Vizuális programozás oktatása középiskolákban

Csapó Gábor

glerikud.strawhat@gmail.com
DE IK

Absztrakt: Napjainkban egyre nagyobb szerepet kap a programozás az informatikai mindennapokban. Azonban a diákok jó része nem rendelkezik a szükséges algoritmizálási készségekkel az iskola elvégzése után. Cikkemben ezt a problémát vizsgálom, illetve a vizuális programozás felől megközelítve egyfajta megoldást is kínálok. Az egyre népszerűbb és dinamikusan fejlődőt vizuális fejlesztőkörnyezetek segítségével úgy adható át az algoritmizálás, hogy a tanulók azt élvezetes és kreatív folyamatként élik meg, amely azonnal visszajelzéssel szolgáltat a munkájukról. Erre épül az általam kidolgozott oktatási tervezet, amely feladatokon keresztül vezeti bele a diákokat a vizuális programozásba és az algoritmizálásba.

Bevezetés

Manapság egyre több fiatal érdeklődik a számítógépes szoftverek fejlesztése iránt, szeretnék ötleteiket gyakorlatba átültetni, megvalósítani őket. Sajnos azonban többségük számára a megfigyelések alapján nem járható út a különböző programozási nyelvek kellő mélységű elsajátítása, mivel leginkább bonyolultnak, semmitmondónak, vagy átláthatatlannak találják ezeket, és nem látják a lehetséges utat, ami majd a kész produktumukhoz vezet. Ennek ellenére azonban nem elhanyagolható a programozás oktatása, hiszen az általa fejlődő, fejleszthető képességekre szüksége van a tanulónak a mindennapjaiban.

Korunkban egyre nagyobb teret hódítanak azok a fejlesztői környezetek, amelyek segítségével a felhasználó anélkül a képernyőre viheti ötleteit, hogy akár egy sor programkódot írnia kellene. A kód kézi írása helyett egy grafikus felületet alkalmazva párosítja össze az objektumokat az eseményekkel és az azokhoz kapcsolódó utasításokkal. Az ilyen szoftverek segítségével a fejlesztők könnyedén és gyorsan valósíthatják meg programbeli elképzeléseiket még akkor is, ha számukra a programozási nyelvek többsége átláthatatlan vagy nem áll rendelkezésükre idő, lehetőség azok megtanulására. Ilyen szoftverek közé tartozik a dinamikusan fejlődő és egyre népszerűbb Construct 2 (3, 2009), a Mark Overmars, egy egyetemi oktató által fejlesztett Game Maker (4, 2007), valamint számos konkurens alkalmazás is: Stencyl (55, 2011), GameSalad (6, 2009), Game Editor (7, 2011).

Ezen programok oktatásának bevezetése megkönnyítené sok diák számára a szoftverfejlesztés mechanizmusának átlátását és emellett pozitívabb kép alakulna ki bennük a programozást illetően, így a későbbiekben (például felsőfokú tanulmányaik során) tanult programozási módszerek könnyebben elsajátíthatóak lennének. Mivel ezek a szoftverek vizuális felületet alkalmaznak a programozás lebonyolításához, tökéletesen alkalmasak arra, hogy könnyű átláthatóság mellett alakuljon ki az algoritmikus készség. A programozás során az egyes utasításokat „fogd és vidd” módszerrel választhatják ki a diákok, így a kód szintaktikai pontossága helyett a figyelem az algoritmizáláson van.

Célom, hogy ismertessem ezt a szoftverkategóriát, valamint napjaink egyik leghatásosabb és legdinamikusabban fejlődő vizuális nyelvét és annak fejlesztőkörnyezetét, és egy olyan oktatási terv kidolgozása, amely a vizuális programozást széles körben prezentálja a tanárok, a diákok számára, amellyel az alapvető programozási módszereket elsajátítják, amellett, hogy saját munkájuk eredményét is kézzel foghatónak érzik.

Iskolában használt programozási környezetek

Az informatikai eszközök és az internet széles körű elterjedésével szükségessé vált először az informatikai alapismeretek, később már mélyebb szintű informatikai témakörök oktatása. Jelenleg a modern informatika tananyagok szerves része a programozás általános- és középiskolákban egyaránt (8, 2012).

