ő ugyan már „túlkorosnak" számít az ajánlott életkorhoz képest, azonban eddig soha nem tanult programozást, így az életkorával nem került előnybe a többiekhez képest.

A gyerekeket arra kértem, hogy a portálon található tananyagot olvassák el, készítsék el a hozzájuk kapcsolódó projekteket, tehát az anyagot önálló tanulással dolgozzák fel. Természetesen több elérhetőségemet is megadtam nekik, amelyeken bármikor tehettek fel kérdéseket. Érdekes módon ezzel a lehetőséggel csak az egyikük élt, ő e-mailen keresztül keresett meg.

A tananyag elsajátítását egy feladatsorral, valamint egy kötetlen elbeszélgetéssel ellenőriztem. A teszt hat feladatot tartalmaz. Ezek három kategóriába sorolhatók. Az első „Mit csinál a szereplő"-típusú. Itt egy-egy szereplő feladatai alapján kell meghatározni, hogy az hogyan irányítható, vagy hogy mi történik vele a projekt során. Ezek a feladatok eltérő nehézségűek, a tananyag különböző területeinek ismeretére kérdeznek rá. Egy ilyen típusú feladat bizonyos szintű fogalmazási készséget is megkövetel a gyerekektől, de úgy gondolom, hogy ez ilyen korú tanulók esetében már nem jelenthet problémát.

A másik feladattípusban szövegesen szerepel, hogy mi történik egy projektben két szereplő között, majd a megadott három-három lehetséges megoldás (mindkét szereplőhöz tartozó lehetséges feladatok) közül kell a gyerekeknek kiválasztaniuk, hogy mely feladatok valósítják meg a szövegben leírtakat. Ennél a feladattípusnál különösen figyeltem arra, hogy a szövegértés esetleges nehézségei ne lehessenek a feladat megoldásának gátjai. A projektek működésének leírása legfeljebb három sorból áll.

A harmadik típusú feladat a legnehezebb, ebből csak egy került a tesztbe. Ez egy projekt szereplőinek feladatait mutatja be képekkel, illetve leírással. A gyerekeknek ez alapján kell leírniuk a projekt működését. Bár e feladat hasonlít az első típusra, mégis külön kategóriába tartozik. Míg ott csak egyetlen szereplő viselkedését kell meghatározni, itt sok szereplő interakcióját és annak mechanizmusának felismerése a cél. A feladat nehézségét tovább növeli, hogy az utolsó, listákról szóló lecke ismeretére

is szükség van a megoldásához. Eddig a leckéig nem is minden gyerek jutott el – ebben az esetben természetesen nem tudták megoldani ezt a feladatot.

A három gyerekkel az egyéni sajátosságaik miatt gyakorlatilag három különböző típusú kísérletet végeztem – annak ellenére, hogy ugyanazt a feladatot és később ugyanazt a tesztet kapták. A következőkben e három kísérlet lefolyását és tapasztalatait ismertetem.

Első kísérlet

A legfiatalabb kísérleti személy a 12 éves Gergő. Ötödik osztályos (korábbi iskolaértetlensége miatt évvesztes), jó tanuló, az informatika iránt érdeklődő gyerek. Heti egy informatika órájuk van, ahol a számítógépek felépítése témakör után most a Paint rajzolóprogrammal ismerkednek. Programozást eddig nem tanult. A kísérletben részvételre való felkérést egykedvűen fogadta. Nem lelkesedett igazán, ám ellenkezést sem fedeztem fel a viselkedésében. Kérdéseket sem tett fel a programmal kapcsolatban. Ennek azonban az is lehetett az oka, hogy csendes, visszahúzódó gyerek, aki a – szándékom szerint – bátorító viselkedésem ellenére sem mert kérdezni.

A tananyagot valóban önállóan sajátította el, sem tőlem, sem a családjától nem kért segítséget hozzá. Egy nap egy leckét olvasott el és elkészítette a hozzá kapcsolódó projektet, vagy projekteket. Azonban az utolsó két leckét túl nehéznek találta, így azok anyagát nem sikerült megtanulnia.

A teszt

A gyerek a teszt kitöltésére szervezett találkozónkon már határozott lelkesedést mutatott a Scratch iránt. A beszélgetésünk alkalmával többször említette, hogy nagyon tetszik neki a környezet, szeret projekteket készíteni a segítségével. Azt is megtudtam tőle, hogy inkább a leckékhez tartozó projektekhez hasonlóakat szokott készíteni, vagy csak azokon változtat. A tananyag és a projektek egyébként is elnyerték a tetszését, főként a színvilágúk miatt. A szövegek buborékos megjelenítése (és ezáltali rövidsége) szintén elnyerte a tetszését.

A tesztet önállóan kellett kitöltenie. Ez körülbelül 30 percet vett igénybe. A feladatok megoldása többnyire sikeres volt, bár néhány esetben félreértette a feladat szövegét. Az esetleges bizonytalanságai ellenére ebben az esetben sem tett fel kérdéseket. A teszt a

mellékletben megtalálható. A következőkben idézem a gyerek kérdésekre adott válaszait.

1. feladat

Ez a feladatsor legegyszerűbb feladata. Azért került bele a tesztbe, hogy a gyerekeknek megadja a kezdeti sikerélményt, hogy meglássák, nem nehéz a feladatsor. Egyszerűségének forrása, hogy egyrészt nagyon hasonlít az egyik tananyagban szereplő projektre, tehát ismerős lehet a gyerek számára. Másrészt pedig csak háromféle parancs szerepel benne, az a három, amelyek alapparancsoknak számíthatnak olyan szempontból, hogy a Scratch-et bemutató órákon, foglalkozásokon ezekkel szokták szemléltetni a környezet használatának egyszerűségét.

A feladatban egy „kerge" helikopter irányítását kell értelmezni. Ennek megvalósítása a második leckében ismertetett mozgatáshoz hasonló, azzal a különbséggel, hogy ez a szereplő az ottani papagájhoz képest mindent fordítva csinál. Például a jobbra gomb megnyomása esetén balra fordul. Tehát a „helyes" válasz a következő: a helikopter minden gomb megnyomásakor pontosan az ellenkezőjét csinálja annak, amit várnánk.

A gyerek válasza:„Ha lenyomjuk a fel gombot, megy mínusz öt lépést. Ha lenyomjuk a le gombot, akkor megy lefelé öt lépést. Ha lenyomjuk a bal gombot, akkor nem fordul semmit. Ha lenyomjuk a jobb gombot, akkor fordul mínusz tíz fokot."

A válasz csak részben jó. Néhány helyen egyértelműen elnézte a parancsban megadott számokat (pl. a balra gomb megnyomása esetén). Máshol pedig nehéz eldönteni, hogy a gyerek nem érti, hogy mi történik pontosan egy gomb lenyomására, vagy a feladat szövege és szóbeli ismertetése nem volt elég precíz. A megoldásban úgy tűnik, hogy Gergő csak mondatba foglalta a parancsokat ahelyett, hogy a saját szavaival leírta volna a helikopter működését.

2. feladat

Ez a feladat már összetettebb. Több parancsot használ, köztük ciklust és abba ágyazott elágazást is. Az elágazás feltétele azonban könnyíti a megoldást, hiszen az erről szóló lecke is hasonló feltételt használ. A különben ágon elhelyezett parancsok az előző feladatban alkalmazottakhoz hasonló stílusúak, szintén az első néhány lecke anyagába tartoznak.

Itt egy rendőr működését kell leírni, aki a projekt indításától kezdve folyamatosan mozog (balra-jobbra) a játéktérben. Amikor piros színt érint, akkor megáll és a „Piros!" szöveget mondja.

Gergő megoldása: „Ha a zöld zászlóra kattintunk és a rendőr a pirosat érinti, akkor azt mondja, hogy piros 3 mp-ig. Megy öt lépést, vált jelmezt, vár nulla egész egy század másodpercet és ha szélen van, nem megy ki."

Ez a megoldás szintén csak részben jó. Mert valóban az történik a projektben, amit a gyerek leírt, azonban ezt a parancsnevek mondatba foglalásával tette. Tehát nem derül ki, hogy látja-e, hogyan működnek pontosan a „mindig" és a „ha…különben" parancsok, valamint mi a jelentőségük ebben a konkrét esetben.

3. feladat

Ez típusának legnehezebb feladata. Változók, véletlenszámok, ciklus és több elágazás együttes működését kell felismernie a gyereknek. Egy részük hasonló módon került alkalmazásra, mint az azokat tárgyaló leckékben, ám ezek mellett olyan megoldások is szerepelnek a projektben, amelyek a tananyagban nem fordultak elő pontosan ilyen szerkezetben.

Egy szellem feladatát kell értelmezni. Addig mozog, amíg az 500 kezdeti értékű energiája el nem fogy. Ez az energia a projekt működése során folyamatosan csökken (minden ciklusléptésben 1-gyel). A szellem véletlenszerűen mozog, így néha érintkezik az energiaital nevű szereplővel. Ebben az esetben az energiája 5-tel nő.

Gergő megoldása: „Zöld zászlóra kattintasz, a szellem energiája 500 lesz és megjelenik. Mindig változik -1-gyel, megy öt lépést és megfordul -10 fok és 10 fok között és nem megy ki a képből. Ha érintesz egy energiaitalt, akkor öt energiát nő az ereje a szellemnek, ha az energiája nulla, akkor eltűnik és mindent leállít."

Itt ugyanaz elmondható, mint az előző két feladat esetén. A gyerek szövegesen leírta a Scratch parancsait. Ez onnan is látszik, hogy bár a változókról szóló leckét már nem tanulta meg, a feladatban szereplő energia változó működését jól leírta. Ennek oka az lehet, hogy valóban csak a parancsokat írta le szövegesen, azonban az is elképzelhető, hogy annak ellenére, hogy a változókat nem tudja alkalmazni, értelmezni még képes azok működését. E feladat esetén tehát nem lehetünk biztosak a feladat megoldásának helyességében.

4. feladat

Ez már más típusú feladat. A gyereknek itt három lehetőség közül kellett kiválasztania a helyeset, amely egy szövegesen leírt projekthez megadja a szereplők feladatait. A projekt leírása a következő: a varázsló közeledik a táncoló gyerekhez. Amikor hozzáér, a gyerek eltűnik. A helyes megoldást úgy „rontottam el", hogy ne legyen egyértelmű, hogy melyik lehetőség hibás. Mindenhol, ugyanazokat a parancsokat alkalmaztam, csak esetleg más sorrendben, ami természetesen megváltoztatja az egész projekt működését. Például a két rossz megoldás esetén a varázsló vagy nem tűnteti el a gyereket, amikor hozzá ér, vagy saját maga tűnik el.

Gergő sikeresen kiválasztotta a jó megoldást. Saját bevallása szerint ez egy nagyon könnyű feladat volt. Ebben igaza is van, hiszen csak az első négy leckében megismert parancsokat alkalmaztam, ezek működését kellett ismerni a feladat helyes megoldásához.

5. feladat

Ez az előzővel azonos típusú feladat. A projekt leírása a következő: Két macska beszélget. „Ma balszerencsém lesz!" – mondja az egyik. „Miért?" – kérdezi a másik. „Reggel átment előttem egy fekete kutya…"

Itt tehát a beszélgető macskák feladatait kell meghatározni. Ebben az esetben nem csak ugyanolyan parancsokat használtam a három lehetőség esetén. Két esetben üzenetküldéssel valósítottam meg a párbeszédet, míg a harmadikban a mondd parancs időzítésével. A helyes megoldás az egyik üzenetküldéses változat volt. A másik két esetben a macskák egyszerre beszélnek.

E feladat helyes megoldásához tehát jól kell ismerni az üzenetküldések működését. Ez Gergő esetében nem teljesül, ugyanis azt a változatot jelölte meg, amelyben egyáltalán nem használtam üzenetküldést, tehát a macskák egyszerre mondják el a szövegeiket. Utólag, a feladatok megbeszélése során észrevette és kijavította ezt a hibát, de csak az üzenetküldésről szóló rövid magyarázatom után. Azt nem sikerült megtudnom tőle, hogy túl nehéznek tartotta-e az erről szóló leckét. Azaz hogy miért nem válaszolt jól erre a kérdésre, ugyanis azt mondta, hogy érthető volt a lecke.

6. feladat.

Az utolsó feladatban egy teljes projekt működését kellett kitalálni a megadott információk (képek, szöveg) alapján. A projekt a palacsinta készítés folyamatát mutatja

be – nagy vonalakban. A játék indításakor megjelennek az asztalon a hozzávalók: liszt, tej, cukor és tojás. Kattintással lehet őket a palacsintához adni. A palacsinta ebben az esetben egy lista. Ha az összes hozzávalót belekevertük (azaz amikor a lista négy hosszú lesz), kész az édesség, a háttér egy tál gőzölgő palacsintává változik.

Ez a feladat túl nehéznek bizonyult Gergő számára. A listák működését nem ismeri, így csak szóban találgatott a képek alapján a későbbi beszélgetés során, hogy mi történhet a projektben, de a papírra nem írt semmit.

Projekt

A teszt kitöltése után arra kértem a gyereket, hogy készítsen egy projektet. A témáját és tartalmát maga választotta ki, ötleteket sem kért. Egy egyszerű játékot készített, amely hasonló ötleten alapul, mint az elágazásokat bemutató leckéhez tartozó labirintusos játék. A háttér egy földalatti járatrendszert, fölötte pedig csillagos égboltot ábrázol. A járatból kell kivezetni egy kerek szereplőt úgy, hogy közben nem érhetünk a járat falához. A hátteret és a szereplőt Gergő maga rajzolta, nem a beépített szereplők közül választott. A festőablakot nagy kedvvel és ügyesen használta, az erről szóló leckékben bemutatott „trükköket” szívesen alkalmazta (pl. szabályos kör rajzolásához tartsuk lenyomva a shift billentyűt), még az én figyelmemet is felhívta rájuk az alkalmazásuk közben.

A játék megvalósítása a leckékre támaszkodik, ám használ benne azoktól kissé eltérő megoldásokat is. A szereplő irányítása a második leckében bemutatott módon történik. A fal érintésének ellenőrzését azonban nem az irányítást megvalósító parancsokhoz kapcsolta (mint ahogy az a negyedik leckében bemutatásra kerül), hanem egy külön feladatot adott a szereplőnek, amelyben ez az ellenőrzés történik.

A projekt nem igazán összetett, ám ez betudható annak, hogy a készítése a teszt kitöltése után történt, így a gyerek már fáradt volt. Azonban ebből is látható, hogy szívesen használja a környezetet, hiszen a fáradtsága ellenére is vállalta a projekt elkészítését.

Összességében azt mondhatom, hogy a gyerek szívesen foglalkozott a tananyaggal és az életkorának, vagy évfolyamának megfelelő szintű anyagot sikeresen elsajátította. Természetesen a Scratch használatában még gyakorlatot kell szereznie és sok projektet kell elkészítenie ahhoz, hogy magabiztosan tudja kezelni a környezetet és jól lássa, valamint megfelelően kihasználhassa a parancsok felhasználásának lehetőségeit.

Második kísérlet

A második kísérleti alany a 15 éves Viktória volt. Mint a fejezet bevezetőjében olvasható, ő már túllépett a 10-14 éves ajánlott korosztályon. Egy gimnázium humán tagozatának kilencedikes, kiemelkedő tanulmányi eredménnyel rendelkező tanulója. Kifejezetten humán érdeklődésű, az informatika programozás része nem igazán érdekli. Programozást eddig nem is tanult – vagy legalábbis nem emlékszik rá, hogy tanult volna –, az informatikatanárai inkább az alkalmazói ismereteket helyezték eddig előtérbe. A jelenlegi tanárával is az ECDL vizsgára készülnek, annak anyagát tanulják.

Viktória az életkorából vagy az érdeklődéséből adódóan nem bizonyult megfelelő tesztalanynak, ugyanis csak az első két leckét olvasta el. A többi esetében csak kipróbálta a játékokat. Elmondása szerint nem volt ideje a tananyagra, vagy mindig elterelték róla a figyelmét.

Ezért nem volt értelme az előzőhöz hasonlóan kitöltetni vele a tesztet, hiszen a kérdések többségére nem tudott volna érdemben válaszolni. Ezért őt arra kértem, hogy úgy próbálja megoldani a feladatokat, hogy közben használhatja a tananyagot, illetve a portálon található többi segítséget (például a parancslistát). Természetesen így nem volt ideje az egészet átolvasni, azonban a szükséges parancsokról azonban el tudta olvasni a leírásokat. A tesztet így is hamar, fél óra alatt töltötte ki.

1. feladat

A gyerek válasza: „Ha a fel gombot lenyomom, akkor hátra mozog, ha a le gombot, akkor előre. Ha balra gombot, akkor jobbra fordul, ha jobb gombot nyomom, akkor balra mozog.”

Ez a válasz tökéletesnek mondható. Ezen parancsok leírása szerepel az első két leckében, ezt megtanulta a gyerek. A válasz stílusán észrevehető a két gyerek közötti korkülönbség, Viktória sokkal összeszedettebben, sokkal inkább lényegre törően fogalmazott, mint Gergő.

2. feladat

Viktória válasza: (a rendőr)„megy oda-vissza, mert vissza tud pattanni a széléről.”

Itt már látszik, hogy nem olvasta végig a tananyagot. Csak a beszélt nyelvhez legközelebb álló szövegezésű parancsokat vette figyelembe a válaszadás során. A mindig parancs jelentése talán a „megy oda-vissza” megfogalmazásba beleérthető, de az elágazás egyáltalán nem jelenik meg a válaszban. A teszt kitöltése után megmutattam a

projekteket működés közben is Viktóriának. Ekkor magától mondta a jó megoldást, miszerint azért használtam a „ha…különben” parancsot, mert rendőr csak olyankor megy, amikor nem érint piros színt. Ellenkező esetben azt mondja, hogy „Piros!”.

3. feladat

A gyerek válasza: „A szellem repked össze-vissza. Közben csökken az energiája. Ha odaér az energiaitalhoz, akkor kap energiát, hogy tovább tudjon repkedni. De ha elfogy az energia, akkor a szellem eltűnik.”

Ez jó válasz, a szellem tényleg a leírtak szerint viselkedik. Viktória ennél a feladatnál olvasott utána először a parancsoknak. Az előző két feladat megoldásához egyáltalán nem használta a tananyagot, most azonban alaposan átolvasta a vonatkozó részeket. Ennek oka nem derült ki számomra. Talán itt kezdett el igazán érdeklődni a Scratch iránt, ekkorra hangolódott rá a teszt megoldására, vagy csak egyszerűen megtetszett neki a szellem szereplő jelmeze. Ezt nem árulta el.

4. feladat, 5. feladat

A két feleletválasztós feladatot Viktória túl könnyűnek találta. Mindkét esetben a jó válasz betűjelét karikázta be. Véleménye szerint ezeket a feladatokat a tananyag és a parancsok ismerete nélkül, pusztán a „józan észre” támaszkodva is meg lehet oldani, hiszen a rossz megoldások hibája szembetűnő.

6. feladat

Ez a feladat okozta a legnagyobb fejtörést a gyereknek. A segítségemet is kérte a megoldáshoz. Konkrét segítséget nem adtam neki, viszont a projekt leírását közösen átnéztük, közben pedig néhány rávezető kérdéssel próbáltam a helyes megoldás felé terelni. A megoldásból látszik, hogy ez sikerült is.

Viktória válasza: „Először a pult látható, és itt töröljük a palacsinta listát, hogy újult erővel kezdhessünk neki a palacsintakészítésnek. Megjelennek a hozzávalók kattintáskor és folyamatosan hozzáadjuk a hozzávalókat a hozzávalók kattintásakor és így elkészül a palacsinta és megjelenik a másik háttér.”

A fogalmazása itt bizonytalanabb, mint a 3. feladat megoldása esetén, néhány helyen nem is teljesen egyértelmű. Ennek az lehet az oka, hogy a gyerek talán még a válasz leírása közben sem volt teljesen biztos annak helyességében. Attól félt, hogy mivel a listák a nyolcadik lecke anyagát képezik, azok megértése neki már nem megy, tehát

valamit biztosan rosszul gondolt a megoldás során. Azonban megmutattam neki a projektet működés közben, így meggyőződhetett róla, hogy helyesen gondolkodott.

Szerettem volna, ha Gergőhöz hasonlóan Viktória is készített volna egy projektet, ő azonban erre nem vállalkozott. Ezt azzal indokolta, hogy ő „nem ért a Scratch-hez". Azonban megígérte, hogy a későbbiekben megpróbálkozik majd projektek készítésével.

Viktória esetében azt láttam, hogy a humán beállítottsága miatt „fél" a más jellegű gondolkodást igénylő témáktól. Ezen kívül azt gondolja magáról, hogy ő ezt úgysem fogja megérteni, ezért inkább bele sem kezd a megtanulásába. A teszt kitöltése közben látszott rajta, hogy felkeltette az érdeklődését a projektkészítés, azonban valószínűleg mégsem fog a későbbiekben a Scratch-csel foglalkozni.

Harmadik kísérlet

A harmadik kísérlet kimenetele az első kettőtől teljesen eltérő. Az alanya, Sára 13 éves, egy kőbányai általános iskola hetedik osztályos tanulója. Egy informatika órájuk van hetente, amelyen éppen az algoritmusok és adatok témakört tanulják az Imagine Logo környezet segítségével.

Amikor először megmutattam a tananyagot és magát a Scratch környezetet Sárának, akkor nagyon tetszett neki. Nagy lelkesedéssel kezdte el a tananyag elsajátítását. Készített egy táblázatot is, melyben minden leckéhez megjegyzések fűzhetők. Ő volt az egyetlen, aki segítséget is kért tőlem e-mail-en keresztül: „Meg szeretném kérdezni, hogy lehet a 3. leckénél hogy a nap + kutya változtatni tudja a jelmezét." Már a kérdésből látszik, hogy az előző leckék feldolgozása nem sikerült teljesen a gyereknek, ugyanis azokban esik szó a jelmezekről. E-mailben elmagyaráztam neki a jelmezek használatát. Több kérdést nem kaptam tőle.

A teszt kitöltésére szervezett találkozónkon derült ki, hogy csak az első négy leckét olvasta el. Amikor rákérdeztem ennek okára, azt válaszolta, hogy bár a példaprojektek nagyon tetszettek neki, a megvalósításuk összezavarta őt. Ugyanis már megszokta az Imagine kezelését és nehéznek találja a váltást, zavarja, hogy nem ugyanaz a parancsok elnevezése és a projektkészítés módja. Éppen ezért a tesztet sem töltötte ki és nem készített projektet sem. Pedig az általa elolvasott leckékhez tartozó projektek közül a legegyszerűbbet a tananyag olvasása során elkészítette, de magától nem szeretett volna animációt vagy játékot összerakni a találkozásunkkor.

Ebből a kísérletből az a tapasztalat vonható le, hogy bár a Scratch sokak számára egyszerűbbnek tűnik a többi, oktatásban használt környezetnél, lehetnek olyan gyerekek, akik mégis előnyben részesítik vele szemben például a Logót. Ez főként olyanokkal fordulhat elő, akik nagyon ragaszkodnak azokhoz a programokhoz, környezetekhez, amelyekhez hozzászoktak és nem tudnak áttérni más, az előzővel azonos funkciókat ellátó eszközre (például miután megtanulták a Microsoft Word kezelését, nem tudnak áttérni az azzal egyenértékű Open Office.org Writer-ére). Sári esetében az történt, hogy egy másik környezetet ismert meg előbb és azt követően találkozott a Scratch-csel. Nem tudott – vagy esetleg nem akart – áttérni az új környezetre. Fordított esetben, tehát ha az iskolában a Scratch-csel ismerkedik meg előbb, lehet, hogy az Imagine nem tetszett volna neki. Tehát a kísérlet kimenetele a gyerek tulajdonságaiból adódott, ebből nem következik, hogy minden gyerek, aki már megismerkedett egy programozási környezettel, idegenkedne egy újtól.

Fejlesztési lehetőségek

A tananyag a 10-14 éves korosztály számára készült azzal a céllal, hogy megismertesse a gyerekeket a Scratch környezettel, valamint a programozással. A fejlesztés tehát három irányban történhet. Készülhetnek új leckék a 6-10, a 14-18 évesek számára, valamint haladó leckék a 10-14 éveseknek.

A legkisebbeknek szóló leckék igénylik a leggondosabb tervezést. Ők még nem olvasnak magabiztosan, vagy egyáltalán nem is tudnak olvasni. Ezért a nekik szóló leckékben szinte csak képek és videók szerepelhetnek. Azonban a videók sem lehetnek olyan stílusúak, mint amelyek a tananyaghoz készültek, mert azokban is feliratok szerepelnek. A kicsiknek készült bemutatóknak szöveges alámondást kell tartalmazniuk. Egyes kiegészítő leckék videó változatát is el kellene készíteni, mivel a 6-10 évesek esősorban animációk készítésére használja a környezetet, ehhez pedig ismerniük kell a hangokról, a forgási stílusokról és a festőablakról szóló kiegészítő leckék tartalmát. Számukra több példaprojektet is kellene készíteni, amelyek felkeltik az érdeklődésüket a környezet iránt. Ezek lehetnének mesék, animációk, vagy interaktív képeskönyvek.

A 10-14 évesek számára készült haladó leckék többféle szerkezetben készülhetnek. Lehetnek a bevezető leckék továbbfejlesztései, amelyekben egy-egy bevezető leckében bemutatott parancstípust bonyolultabb szerkezetű projektben használnak fel (egy ilyen lecke így gyakorlatilag csak programozási technikákat mutatna be). A haladó leckék olyan értelemben is képezhetnék az jelenlegi tananyag folytatását, hogy a Scratch azon parancsait mutatnák be, amelyek a bevezető leckékben nem szerepeltek. Így talán kevesebb új lecke születne, de a tananyag teljes egészében lefedné a Scratch parancskészletét.

A 14-18 éveseknek szóló leckék inkább a kinézetükben különböznének az eddigiektől. E korosztály számára a tananyag a színessége, szövegbuborékos megjelenítése miatt túl gyerekesnek tűnhet. Számukra a jelenlegi szövegezéssel, de más megjelenítéssel megfelelőbbek lennének a bevezető leckék. A tananyag fejlesztése itt elsősorban annak megvalósításából állna, hogy a képek, szövegek a nagyobbak számára is vonzó módon is megjeleníthetők legyenek, azaz a honlapon több nézet is szerepeljen. Természetesen csupán a kinézet megváltoztatásával nem mondhatnánk azt, hogy a tananyag most már a 14-18 éves korosztálynak szól. Az e korosztály számára írt

szövegek lehetnek hosszabbak, így a magyarázatokban jobban el lehetne mélyedni egy-egy megvalósítás okaiban és hasznosságában.

Tehát a bevezető leckék elkészültével nem merült ki az összes Scratch-csel összefüggő tananyag készítésének lehetősége. A fentiekből látható, hogy még legalább három „csokornyi” lecke készíthető különböző korosztályok részére, különböző nehézségi szinteken.

Irodalomjegyzék

1. A Nemzeti alaptanterv kiadásáról, bevezetéséről és alkalmazásáról szóló 243/2003 (XII. 17.) Korm. rendelet
   http://www.okm.gov.hu/letolt/kozokt/nat_070926.pdf
2. Attitűd vagy „vas”? - Kerekasztal-beszélgetés a digitális tananyagokról és az SDT-ről
   Új Pedagógiai Szemle, 2005/10, 76-85. o., 2005.
3. Dancsó T.: Tehetséggondozás Logo pedagógiával – az informatika tantárgyi versenyek tükrében
   http://www.oki.hu/oldal.php?tipus=cikk&kod=informatika-Dancso-Tehetseggondozas
4. Domján E.: A hypertext gondolat felhasználása az üzemtan oktatásában
   http://www.kfki.hu/chemonet/hun/food/phd/kertesz/domjan.html
5. Katona N. – Oakland, T.: Tanulási stílus – Egy integratív megközelítés
   Alkalmazott Pszichológia, I/01, 17-29. o., 1999.
6. Kárpáti A.: Differenciálás a nevelés szemszögéből
   http://edutech.elte.hu/multiped/ped_09/ped_09.pdf
7. Kerettanterv, Informatika
   http://www.okm.gov.hu/letolt/kozokt/kerettanterv/korrekturas/alapfok/f05_informatika.rtf
8. Komenczi B.: Didaktika elektromagna? Az e-learning virtuális valóságai
   Új Pedagógiai Szemle, 2004/11, 31-49. o., 2004.
9. Kovács Ilma: Távoktatástól távoktatásig
   http://mek.niif.hu/04500/04524/04524.pdf
10. Lakosné Makár E.: Módszertani ajánlás tanítóknak az informatika alsó tagozatos tanításához és alkalmazásához
    http://www.oki.hu/oldal.php?tipus=cikk&kod=gyermek-Lakosne-modszertan
11. M Nádasi M.: Az oktatás szervezeti keretei és formái
    In: Falus Iván (szerk.):Didaktika, 345-368. o.
    Nemzeti Tankönyvkiadó, Budapest, 2003.