Az algoritmikus gondolkodás kialakítására és fejlesztésére általánosan öt módszert alkalmaznak:

- A Logo szoftvereket (Comenius és Imagine Logo), amelyek a Logo programozási nyelvre épülve oktatási céllal lettek kifejlesztve és kezdő szinten alkalmasak a programozási folyamatok bemutatására. Elsősorban általános iskolákban alkalmazzák, melyből a diákok gyorsan tanulnak és látványosan. Egyszerűsége miatt idősebb korosztályok számára nem ideális. A Logo másik hátránya, hogy szinte kizárólagosan iskolai felhasználásra készült, az algoritmikus készség és kreativitás fejlesztésére. Iskolán kívül, a valós programozásban nem játszik szerepet, és mint ilyen a diákok nem tekintik olyan programozási nyelvnek, amit érdemes megtanulni.
- A Lego robotokat, amelyek megépítésük után egy egyszerű felületen, logikai blokkok segítségével vezérelhetőek. Ez a módszer is főként általános iskolákban terjedt el a korosztálybeli érdeklődés miatt.
- Hagyományos programozási nyelveket, amelyek már kifejlettebb logikai készségeket igénylenek. Ezért főként középiskolai tananyagban találkozhatunk velük. Sajnos azonban ez helyenként idejétmúlt nyelv oktatását jelenti (1, 2000) (2, 2002), amelyet a diák szinte biztos, hogy nem fog alkalmazni a későbbiekben. Emellett a programozási nyelveket sok diák nem érti, mivel bonyolultnak találja és elsődleges célja nem annak élvezete, vagy lehetőségeinek megtanulása, kiaknázása, hanem a tantárgy teljesítése.
- Táblázatkezelő programokat, amelyek néhány iskolában teljesen leváltják a hagyományos programozási nyelveket. Ennek a módszernek nagy előnye az időtakarékosság és a programozási készségek kialakításával párhuzamosan járó táblázatkezelés elmélyülése. Hátránya azonban, hogy nem alakul ki a diákban annak az elképzelése, hogy a megszerzett tudást a táblázatkezelőn kívül hol tudja majd felhasználni.
- Mobil programozást, amely során nem hagyományos PC-re, hanem a trendeknek megfelelően, mobil eszközökre készítenek alkalmazásokat a diákok és ezen keresztül sajátítják el az algoritmizálást. Ennek hátránya, hogy a legtöbb esetben hagyományos programozási nyelveket használnak, amely továbbra sem a legkönnyebben befogadható a diákok számára. Előnye azonban, hogy ezek az alkalmazások már látványosnak készülnek, így a tanulók könnyebben meg tudják ragadni a munkájuk eredményét. Megjegyezném, hogy a Construct 2 segítségével is lehetőségünk van mobil alkalmazásokat fejleszteni a napjainkban elterjedt mobil operációs rendszerekre is.

A programozás oktatásának egyértelmű célja az algoritmikus gondolkodás fejlesztése és annak későbbi tanításának megkönnyítése, valamint egy alapvető tudás átadása, amelyet a tanulók a későbbiekben alkalmazni tudnak.

Logo nyelvek

A Logo nyelvek esetében a diákok könnyen szembesülnek a munkájuk eredményével grafikus módon. A beírt utasítások okozta változásokat azonnal láthatják a képernyőn, és ez pozitívan befolyásolja őket. A szoftver egy játékos grafikus környezetet alkalmaz, így a fiatalabb diákok számára ideális, mert nem bonyolódnak bele, zavarodnak össze komplexebb menüpontok és bonyolult felhasználói felületek láttán. Korábban a legtöbb általános iskolában Pascal programozási nyelvet oktattak bevezetésképpen a programozás világába a diákoknak. Azonban a Comenius Logo használata és tanítása gyorsabb eredményeket mutatott.

A tanulói réteg korosztályából kiindulva a diákok egy része nem hajlandó komolyan venni a feladatokat, vagy nem képes az ehhez szükséges algoritmikus gondolkodásra. Az önálló feladatok megoldása (természetesen azok nehézségi szintjétől függően) nehézségekbe ütközik, esetenként szövegértési problémák is felmerülnek. A tanultakat gyakorlatban alkalmazni csupán egy kisebb réteg képes önállóan. A diákok érdeklődésének növelésével növelhető a tanórák teljesítménye, ezért a tanár feladata olyan munkák összeállítása, amelyek a csoporthoz igazodva megragadják a tanulók figyelmét és ezért érdeklődéssel hajtják végre azokat. Ezek ellenére azonban a szoftver és a nyelv egyszerűsége miatt már az első órán szembesülnek a diákok munkájuk eredményével: egyszerű mozgatást és rajzolást valósítanak meg.

Magas szintű programozási nyelvek

Középiskolában már egy fokkal könnyebb a programozási nyelvek oktatása. A tanulók fejlettebb és érettebb gondolkodással rendelkeznek és könnyebb őket ráhangolni az algoritmikus feladatmegoldásra. Ennek ellenére az érettségit tett tanulók által nyújtott teljesítmény mégis elmarad az elvárttól.