12. Resnick, M.: All I Really Need to Know (About Creative Thinking) I Learned (By Studying How Children Learn) in Kindergarten
ACM Creativity & Cognition Konferencia, Washington, 2007.
13. Resnick, M. – Kafai, Y. – Maeda, J.: A Networked, Media-Rich Programming Environment to Enhance Technological Fluency at After-School Centers in Economically-Disadvantaged Communities
Javaslat a National Science Foundation számára, 2003.
14. Sós M.: 10–14 éves diákok számítógép-használati szokásainak vizsgálata
Új Pedagógiai Szemle, 2005/11, 83-99. o., 2005.
15. Szántó S.: Az algoritmikus gondolkodás fejlesztése általános iskolában
Új Pedagógiai Szemle, 2002/05, 84-175. o., 2002.
16. Széplakiné Józsa E.: Comenius-LOGO
http://www.oki.hu/oldal.php?tipus=cikk&kod=gyermekuj-szoftver-Szeplakine-Logo
17. Török B.: A diákok számítógép-használati szokásai – internetezés és elektronikus levelezés
Új Pedagógiai Szemle, 2007/07-08, 105-122. o., 2007.
18. Turcsányiné Szabó M.: Képességfejlesztés teleházakban – A mentorálás egy működő modellje
http://www.oki.hu/oldal.php?tipus=cikk&kod=iii-Turcsanyine

MELLÉKLETEK

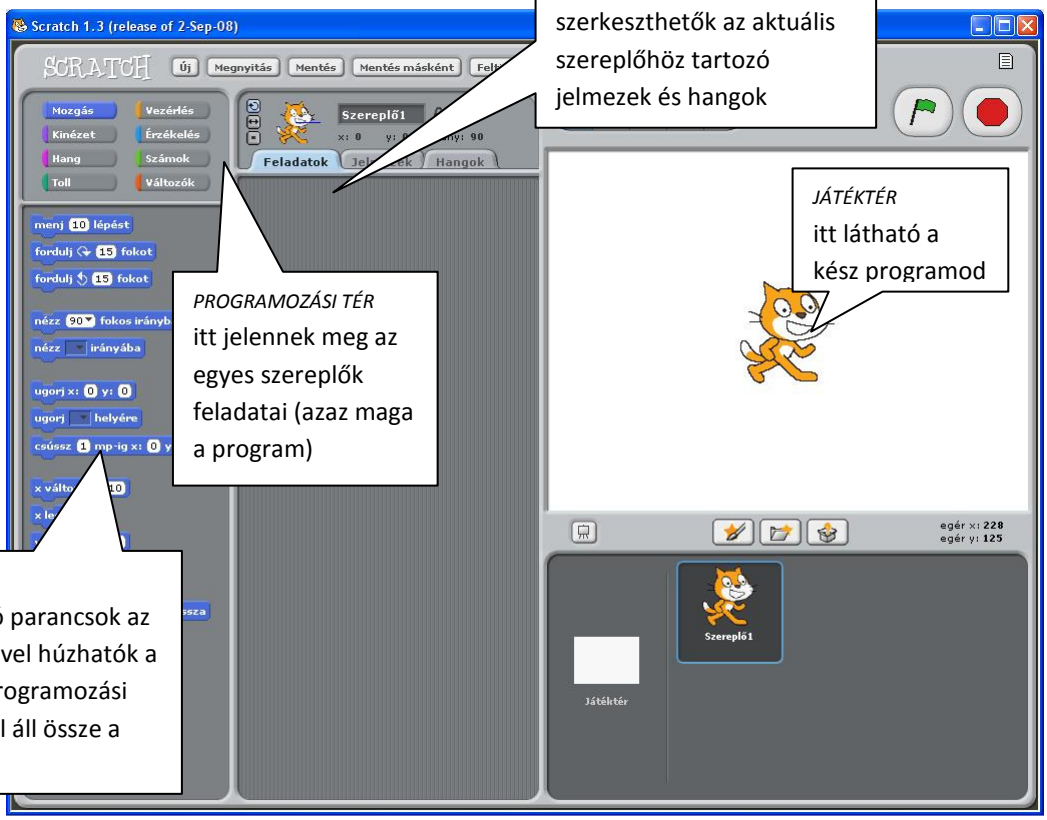
Bevezető leckék

Scratch Magyarország Portál

Takács Valéria
2009.

1. Ismerkedés a Scratch környezettel

A Scratch mesék és játékok készítésére alkalmas programozási környezet. A következőkben megtanulhatod a kezelésének első lépéseit.
A Scratch indításakor ez a kép látható:



JELMEZEK, HANGOK
ezen a két fülön láthatók és szerkeszthetők az aktuális szereplőhöz tartozó jelmezek és hangok

JÁTÉKTÉR
itt látható a kész programod

PROGRAMOZÁSI TÉR
itt jelennek meg az egyes szereplők feladatai (azaz maga a program)

PARANCSKÉSZLET
Az itt található parancsok az egér segítségével húzhatók a szomszédos programozási térbe. Ezekből áll össze a program.

A Szereplők

A Scartchben a játékok, animációk szinte elképzelhetetlenek szereplők nélkül, ezért fontos megismerni a kezelésüket. Kezdetben egyetlen szereplő van a játéktérben, a képen látható macska. Ezen könnyen változtathatsz: néhány kattintással új szereplőket tölthetsz be vagy módosíthatod a már meglévőket. Ehhez a játéktér alatti, illetve a programozási tér jelmezek fülén lévő gombokat használhatod.

ÚJ SZEREPLŐ FESTÉSE
saját, teljesen egyéni hátteret rajzolhatsz a *festőablak* használatával

SZEREPLŐ BETÖLTÉSE VÉLETLENSZERŰEN
az új szereplőt a program véletlenszerűen „sorsolja ki” a Scratch beépített szereplői közül

AKTUÁLIS SZEREPLŐ
ő az, akinek feladatokat adhatunk a programozási térben.
Ha egy szereplő jelmezeit szeretnéd változtatni, akkor először itt kell rákattintanod, hogy ő legyen az aktuális.

SZEREPLŐ BETÖLTÉSE
választhatsz egy szereplőt a Scratch beépített szereplői közül, vagy szereplőként betölthetsz egy képet a számítógépedről. Az új szereplő megjelenik a játéktérben, a főbb jellemzői (neve, feladatainak száma) pedig a játéktér alatti szürke mezőben.

JELMEZEK
Egy szereplő jelmezei hasonlóak, mint egy film képkockái. Általában a szereplő „pillanatfelvételei” egy bizonyos tevékenység elvégzése közben (pl. a mozgásának fázisai). A jelmezek nagyon fontosak, gyakran tőlük válik „élővé” a szereplő.

SZEREPLŐ NEVE
az új szereplők automatikusan a *szereplőx* elnevezést kapják. Ez több szereplő esetén kavarodást okozhat, ezért a szereplőket **mindig nevezd el**!

BETÖLTÉS
betölthetsz egy jelmezt a Scratch beépített jelmezei közül, vagy egy képet a számítógépedről. Az új jelmez az utolsó lesz a jelmezek sorában.

FESTÉS
saját, teljesen egyéni jelmezt rajzolhatsz a szereplőnek a *festőablak* használatával

TÖRLÉS
a jelmezt törölheted ezzel a gombbal, de csak ha van még rajta kívül másik a sorban

AKTUÁLIS JELMEZ
A jelmezek közül a kékkel keretezett. Ilyen alakban jelenik meg a szereplő a játéktérben

MÓDOSÍTÁS
a *festő ablak* használatával módosíthatod a jelmezt

MÁSOLÁS
valójában a jelmez kettőzését jelenti, megjelenik belőle még egy példány a jelmezek sorában. Akkor érdemes alkalmazni, ha két jelmez csak kevésben különbözik egymástól, így az új jelmezt a másolatból könnyen elkészítheted

Új háttér betöltése

Ahhoz, hogy egy animáció vagy egy játék szép és élvezetes legyen, fontos a megfelelő háttér. Ez általában nem fehér, szóval változtasd meg! Mivel a háttér valójában a Scratch egy speciális szereplője, ezért a kinézetének módosítása hasonlít a szereplők változtatásához, tehát nem lesz nehéz dolgod:

- Kattints a játéktér alatti mezőben a fehér téglalapra (ez most a beállított háttér)
- a programozási környezetben válaszd ki a hátterek fület
- itt megjelennek a szereplőknél megismert lehetőségek:

FESTÉS
saját, teljesen egyéni hátteret rajzolhatsz a *festőablak* használatával

BETÖLTÉS
betölthetsz egy már létező képet a számítógépedről (vagy a Scratch hátterei közül). Az új kép az utolsó lesz a hátterek sorában

TÖRLÉS
a hátteret törölheted ezzel a gombbal, de csak ha van még rajta kívül másik a sorban

MÓDOSÍTÁS
a *festő ablak* használatával módosíthatod az aktuális hátteret

MÁSOLÁS
valójában a háttér kettőzését jelenti, megjelenik belőle még egy példány a hétterek sorában

2. Megmozdulnak a szereplők

Egy játék akkor szórakoztató, ha irányíthatjuk a szereplőit (vagy legalább a főszereplőt). Ebből a leckéből megtudhatod, hogyan lehet ezt megtenni.

Egy egyszerű játékot fogunk elkészíteni, amelyben egy papagájt lehet röptetni az égen:

A játék elkészítése

- Először töröld ki a macskát a játéktérből és töltsd be az Animals könyvtárból a parrot1-a nevű papagájt. A nevét változtasd Papagájra. A háttér legyen kék ég felhőkkel (ezt letöltheted a honlapról vagy megrajzolhatod a festőablakban.)
- Most a papagájnak adunk feladatokat. Ehhez húzd a parancsokat az egérrel a parancskészletből a programozási térbe. A parancsokból akkor lesz feladat, ha megfelelő sorrendben összeillesztjük őket egy sapka alatt.
- Először oldjuk meg azt, hogy a fel nyíl lenyomásakor a papagáj haladjon egy kicsit előre. Ehhez a következő parancsokra lesz szükség:

A sapka alapbeállításban a szóköz billentyű lenyomását figyeli:

A fekete nyílra kattintva legördülő listából választható másik billentyű.

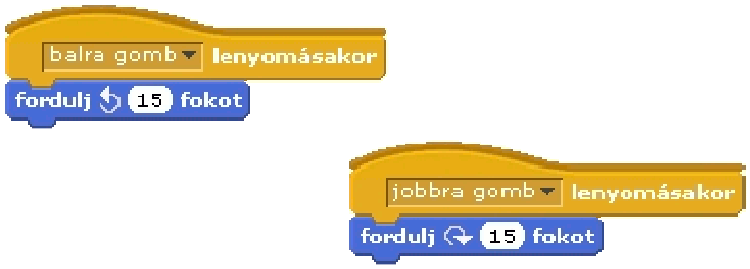
A programot azonnal kipróbálhatod. A fel billentyű lenyomásakor a papagáj a játéktérben 10 lépést (10 képpontnyit) halad előre. Azonban nem áll meg a szélénél, kirepül a képből. Ennek elkerülésére illeszd a

parancsot az eddigi kettő közé. Hatására a fel gomb lenyomásakor ha a papagáj éppen a játéktér szélén van, akkor megváltoztatja az irányát és csak azután halad tovább.

A parancsok színe megegyezik annak a parancscsoportnak a színével, ahol megtalálható. Tehát ez a parancs a narancssárga *vezérlés* csoportba tartozik.

Ha szeretnéd, hogy a madarad gyorsabban repüljön, írd át a „sebességét” 20 lépésre! Ha ide negatív számot írsz, akkor a szereplő hátrafelé fog haladni.

- Ahhoz, hogy a papagáj haladási irányát is változtatni tudjuk, még két egyszerű feladatot kell adnunk az eddigihez:



- Most már csak az a baj, hogy a papagáj nem mozgatja a szárnyait repülés közben. Ezen könnyen lehet segíteni az előző leckében említett jelmezek ügyes váltogatásával. Töltsd be a Papagáj jelmezei közé az Animals könyvtárból a parrot1-b nevűt. Ha egymás után kattintasz a jelmezekre, a játéktérben úgy látszik, mintha a papagáj csapkodna a szárnyaival. A jelmezeket nem csak így lehet váltogatni.

  Ha minden egyes feladathoz hozzáilleszted a 

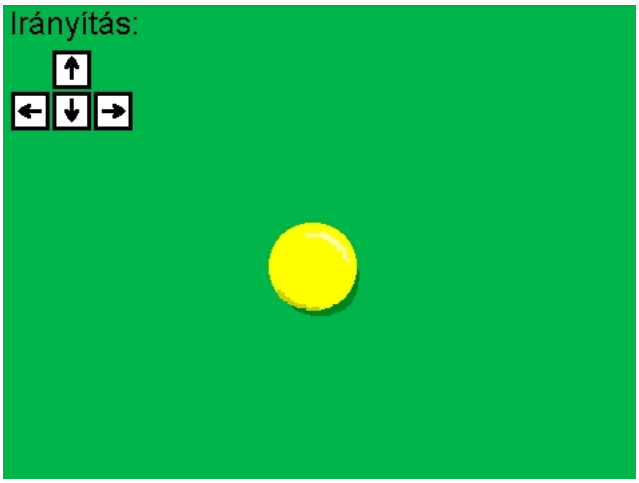
parancsot, akkor a szereplő a gombnyomások hatására elfordul vagy repül egy kicsit és még a sorban következő jelmezt is magára ölti. (Mivel most csak két jelmez van, ezért e kettő fog váltakozni.)

A papagáj feladatai tehát:

Mentsd el a projektedet papagaj néven!

Variáció

Hogyan készülhetett ez a program? Próbáld meg te is elkészíteni!



Ha elakadtál, töltsd le a programot a honlapról vagy olvasd el a hozzá fűzött megjegyzéseket itt:

3. Ismétlődések

Az előző leckében a szereplő akkor tett meg valamit, ha leütöttünk egy billentyűt. Most megmutatjuk, hogy szereplőink beavatkozás nélkül is tudnak feladatokat elvégezni.

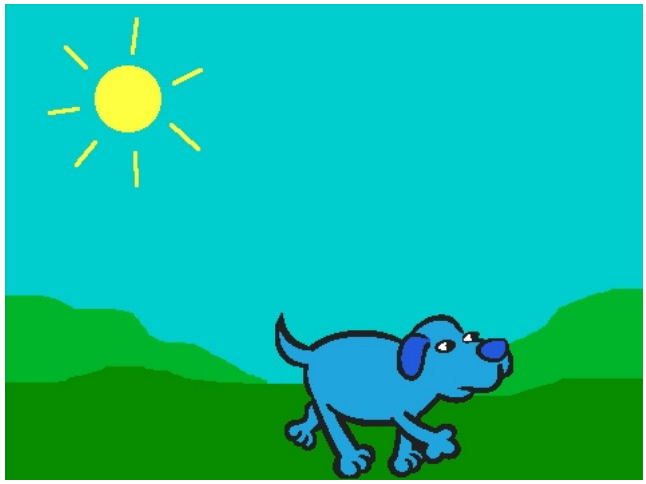
Ciklusok

A ciklusok ismétlődő tevékenységek megvalósítására szolgálnak. Ebben a leckében két ciklusfajtával ismerkedhetsz meg:

MINDIG
folyamatosan ismétli a benne elhelyezett parancsokat

ISMÉTELD [10]
a megadott számú alkalommal ismétli a benne elhelyezett parancsokat (ebben az esetben 10-szer)

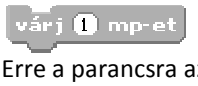
Most lássunk egy példát is a parancsok használatára. Egy olyan programot készítünk, amelyben egy napsütéses délutánon egy kutya végigsétál a képernyőn:



A program elkészítése

- Szereplők: Nap (a jelmezeit letöltheted a honlapról), Kutya (jelmezei: Animals/dog2-b, Animals/dog2-c)
- A parancsok végrehajtása a zöld zászlóra kattintáskor indul, tehát a 

sapkához kapcsoljuk őket.
- A Nap a program indításától kezdve folyamatosan – mindig – változtatja a jelmezeit, tehát az ő feladata:



Erre a parancsra azért van szükség, mert a Scratch nagyon gyorsan hajtja végre a parancsokat, így a Nap túl gyorsan váltogatná a jelmezeit. A várakozás parancsnak köszönhetően a jelmezek cseréje után vár egy fél másodpercet és csak azután folytatja a parancsok végrehajtását.

Ez a nyíl jól érzékelteti a *mindig* ciklus működését. A szereplő jelmezt vált, vár fél másodpercet, majd visszatér az első parancshoz (válts jelmezt). Ezért van az, hogy a mindig ciklus után illesztett parancsok nem hajtódnak végre.

- A Kutya nem ismétli végtelen sokszor a feladatát: elindul, megy egy darabig – *valahányszor* A Kutya nem ismétli végtelen sokszor a feladatát: elindul, megy egy darabig – valahányszor megismétli a 

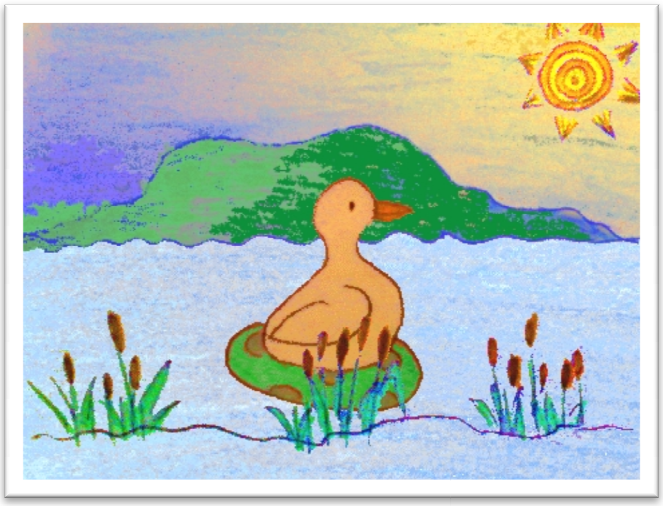
parancsot – majd megáll. Az ő feladata:

A szereplő most 20-szor megismétli a cikluson belüli parancsokat, majd megáll (mivel más feladatot nem adtunk neki).

Erre a parancsra azért van szükség, mert a zöld zászló lenyomásakor a kutya mindig onnan indul, ahol az előző futtatáskor befejezte a sétát, tehát előfordulhat, hogy „kimegy a képből". Ahhoz, hogy a kutya visszapattanáskor ne álljon fejre, a forgási stílusát át kell állítani erre: 

Mentsd el a projektedet kutya néven!

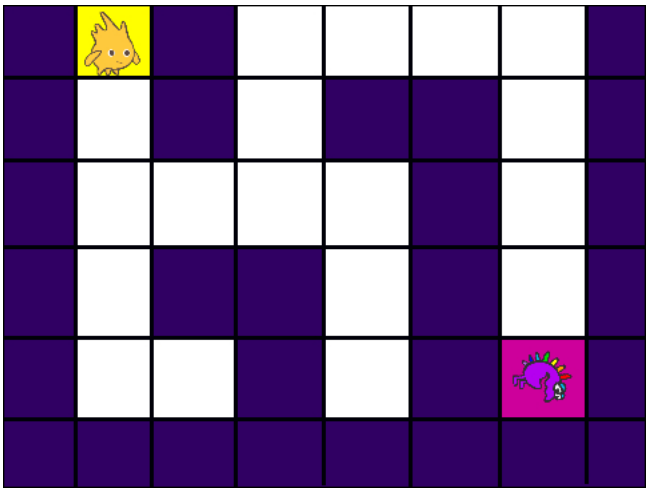
Feladat
Készíts animált képeslapot az alábbi példák alapján:



4. Feltételek

A játékok jelentős részében a szereplők nem ugyanúgy viselkednek minden helyzetben. Ennek megvalósítására feltételekhez kötjük az egyes parancsok végrehajtását.

Ezt az egyszerű labirintusos játékot fogjuk elkészíteni, melynek célja, hogy a Gobo nevű szereplőt a sárga négyzetről a magenta színűre juttassuk anélkül, hogy a falhoz érnénk. A Lila szereplő ekkor örömmel kiáltson fel, hogy barátja megtalálta:



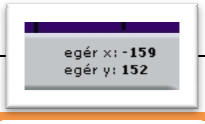
A játék elkészítése

- A háttér letölthető a honlapról. Szereplők: Fantasy/gobo1 (mérete az eredeti méret 40%-a), Fantasy/fantasi11 (mérete az eredeti 35%-a).
- Gobo mozgatása tetszőleges (pl. a második leckében megismert is lehet). Ezúttal egy koordináta-rendszerre támaszkodót használunk, mivel a szereplő csak jobbra-balra, illetve fel-le mozog és az ilyen típusú mozgás az x és y koordináták változtatásával jól megvalósítható:

- Fontos, hogy a játék kezdetén Gobo a start (sárga) négyzeten álljon. Ehhez a következő feladatot kell adnunk neki:

UGORJ
ez a parancs a játéktér megadott koordinátájú pontjába helyezi át a szereplőt

A kívánt koordináták könnyen „megszerezhetők”: helyezd a szereplőt a sárga mezőre. Ha ezután kiválasztod a parancslistán a mozgás csoportot, akkor az *ugorj* parancs koordinátái a szereplő aktuális helyét mutatják (azaz a most szükséges „startpozíciót”).
Egyszerűbb megoldás, ha leolvasod az egérmutató koordinátáit a játéktér alatti sávból akkor, amikor az a sárga négyzet fölött áll.



- A játék lényege, hogy a szereplőnek úgy kell végigmennie a labirintuson, hogy nem érhet hozzá a falhoz. Azonban Gobo ezt most gond nélkül megteszi. Ennek megakadályozásához arra lenne szükség, hogy ha falhoz ér, akkor „kapjon büntetést”, pl. kezdje újra a játékot.

Ezt a 

paranccsal valósíthatod meg:

HA
ha teljesül a megadott feltétel, akkor végrehajtódik a benne megadott parancs (csak akkor)

érintesz szín?
Az érzékelés csoportba tartozó parancs. Igaz értéket ad, ha a szereplő érinti a megadott színt a játéktérben (szín megadásához kattints a színmintán, majd válassz egy színt a palettáról vagy a játéktérből).

Ebben az esetben tehát a szereplő mindenképpen lép egyet és csak azután ellenőrzi, hogy érinti-e a falat.

A fel gomb lenyomásakor a szereplő 5 lépést megy felfelé. Ez után a program megvizsgálja, hogy érint-e sötétlila színt (ez most a labirintus falát jelenti). Ha érint, akkor visszaugrik a sárga mezőre (amelynek megadtuk a helyét az ugorj parancsban).

- A szereplő feladatai tehát:

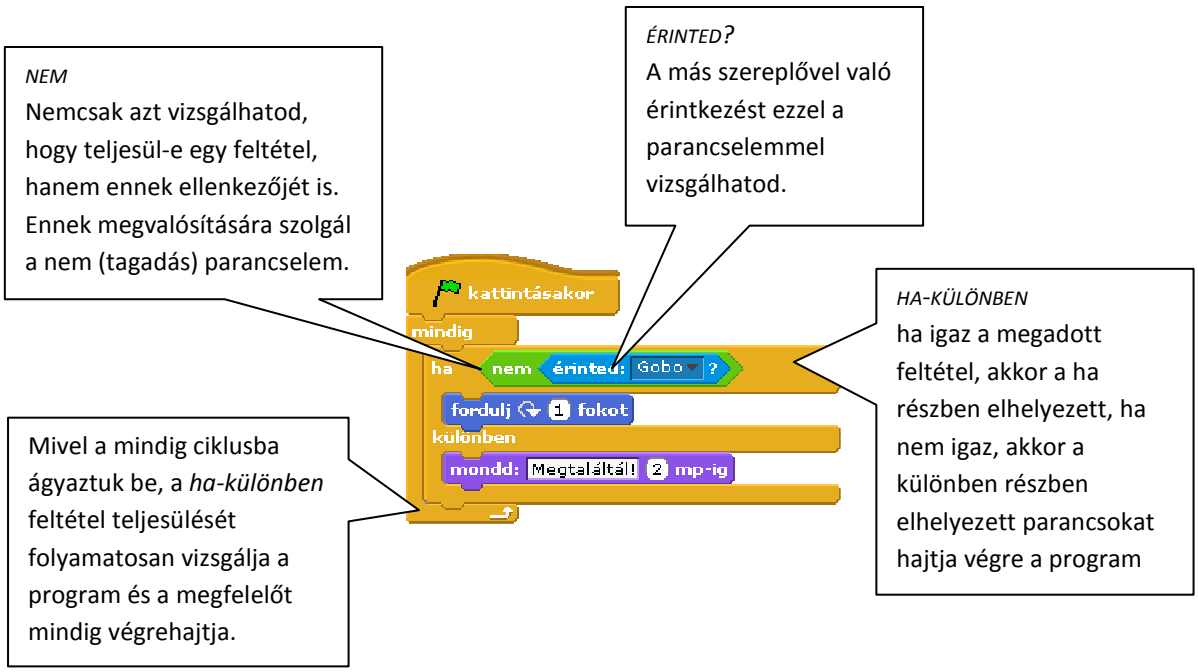
A következő oldalon megismerheted a labirintus végén várakozó másik szereplő feladatait.

Egymásba ágyazás

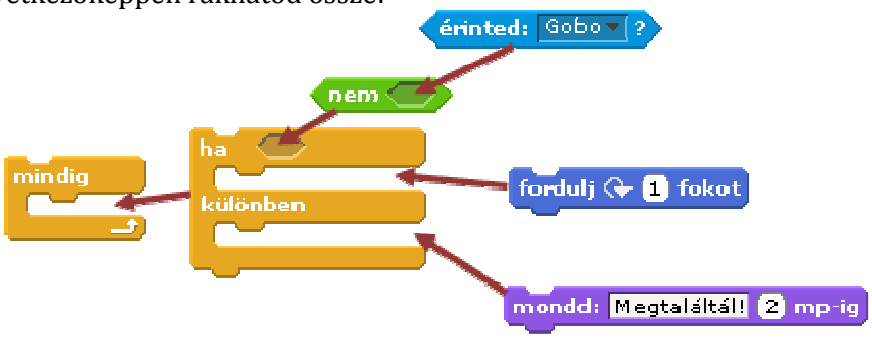
A lila szereplő ha nem érintkezik Goboval, akkor folyamatosan egy helyben forog. Különben (a Goboval való érintkezéskor) azt mondja, hogy "Megtaláltál!".

Ennek megvalósításához több parancs egymásba ágyazására van szükség. Ilyennel már találkoztál a ciklusoknál is, amikor a ciklus belsejébe illesztetted a parancsokat. Ennek a leckének az első felében pedig a *ha* parancs feltételét kellett megadnod beillesztés segítségével.

A szereplő feladatai:



Ezt a feladatot a következőképpen rakhatod össze:



A *ha...különben* parancs két ágába helyezett parancsok felcserélhetők, ha a feltételt tagadod. Így a két feladat hatására ugyanazt csinálja a szereplő:

Mentsd el a projektedet labirintus néven!

5. Üzenetek

Többszereplős játékok esetén elengedhetetlen, hogy a szereplők valamilyen módon kommunikáljanak egymással. Erre valók a Scratchben az üzenetek.

Először megismerheted a használatukat egy nagyon egyszerű programon keresztül, majd megmutatjuk, hogy hogyan teheted a segítségükkel teljesebbé a már elkészült játékaidat.

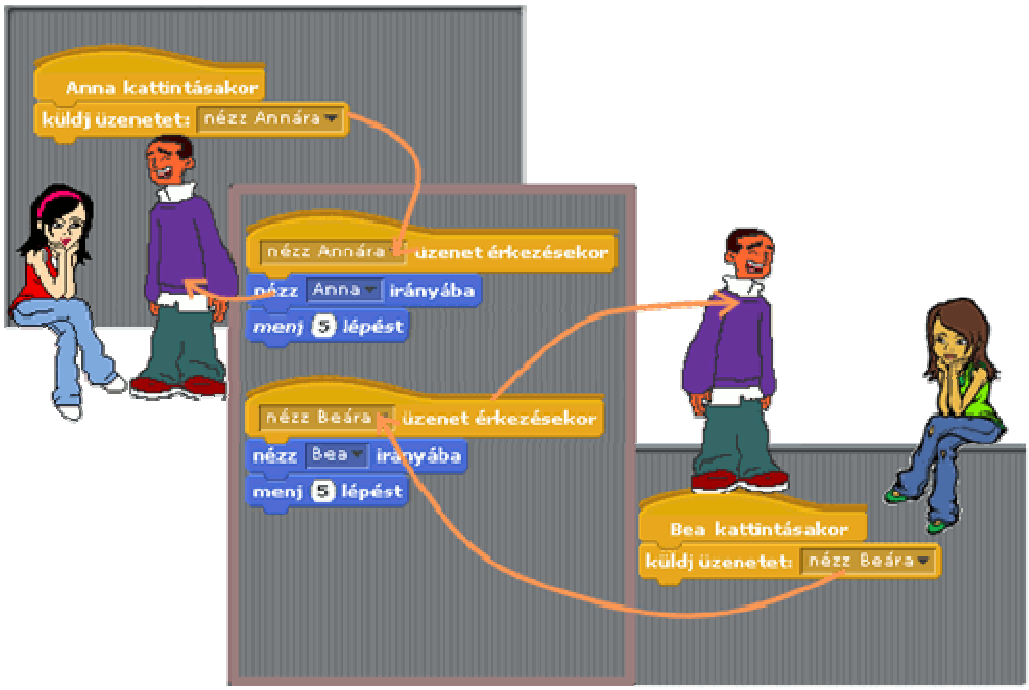
Ismerkedés az üzenetekkel

- A játék szereplői: Anna (People/girl4-sitting), Bea (People/girl5) és a Fiú (boy4 laughing). A háttér letölthető a honlapról.
- A játék igen egyszerű: ha Annára kattintunk, akkor a Fiú Annára néz és közelebb lép hozzá, ha Beára kattintunk, akkor a Fiú felé néz és hozzá lép közelebb. Ezt üzenetek használatával valósíthatjuk meg, ugyanis szereplőink üzenetek küldésével érhetik el, hogy egy másik szereplő csináljon valamit.
- A lányoknak csak egy feladatuk van: ha rájuk kattintunk, küldjenek üzenetet a fiúnak, hogy nézzen rájuk. Ez Anna esetében:

ÚJ ÜZENETET úgy hozhatsz létre, hogy a fekete háromszögre kattintva a legördülő menüből kiválasztod az *Új...* lehetőséget, majd beírod az üzenet nevét. (Itt választhatsz a már létező üzenetek közül is)

- A Fiú pedig a „nézz Annára” üzenet érkezésekor (ami pontosan egy időben történik az üzenet küldésével) Annára néz és közelebb lép hozzá. (A Fiú forgási stílusát persze meg kell változtatni ahhoz, hogy ne álljon fejre akkor, amikor egyik lányról a másikra néz.)

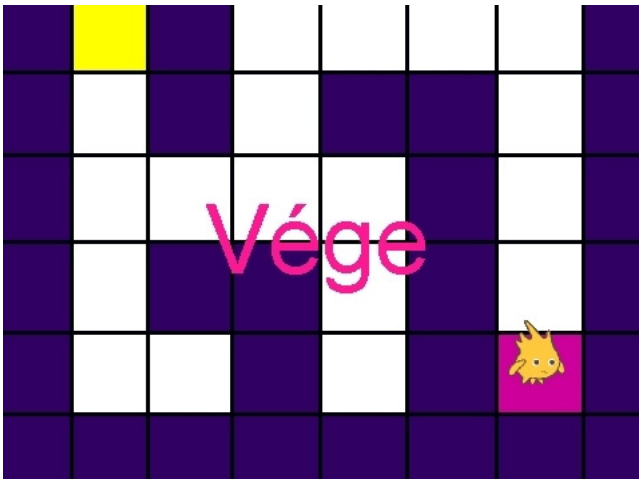
- Bea esetén ugyanezt kell tennünk. Tehát a lejátszódó üzenetküldés a következőképpen zajlik:



Mentsd el a projektedet suli néven!

Vége a játéknak

Üzenetek használatával „továbbfejlesztheted” eddigi programjaidat. Például a 4. lecke labirintusos játékát:



- A játék alapja nem változott. Az egyetlen újdonság, hogy amikor Gobo eléri a cél mezőt, akkor véget ér a játék: megjelenik a Vége felirat.
- Két új szereplő van: a cél mező és a Vége felirat (letölthetők a honlapról, de könnyen el is készíthetők).
- A cél mező feladatai:

VÁRJ EDDIG: [FELTÉTEL]
A megadott feltétel teljesüléséig vár, majd elvégzi az utána következő feladatokat. Ebben az esetben addig vár, amíg Gobo hozzá nem ér, majd vége üzenetet küld.

- A felirat feladatai:

TŰNJ EL
„láthatatlanná” teszi a szereplőt. Most azért használjuk, mert a játék kezdetén és a játék alatt nincs szükség a Vége feliratra, akkor nem kell látszania.

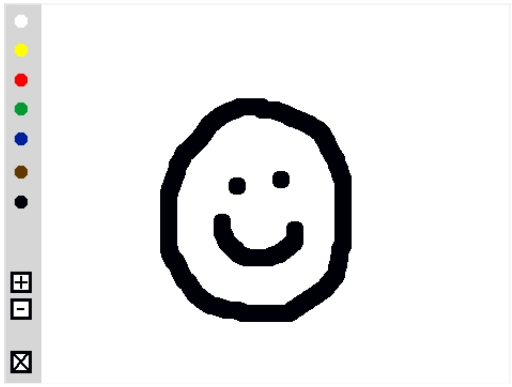
KERÜLJ LEGELŐRE
a szereplők a létrehozásuk sorrendjében egymáson elhelyezkedő *rétegekre* kerülnek, ezért el tudják takarni egymást. Ez a parancs „legfelülre” helyezi a szereplőt, ő eltakarja a többieket, de senki nem takarja őt.

JELENJ MEG
a tűnj el parancs fordítottja. Láthatóvá válik az addig láthatatlan szereplő.

A felirat tehát a játék kezdetén láthatatlanná válik és csak akkor jelenik meg újra, ha a cél mező elküldi neki a vége üzenetet.

6. Rajzoljunk

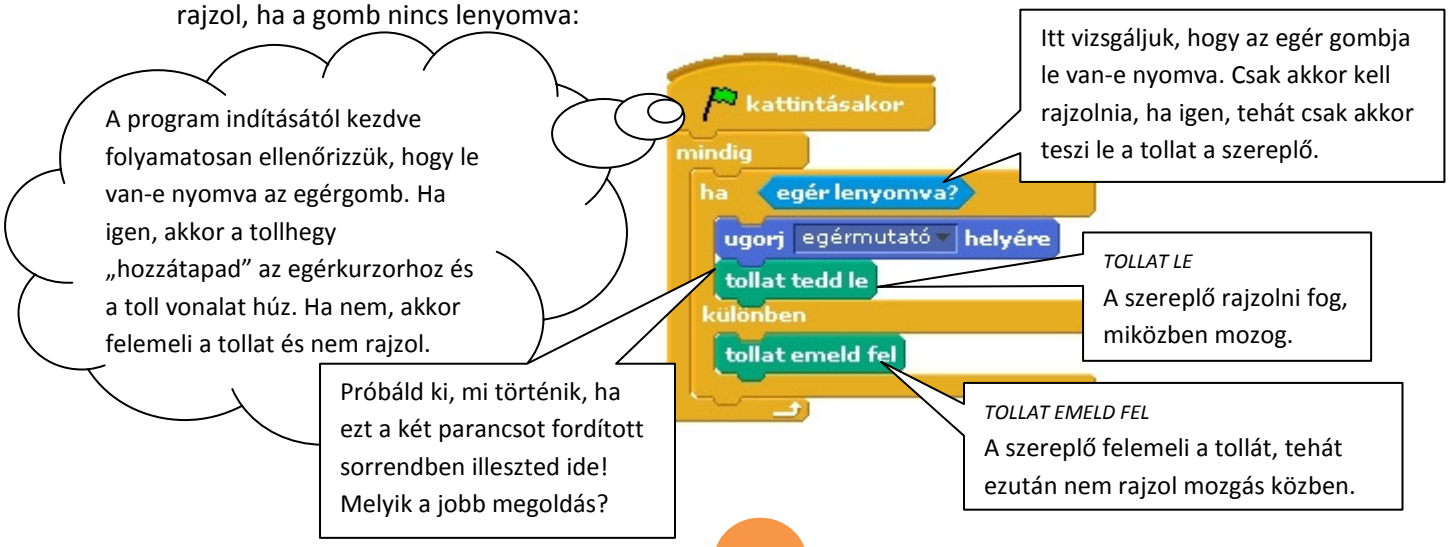
A Scratch nem csak játékok és animációk készítésére alkalmas. Ebben a leckében egy egyszerű rajzolóprogramot állítunk össze.



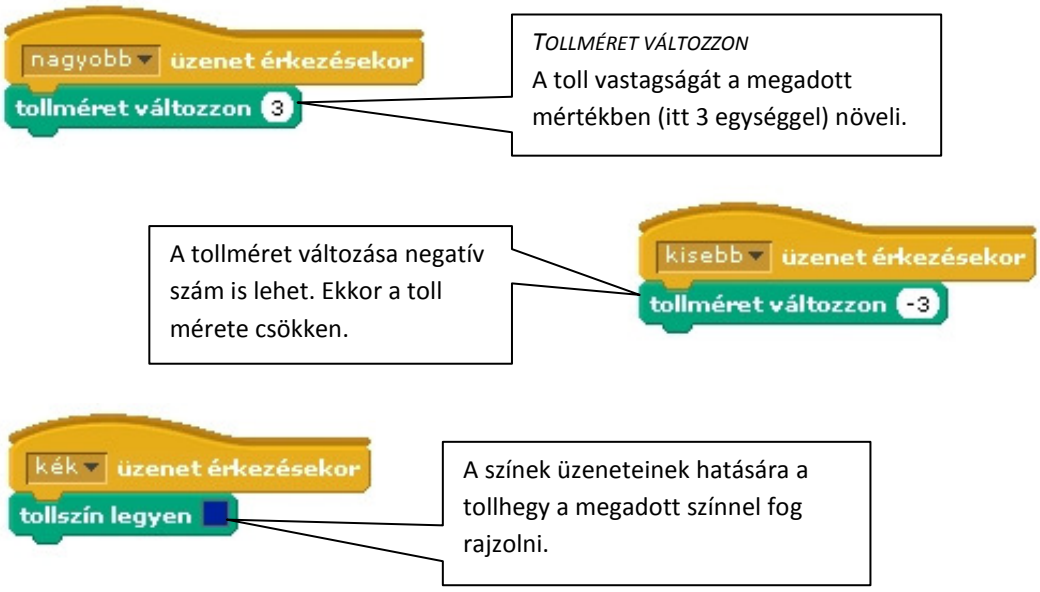
A program elkészítése

- A program lehetőségei: szín változtatása, tollméret változtatása, rajzok törlése és természetesen rajzok készítése.
- Szereplők: színes pontok (színváltáshoz), + és – gombok (méret változtatásához), x gomb (rajzok törléséhez), tollhegy (rajzoláshoz). A tollhegy itt egy pici fekete pont, de természetesen más jelmeze is lehetne (pl. egy ceruza).
- A program elkészítéséhez a **parancskészlet** Toll csoportját fogjuk használni.
- Lássuk először a bal oldali sáv szereplőinek feladatait. Ők többnyire csak üzenetet küldenek a tollhegynek, hogy min változtasson (a színekből csak egyet mutatunk be példaképp):

- A tollhegy a megadott színnel és tollvastagsággal rajzol, ha lenyomva tartjuk az egér bal gombját, és nem rajzol, ha a gomb nincs lenyomva:



- Reagálás a többi szereplőtől kapott üzenetre:

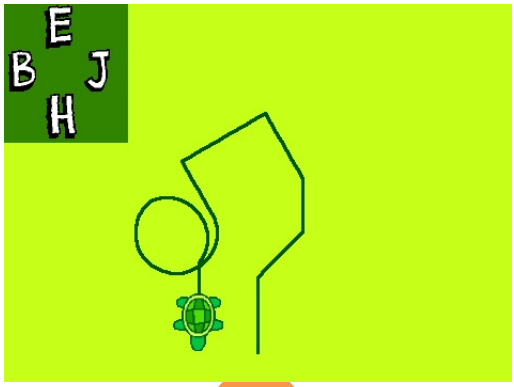


- Már csak az lehet probléma, hogy a szürke sávba is tudunk rajzolni. Ennek két megoldása van: az egyik, hogy a szürke sávot nem a háttérre rajzoljuk, hanem szereplőként hozzuk létre. Mivel a toll a háttérre rajzol, ez a szereplő azt el fogja takarni.

- A másik megoldás egy feltétel beillesztése: a tollhegy csak akkor rajzoljon, ha az x-koordinátája nagyobb, mint a szürke sáv jobb szélének ezen adata:

Mentsd el a projektedet rajztabla néven!

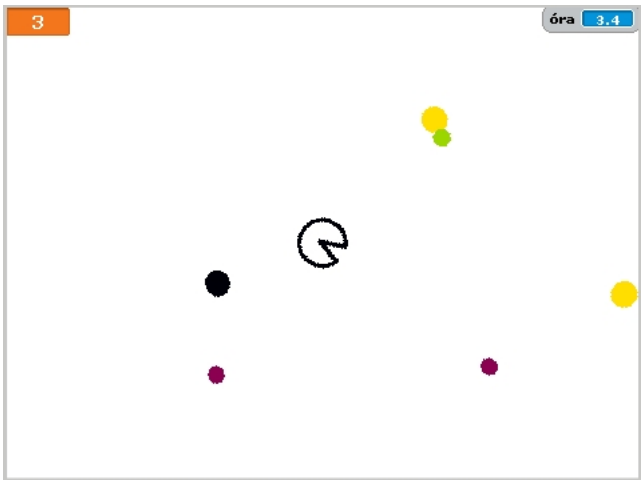
Készíts más típusú rajzolóprogramokat! Ismered az Imagine-t? Próbálj hozzá hasonló programot készíteni!



7. Változók

Az eddig megismert módszerekkel nem lehet igazán izgalmas, „tétre menő” játékokat készíteni. Ebben a leckében végre megtudhatod, hogyan lehet például pontokat szerezni és időt mérni.

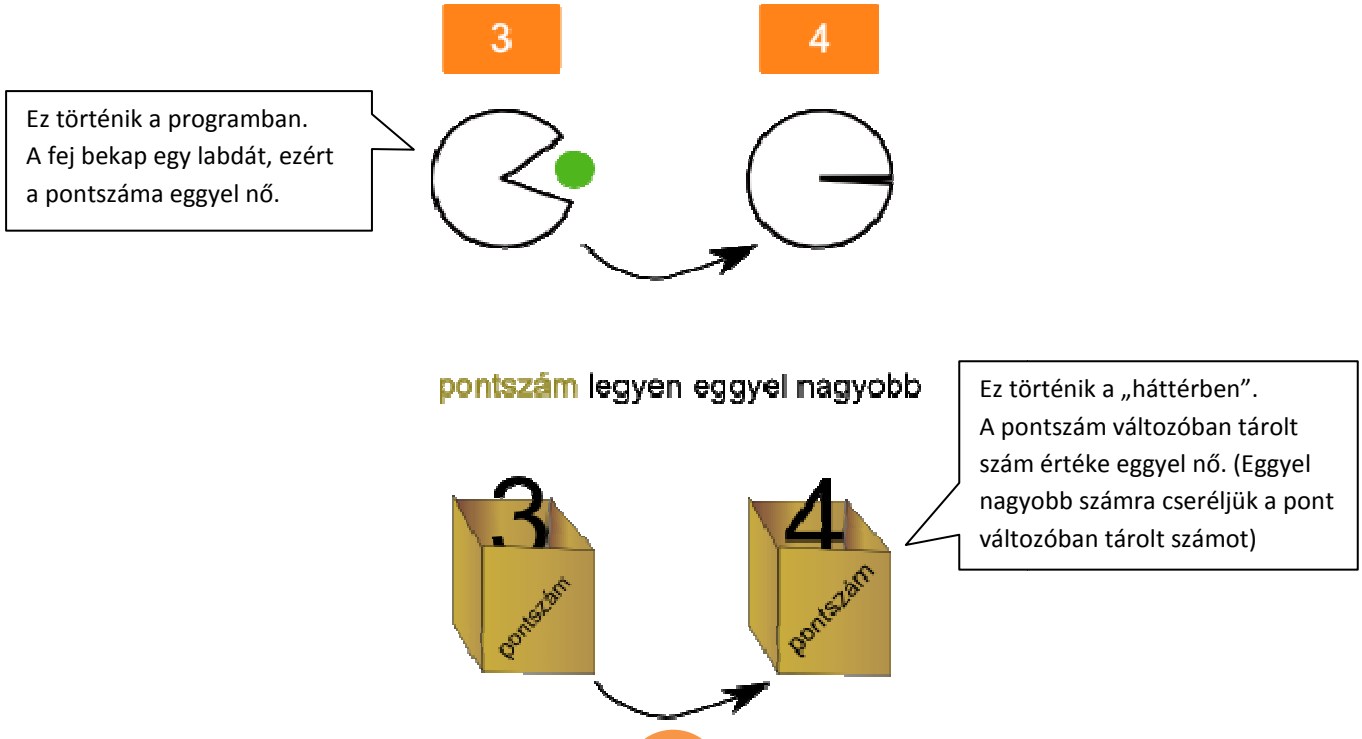
A következő játékot készítjük el:



Változó

A változó (az informatikában) legegyszerűbben megfogalmazva olyan hely a számítógépen, ahol valamilyen mennyiséget tárolunk. A változónak van neve és valamilyen értéke – ez az érték a program futása során változhat.

A változót úgy lehet elképzelni, mint egy dobozt, amelyben tárolhatunk valamilyen értéket. Amikor szükségünk van rá, akkor azt kivesszük és felhasználjuk, majd esetleg egy másik értéket teszünk vissza helyette.

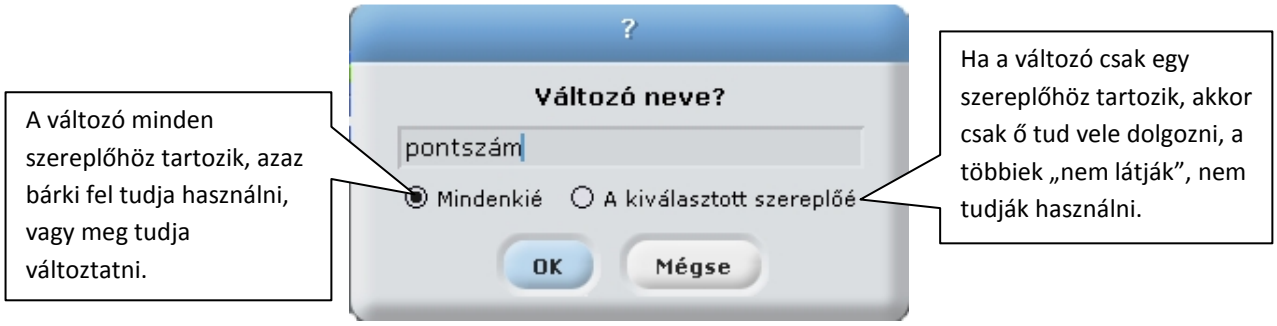


A játék elkészítése

- Szereplők: tátogó fej, színes labdák (könnyen megrajzolhatod, de le is töltheted őket)
- Változó: pontszám
- A szereplők mozgása, irányítása kissé eltér az eddigiektől. Erről a hajtás után, a véletlenszámok kapcsán olvashatsz.
- A játék lényege, hogy a fejet irányítva 30 másodperc alatt (ennek megvalósítását **Az óra** című leckében találod) minél több pontot kell szerezni a véletlenszerűen mozgó színes labdák elkapásával. A fekete labda érintése pontlevonással jár.

Pontszámolás

- A pontszám a játék indulásakor 0. Eggyel nő, ha a fej elkap egy színes labdát és eggyel csökken, ha hozzáér a feketéhez. Tehát a pontszám egy változó. Létrehozása: Változó létrehozása



- Ekkor megjelenik a pontszám változó és a vele kapcsolatos parancsok a parancskészlet változók csoportjában, a játéktérben pedig a változó „kijelzője”. Ez háromféle lehet, a jobb egérgombbal kattintva előjövő helyi menüből választható ki, hogy melyik jelenjen meg a játéktérben. (ha nem szeretnéd, hogy a játéktérben megjelenjen a változó, tüntesd el a pipát a változó neve elől a parancskészletben):

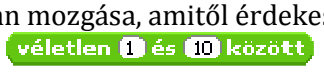
- Első lépésként állítsuk be, hogy a pontszám kezdetben 0 legyen. Ezt a feladatot adjuk pl. a játéktérnek (de mindegy, hogy melyik szereplő kapja ezt a feladatot).

- Ha valamelyik színes golyó hozzáér a fejhez, akkor a pontszám nőjön eggyel:

Ha hozzáér a fejhez, akkor eltűnik (mert a fej megeszi) és a pontszám *+1*-gyel változik (vagyis nő).

- A fekete golyó nem tűnik el érintkezéskor, így máshogyan kell megoldani a pontlevonást, különben több pontot is levonna, amíg áll. Ezért is van szükség a várakozásra, hogy addig a fej el tudjon menni a közeléből. Az érintkezés akár 1 másodpercig is eltarthat, ezalatt az idő alatt a program fut tovább – mindig érintkezést érzékel, így többször is levon egy pontot (pedig látszólag csak egyszer találkozik a két szereplő).
  A színes golyóknál ez azért nem probléma, mert az érintkezéskor azonnal eltűnnek és a láthatatlan szereplőkre nem működik az érintkezésvizsgálat. Tehát a fekete golyó feladatai:

Véletlenszámok

A játékban a labdák véletlenszerűen mozognak, a fej pedig egy kicsit „gyorsabb”, mint az eddigi főszereplőink. Most megtudhatod, hogyan lehet ezt megoldani. A véletlenszámok nagy segítséget jelenthetnek a játékok elkészítésében. Segítségükkel megvalósítható például az „ellenségek” kiszámíthatatlan mozgása, amitől érdekesebbé válik a játék. A Scratch-ben két szám közötti véletlenszámot adhatunk meg. 

Ez a parancs ebben az esetben egy 1 és 10 közötti számot ad és mindegyiket egyforma eséllyel, ugyanúgy, mintha egy 10 oldalú dobókockával dobnánk. Persze az 1 és 10 helyett más számokat (vagy akár változókat) is megadhatsz. Ha mindkét szám (a tartomány végpontjai, amelyből véletlenül választ a program számot) egész, akkor eredményül is egész számot kapsz. Ha valamelyik végpont nem egész szám, akkor a kapott szám sem lesz az.

A mozgások megvalósítása

- lássuk először a golyókat:

Az elfordulás is véletlenszerű. Ennek köszönhető, hogy nem mindig ugyanazt a pályát járja be a szereplő.

A játék kezdetétől a golyó folyamatosan mozog, de nem azonos sebességgel. Véletlenszerű a lépéshossza. Ha növelnénk a tartományt, akkor még kiszámíthatatlanabb lenne a mozgás, de ekkora játéktérben az már zavaró lehet.

- Ha ezek után lemásolod a szereplőt (hogy több golyó legyen a pályán), akkor minden golyó hasonló irányban fog mozogni, mivel ugyanarról a helyről indulnak és nem túl nagy tartományban mozog az elfordulásuk és a lépéshosszuk. Ez könnyen kiküszöbölhető, ha a golyók kezdőpozíciója is véletlen:

- A fej irányításához – az eddigiekkel ellentétben – billentyűérzékelést használunk. Nézzük, miben különbözik ez a „sapkás” megoldástól.

BILLENTYŰ-LENYOMÁS: Ekkor ha folytonosan lenyomva tartod a szóköz billentyűt, akkor nem fog folyamatosan végrehajtódni a hozzá rendelt esemény. Mindig vár egy picit, mielőtt megismétli (kivárja a billentyűismétlési holtidőt). Ráadásul több billentyű együttes lenyomása csak a legutolsót teszi érvényessé.

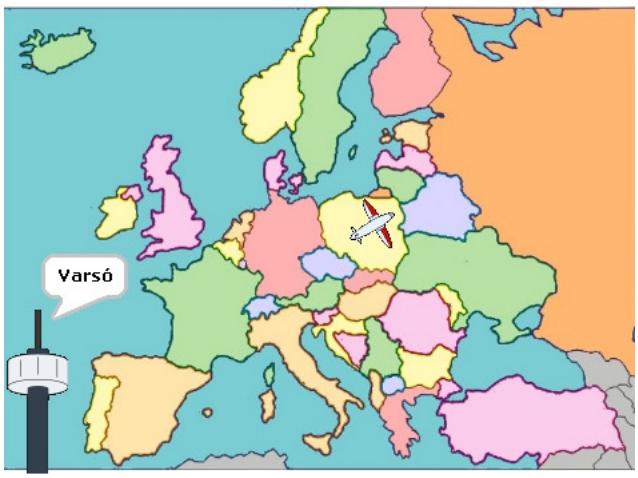
BILLENTYŰÉRZÉKELÉS: Itt nincs várakozás, folyamatosan lenyomott gomb mellett szünet nélkül zajlik az esemény ismétlése.
Az így megvalósított irányítás kiválóan alkalmazható például autóversenyes játékokban.

- Tehát a fej irányítása:

Mentsd el a projektedet hamihami néven!

8. Listák

A változók után ebben a leckében a listákkal ismerkedhetsz meg. A következő játékot készítjük el, melyben az irányítótorony által meghatározott város fölé kell repülni (ha odaértél, nyomd meg a szóköz billentyűt):



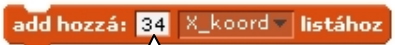
A listákban számokat vagy szövegeket tárolhatunk sorban egymás után. Új elem hozzáadásakor az elem a lista végére kerül, tehát ez a szerkezet hasonlít egy egyszerű bevásárló listához: ha eszünkbe jut valami, amit venni kell, a lista végére írjuk.

A listák létrehozása

Listákat is a változók csoportban hozhatsz létre a 

gombbal. A létrehozás ugyanúgy történik, mint változók esetén. A lista létrejötte után megjelennek a listakezelő parancsok:

A megadott szöveget hozzáadja a kiválasztott (ebben az esetben ez a Város) listához. Az új elem a lista utolsó helyére kerül.



Listához számot vagy akár egy változó aktuális értékét is hozzáadhatsz.

A törlés parancsban az első, az utolsó vagy az összes elem törlését választhatod az adott listából.

A választhatókon kívül törölheted pl. az i változó aktuális értékének sorszámán lévő elemet is.

Ennek a parancsnak a segítségével új elemet illeszthetsz a lista első, utolsó, vagy *egyik*, azaz véletlenszerű helyére.

Ide is illeszthetsz változókat. Ez a parancs a Város lista i-edik helyére szúrja be az j változó értékét. Pl. ha i=3 és j=5, akkor a parancs végrehajtása után a Város lista 3. eleme 5 lesz.

A beszúrás parancshoz hasonlóan le is cserélheted a lista elemeit.

Megadja a kiválasztott lista kiválasztott (első, utolsó, egyik vagy egy konkrétan megadott sorszámú elemét.

Megadja a kiválasztott lista hosszát, azaz elemeinek számát.

A listák elemeit felveheted, törölheted és módosíthatod a program futása közben a fenti parancsok segítségével, vagy a játéktérben megjelenő listakezelőben is:

A listához a + gomb segítségével adhatsz új elemet.

Az elemek egyszerűen begépelheted és később bármikor módosíthatod.

Azt, hogy melyik listakezelési módot érdemes használni, mindig az adott probléma határozza meg.

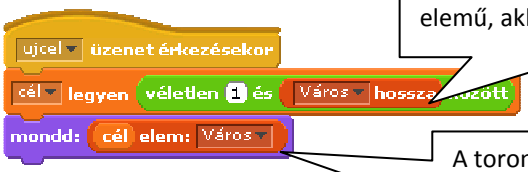
A játék elkészítése

- szereplők: repülő, irányítótorony (tőle kapja a repülő a következő célállomást), város (de ő csak hiba esetén jelzi célváros helyét). A hátteret és a szereplőket letöltheted a honlapról.
- változó: cél (a célállomás listabeli sorszámát tartalmazza)
- listák: város (az európai fővárosok nevét tartalmazza), X_koordináta (a városok x koordinátáját tartalmazza), Y_koordináta (a városok y koordinátáját tartalmazza)
- A játék: repülj a repülővel az irányítótorony által megadott város fölé, majd nyomd meg a szóköz billentyűt. Ha jó város fölé szálltál, akkor új feladványt kapsz. Ha rossz a találatod, akkor egy pillanatra felvillan a város helye, megmutatja, hová kell repülnöd.
- Először hozd létre a listákat. A városok nevét és a megadott háttérhez tartozó koordinátáikat megtalálod a honlapon. Figyelj arra, hogy a város neve, x és y koordinátái az egyes listák ugyanolyan sorszámú helyére kerüljenek. Ha túl hosszúnak találod a listát, elég csak 5-10 város adatait megadnod belőle, a játék működését ez nem befolyásolja.

Az irányítótorony feladatai

A kezdeti beállítások után *újcél* üzenetet küld, mivel szükség van úticélra.

A cél változó értéke egy 1 és Város hossza közötti véletlenszám. Ha a listában 37 város van, akkor ez a parancs egy 1 és 37 közötti számot ad, ha a lista 5 elemű, akkor 1 és 5 közötti számot.



Helyes üzenetet a repülőgép küld, ha eltaláltad a várost. Ilyenkor következhet egy új cél.

A torony megadja az úticélt: a Város lista cél változóban tárolt számú („cél-edik”) elemét.

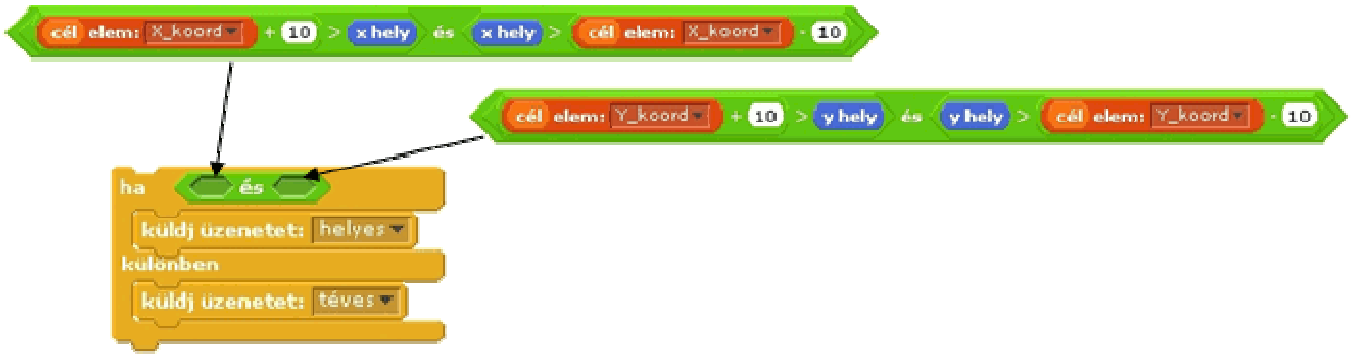
Ha a *téves* üzenetet küldi a repülő, akkor a cél változó értéke nem változik, az előző várost kell megtalálni.

A város feladatai

A város csak téves találat esetén jelenik meg egy másodpercre a célállomás helyén. Ezért x-koordinátája az X_koord lista „cél-edik” eleme, y-koordinátája pedig az Y_koord lista „cél-edik” eleme lesz.

A repülőgép feladatai

A repülőgép irányítása tetszőlegesen megvalósítható. Egy feltételre kell figyelni: amikor a játékos megnyomja a szóköz billentyűt, akkor előfordulhat, hogy bár a repülő a megfelelő város fölött van, mégis téves üzenetet küld a program. Ez azért van, mert nagyon nehéz pontosan eltalálni a játéktér egy bizonyos koordinátájú pontját. A játék akkor is élvezetes marad, ha 10 pixelnyi "szabadságot" adunk a cél eltalálásakor:



Mentsd el a projektedet repcsi néven!

Festőablak

A festőablakban saját szereplőket vagy háttereket rajzolhatsz, illetve módosíthatod a már meglévőket. Az ablakot több gombbal is előhívhatod:

ÚJ SZEREPLŐ FESTÉSE
a játéktér alatti gombok közül az első

A programozási tér hátterek (játéktér esetén) vagy jelmezek (szereplő esetén) fülén.

Az ablak felépítése

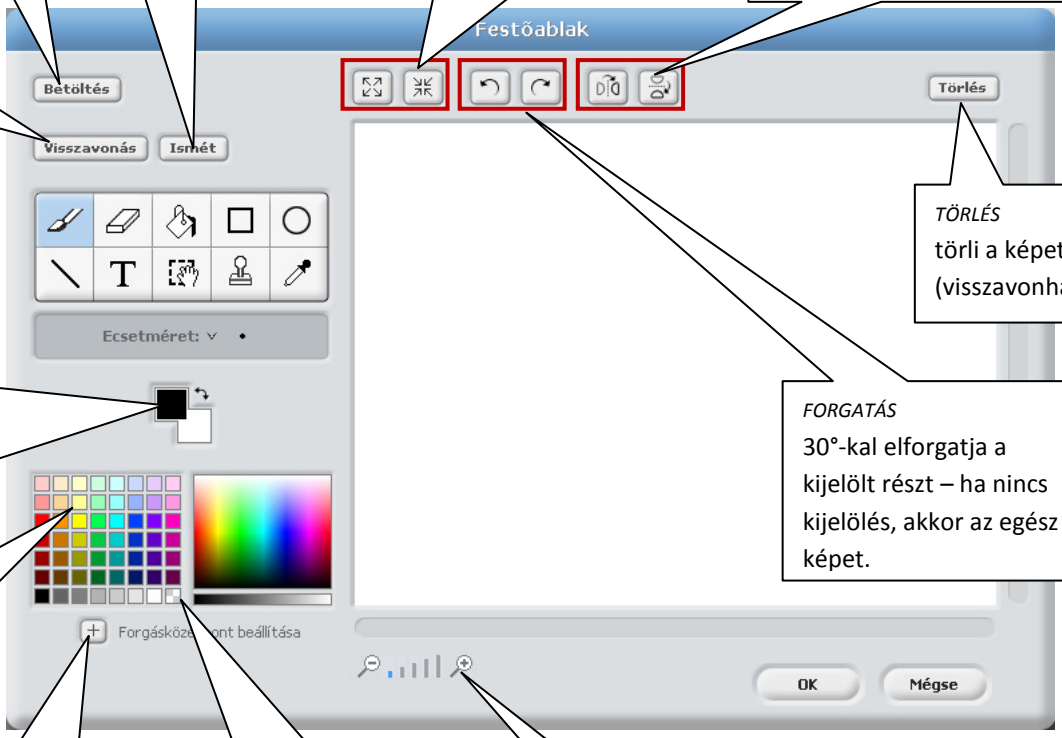
BETÖLTÉS
betölthetsz egy már létező képet.

ISMÉT
megismétli az utoljára visszavont lépéseket.

NAGYÍTÁS, KICSINYÍTÉS
átméretezi a kijelölt részt a rajzon. Ha semmi nincs kijelölve, akkor az egész képet megváltoztatja.

TÜKRÖZÉS
vízszintes vagy függőleges tengelyre tükrözi a az egészképet vagy a kijelölt részt.

VISSZAVONÁS
az utolsó 9 lépést lehet visszavonni



TÖRLÉS
törli a képet (visszavonható)

RAJZOLÓSZÍN, HÁTTÉRSZÍN
az aktuális rajzoló- és háttérszín. Jobb gombbal kattintva lehet váltani közöttük.

FORGATÁS
30°-kal elforgatja a kijelölt részt – ha nincs kijelölés, akkor az egész képet.

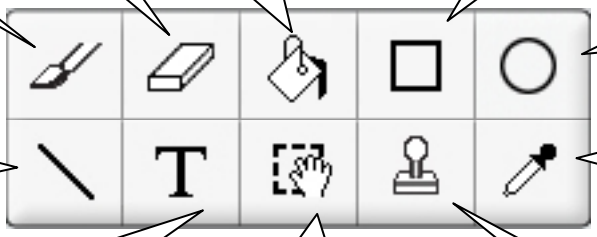
PALETTA
innen választható ki a rajzoló- és a háttérszín

FORGÁSKÖZÉPPONT
az a pont, amely körül elforgatjuk a szereplőt és az a pont, amely a tollhegy lesz a tollat le parancs esetén.

ÁTLÁTSZÓ „SZÍN”
Egy speciális szín. Főleg szereplők rajzolásakor alkalmazzuk. Amit átlátszóra színezünk, nem takarja el a mögötte lévőket.

SKÁLA
a nézet változtatására használható: közelebbről vagy messzebbről nézhetjük a rajzot. Tehát a tényleges mérete nem változik, csak könnyebb rajzolni az apró részleteket.

A rajzeszközök



ECSET
szabadkézi rajzokhoz

RADÍR
átlátszó „színnel” törli a rajzokat.

KITÖLTÉS
előtérszínnel vagy színárnyalattal tölti ki az egybefüggő területeket.

TÉGLALAP
kitöltött vagy körvonalas téglalap rajzolására.

ELLIPSZIS
kitöltött vagy körvonalas ellipszis rajzolására.

VONAL
vonalak rajzolására. A vonal vastagsága az ecsetmérettel állítható.

PIPETTA
A rajzról pipettával kiválasztott szín lesz az aktuális rajzolószín

SZÖVEG
ezzel az eszközzel írhatsz szöveget a rajzra. A szöveg új rétegen jelenik meg. Egy rajzon csak egy szöveg szerepelhet.

KIJELÖLÉS
területek kijelölésére (ezeket lehet másolni, tükrözni, forgatni, stb.)

BÉLYEGZŐ
a kijelölt területtel lehet „nyomdázni”

Forgási stílusok

A szereplők a 

parancs teljesítése után néha „fejre állnak”, máskor viszont egyszerűen megfordulnak. Ez a forgási stílusuktól függ, amelyet a programozási tér fölött állíthatsz be.

Egy kísértet mozgásán keresztül láthatsz példát a stílusokra:

A kísértet feladata:

Viselkedése a forgási stílusától függően:

Minden irányba elfordul

A szereplő ilyenkor mindig a haladási iránya felé néz. (A haladási irányát a kék pálcika jelzi).

Csak balra és jobbra néz

A szereplő ekkor mindig csak jobbra vagy balra néz. Ha jobbra száll (és közben akár emelkedhet vagy süllyedhet is), akkor jobbra néz, ha pedig balra repül, akkor balra fordul el.

Sosem fordul el

A szereplő mindig az eredeti irányba néz, és nem is fordul el.

Kinézet

A leckékben már megismerkedhettél a **parancskészlet** Kinézet csoportjának egy részével. Ez a kiegészítés összefoglalja a velük kapcsolatos tudnivalókat.

Ezekkel a parancsokkal lehet a szereplő kinézetét meghatározó jelmezeket váltani. A vált jelmezt a jelmezek sorában következő jelmezre vált.

Megjeleníti egy szövegbuborékban a szereplőnek adott szöveget (a „Szia” helyére változót vagy listaelemet is illeszthetünk, akkor azok értékét mondja a szereplő). A szövegbuborék az első esetben a megadott ideig látszik, a második esetben újabb Mondd parancs kiadásáig. (Akkor tűnik el, ha a parancsot „üresen” adod ki.)

A Mondd parancshoz hasonlóan működik, csak itt szövegbuborék helyett gondolatfelhő jelenik meg a szereplő fölött.

Ezekkel a parancsokkal a szereplő méretét változtathatod. Az első esetben a megadott mértékkel (csökkentheted is, ha ez a szám negatív). A második esetben azt adhatod meg, hogy az eredeti méretnek (ez a 100%) hány százaléka legyen az új.

A szereplők lehetnek láthatatlanok is (tűnj el). Ilyenkor a többi szereplő nem érzékeli őket. A megfelelő pillanatban láthatóvá teheted őket a jelenj meg paranccsal.

A szereplők külön rétegeken helyezkednek el. Aki előrébb van, az eltakarhatja a mögötte lévőket. Ha azt szeretnéd, hogy egy szereplőt ne takarhasson el senki, akkor használd a kerülj legelőre parancsot, vagy az őt takarókat küldd néhány szinttel hátrébb.

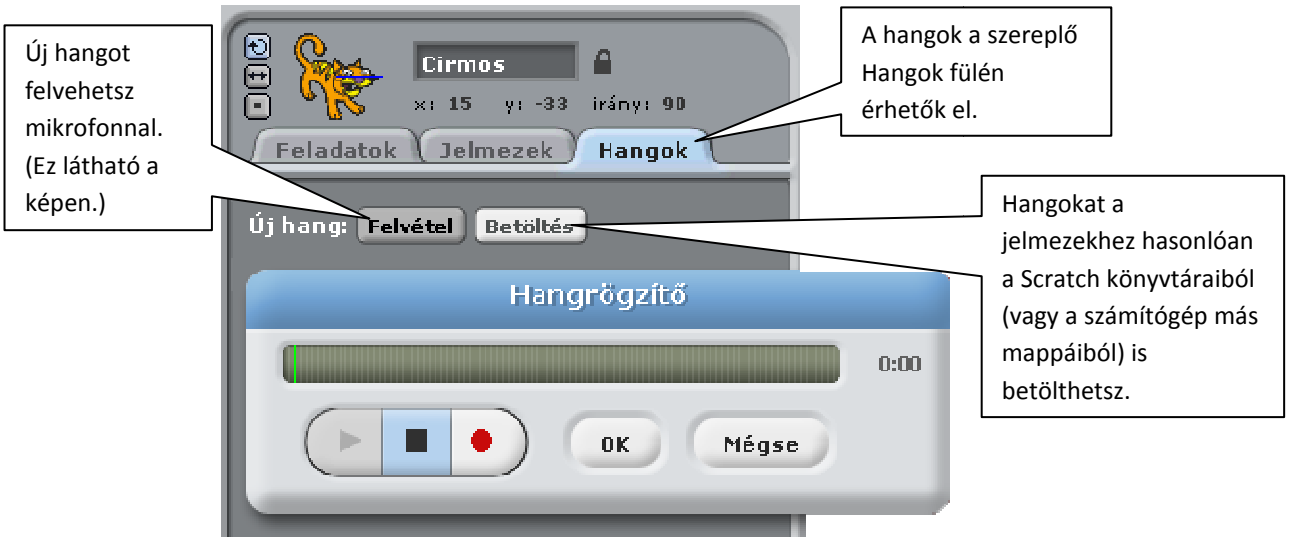
Ezekkel a parancsokkal különböző grafikus hatásokat adhatsz a szereplőkhöz, vagy törölheted ezeket a hatásokat. (Próbáld ki mindet!)

Hangok

Felvett hangok

A szereplők nemcsak szövegbuborékkal jelzett „hangokat”, hanem igaziakat is tudnak produkálni. Ezt ráadásul kétféleképpen is megtehetik.

A hangok kezelése hasonló a jelmezekéhez. A szereplőhöz tartozó hangok a Hangok fülön találhatók. Ez a szereplő létrehozásakor általában üres, tehát a szereplő még nem tud hangot adni.



Új hangot felvehetsz mikrofonnal. (Ez látható a képen.)

A hangok a szereplő Hangok fülén érhetők el.

Hangokat a jelmezekhez hasonlóan a Scratch könyvtáraiból (vagy a számítógép más mappáiból) is betölthetsz.

A hang ezután bekerül a szereplő hangjai közé.

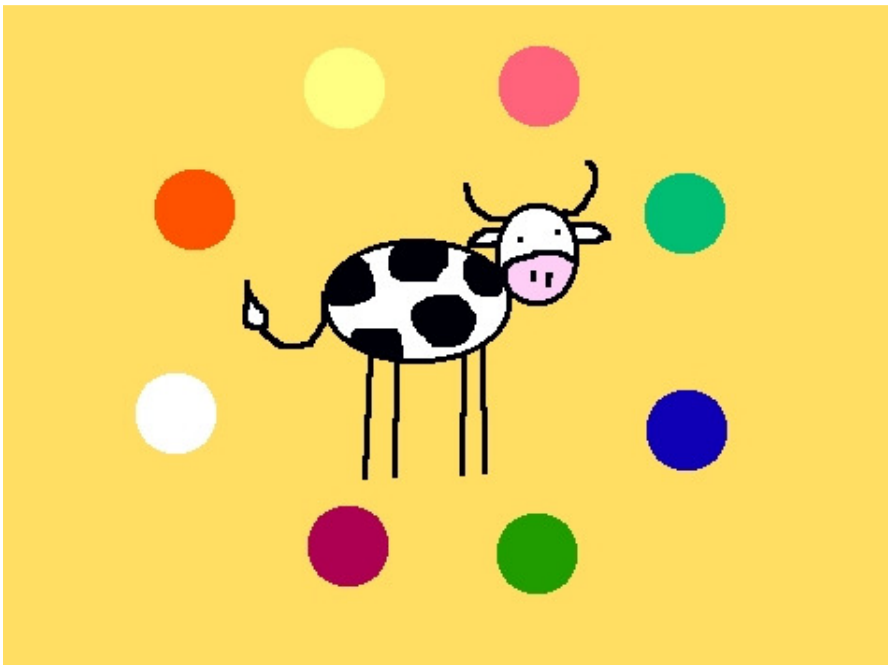
A hangokat a projektben a Hang parancscsoport parancsaival lehet kezelni. A Cirmos macskához tartozó Kitten hangot például így lehet előhívni:

Ez a parancs elkezdi lejátszani a hangot, majd végrehajtja az utána illesztett parancsokat, miközben a hang még esetleg szól.

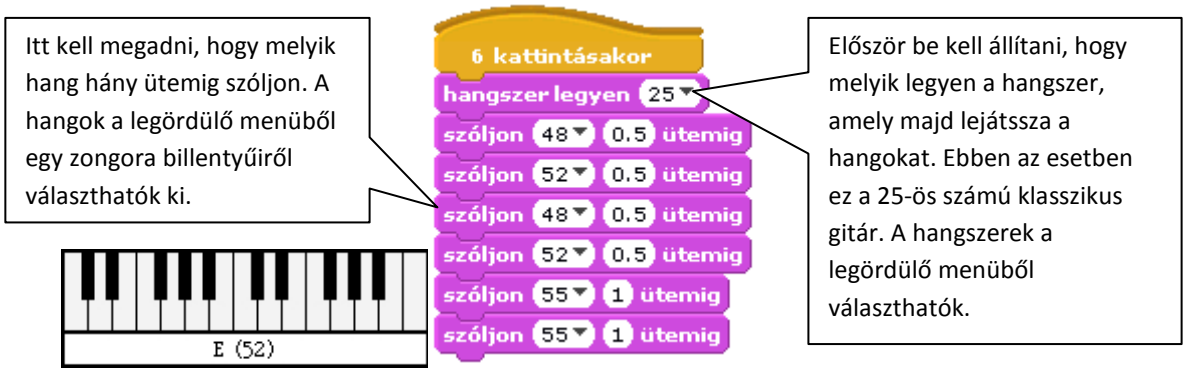
Ez a parancs előbb lejátssza a hangot, és csak azután hajtódhatnak végre az utána illesztett parancsok. Tehát először nyávog a macska és csak azután láp 10 lépést.

Beépített hangszerek

A szereplők nemcsak beépített hangokat tudnak lejátszani, hanem különböző hangszerekkel játszott dallamokat is. A következő projekt például így készült:



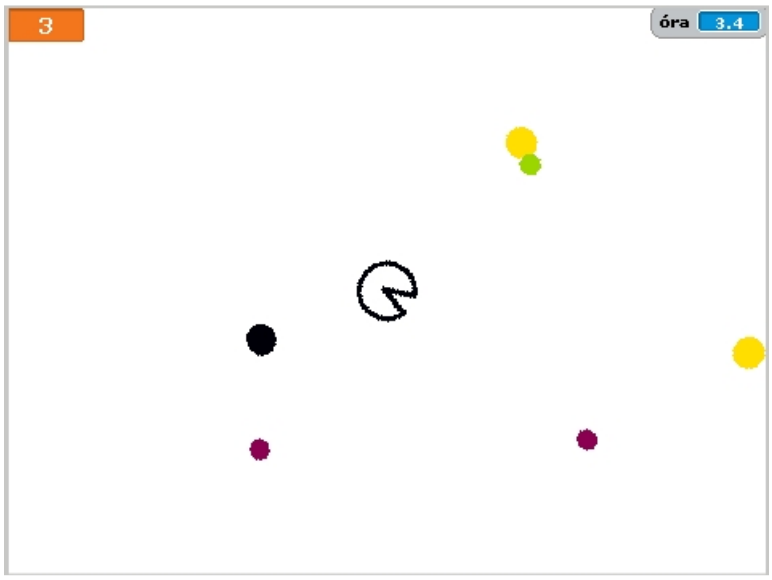
A színes gömbökre kattintva különböző hangszereken csendül fel a mindenki által jól ismert Boci, boci tarka… első néhány hangja. A labdák feladata :



A hangerő és a tempó parancsokkal változtatható, de általában megfelelőek az előre beállított értékek.

Az óra

A 7. leckéhez készített játék pontosan 30 másodpercig tart. Ehhez a Scratch beépített óráját kell használni. Lássuk, hogyan! Emlékeztetőül a játék:



A Scratch órája

Az óra a Scratch indításától kezdve folyamatosan jár, a megnyitás pillanatától eltelt időt mutatja másodpercekben. A parancskészlet érzékelés csoportjában a neve (óra) melletti szürke négyzet kattintásával jeleníthető meg a játéktérben, vagy tüntethető el onnan.

A megjelenítés a változókhoz hasonlóan kétféle lehet: óra 1.0 valamint 1.7

Az órát egyetlen paranccsal lehet befolyásolni: nullázd az órát. Ez értelemszerűen nullára állítja az óra értékét, azaz újrakezdi a számolást. Az időmérést használó játékok elején ezt érdemes megtenni:

INICIALIZÁLÁS
a játék kezdetén érdemes egy külön sapkán beállítani a kezdeti értékeket. Például hogy az óra 0-ról induljon és persze a pontszám is 0 legyen.

Az óra pillanatnyi értékének nemcsak a megjelenítése, hanem a kezelése is hasonló a változókéhoz: a játék akkor ér véget, ha az óra értéke nagyobb, mint 30.

Azért nem azt vizsgáljuk, hogy az óra=30 feltétel teljesül-e, mert a programnak egyszerre sok feladatot kell elvégeznie, és lehet, hogy abban a pillanatban, amikor az óra értéke pontosan 30 másodperc, a program éppen nem azzal a vizsgálattal foglalkozik, amelyben az óra=30 feltételt ellenőrizzük, így nem állít le mindent.

1. Ismerkedés a Scratch környezettel 1
A Szereplők 1
Új háttér betöltése 2
2. Megmozdulnak a szereplők 3
A játék elkészítése 3
Variáció 5
3. Ismétlődések 6
Ciklusok 6
A program elkészítése 6
Feladat 7
4. Feltételek 8
A játék elkészítése 8
Egymásba ágyazás 10
5. Üzenetek 11
Ismerkedés az üzenetekkel 11
Vége a játéknak 13
6. Rajzoljunk 14
A program elkészítése 14
7. Változók 16
Változó 16
A játék elkészítése 17
Pontszámolás 17
Véletlenszámok 18
A mozgások megvalósítása 19
8. Listák 20
A listák létrehozása 20
A játék elkészítése 21
Az irányítótorony feladatai 22
A város feladatai 22
A repülőgép feladatai 22
Festőablak 23
Az ablak felépítése 23
A rajzeszközök 24
Forgási stílusok 25
Minden irányba elfordul 25
Csak balra és jobbra néz 25
Sosem fordul el 25
Kinézet 26
Hangok 27
Felvett hangok 27
Beépített hangszerek 28
Az óra 29
A Scratch órája 29

Ellenőrző feladatsor

Feladatsor a Scratch tananyaghoz

1.) Írd le, hogyan mozog a helikopter, ha a következők a feladatai!

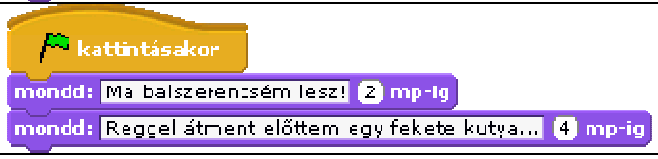
2.) Írd le, mit csinál a rendőr, ha a következők a feladatai!

3.) Írd le, mit csinál a szellem, ha a következők a feladatai!

4.) A varázsló közeledik a táncoló gyerekhez. Amikor hozzáér, a gyerek eltűnik. Melyek a szereplők feladatai? Karikázd be a helyes válasz betűjelét!

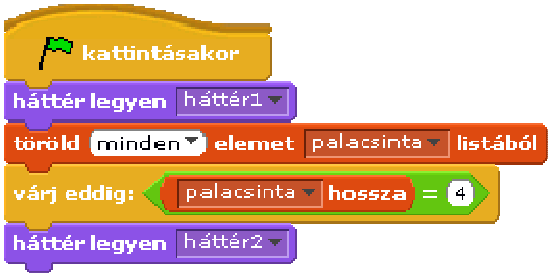
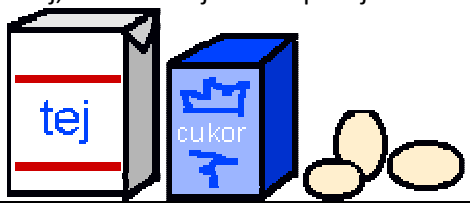
	A	B	C
			
			

5.) Két macska beszélget: „Ma balszerencsém lesz!” – mondja az egyik. „Miért?” – kérdi a másik. „Reggel átment előttem egy fekete kutya…”
Melyek a macskák feladatai? Karikázd be a helyes válasz betűjelét!

	Egyik macska	Másik macska
A		
B		
C		

6.) Mi történik ebben a projektben?

A projektben négy szereplő található feladatokkal, sőt a játéktérnek is van két jelmeze és feladata.

<table>
<tr>
<td>A játéktér jelmezei:
 
</td>
<td>A játéktér feladatai:
 
</td>
</tr>
<tr>
<td>A liszt szereplő jelmeze:
  </td>
<td>A liszt szereplő feladatai:
 
</td>
</tr>
<tr>
<td>A tej, cukor és tojás szereplők jelmezei:
 
</td>
<td>E három szereplőnek majdnem ugyanaz a feladata, mint a lisztnek, annyi különbséggel, hogy az add hozzá parancsban a saját nevük szerepel a liszt helyén.</td>
</tr>
</table>