Ennek oka elsősorban a szűk óraszámra tehető, ugyanis a programozás oktatására rendelkezésre álló óraszám mellett (az éltanulók kivételével) még a nyelv alapjai sem sajátíthatóak el az előírt mélységben. A másik oka a diákok alulteljesítésének a programozási nyelvek érdektelensége számukra. Egy átlagos tanuló nem találja kellően érdekesnek egy parancssoros felületen programkódot írni, különösképp, ha az informatika nem is tartozik a kedvelt tantárgyai közé. Az így kialakuló lelki eltávolodás eredményeképp még kevesebb algoritmikus gondolkodást fog elsajátítani, holott arra szüksége van a mindennapi élet során. Bizonyos szakiskolákban felmerülhet az a helyzet is, hogy az informatika olyan minimális hangsúllyal bír az iskola szakterülete mellett, hogy a tanárok és tanulók is közös véleményen vannak, miszerint annak oktatása majdhogynem már felesleges. A kerettantervben a szakiskoláknál csak választható tantárgyként szerepel az informatika (9, 2013).

Vizuális programozás

Construct 2 ismertetése

Az általam választott szoftver a Microsoft Windows környezetben futó Construct 2. A program segítségével alkalmazások és játékszoftverek készíthetőek előzetes programozási nyelvek ismerete nélkül, a manapság egyre népszerűbb HTML5 technológiára támaszkodva.

Egy gyakran frissülő szoftverről beszélhetünk, amely lépést tart a legújabb technológiákkal, figyelembe veszi a programmal dolgozó felhasználók igényeit, a konkurencia közül a legkevésbé korlátozott, valamint elérhető árkategóriában van az átlag felhasználó számára. Sajnos általános hiba a hasonló szoftverek tekintetében, hogy (ellentétben a hagyományos programozási nyelvek alkalmazásával) alapvető korlátokba ütközünk, amelyet nem tudunk áthidalni a fejlesztőkörnyezet és annak motorja okából. Más szavakkal megfogalmazva: olyannyira le van egyszerűsítve a fejlesztési folyamat, hogy nem nyílik lehetőségünk bizonyos funkciók implementálására. A másik véglet ezzel szemben a túlbonyolított környezetek, melyeket egyszerűen felfoghatunk úgy is, mint egy hagyományos programozási nyelv kiszépített és maximálisan felhasználóbaráttá tett fejlesztői környezetét. A Construct 2 ezen két véglet között képviseli a középutat egyszerű és könnyen érthető fejlesztési folyamattal és minimális korlátozásokkal. Azért is választottam ezt a szoftvert feldolgozásra, mert tudom, hogy egy-egy programkód megírása igen hosszadalmas művelet, különösképp ha egy nagyobb projektről van szó. Nem is beszélve az adott programozási nyelv elsajátításáról, valamint a gyakorlati tapasztalat megszerzésére fordított időről. Célom egy olyan alternatíva bemutatása, amely segítségével bármely, kicsit tapasztaltabb számítógép felhasználó fejleszthet alkalmazásokat programozási nyelvek ismerete nélkül, viszonylag rövid idő alatt, látványosan. Illetve egy ilyen lehetőségeket felsorakoztató szoftver oktatásának megtervezése, hogy ennek a lehetősége már fiatal korban eljusson az érdeklődőkhöz.

A szoftver HTML5 alapú motorra építi az elkészült projekteket. Ebből adódóan az elsődleges célplatform a Web (habár nem korlátozódik csupán erre a területre). XML szabványt alkalmazva tárolja a projektek adatfájljait, így azok a program nélkül is szerkeszthetőek és módosíthatóak hozzáértő felhasználók számára (habár lefordításukhoz természetesen szükség van a Construct 2 környezetre). A program ingyenesen letölthető (11, 2014) néhány korlátozással. Az ingyenes változat szabadon terjeszthető, oktatásban alkalmazható, viszont a vele készített termékek kereskedelmi forgalomba nem hozhatók.

A szoftver rendelkezik néhány előre elkészített viselkedésmintával, amelyeket az objektumokhoz hozzárendelve a fejlesztési folyamat jelentősen felgyorsul. Szemléltetve a funkciót arról van szó, hogy egy adott elem viselkedésének a leprogramozását kikerüljük és helyette a már előre elkészítettet szabjuk testre. Ezzel időt és kódot takarítunk meg, nem is beszélve arról, hogy a program ezt nem számolja fel eseményként, így az ingyenes licensz korlátozásában sem kerülünk közelebb a határértékhez.

A szoftver igazi különlegessége az eseménylapok kezelésében és a vizuális programozás folyamatában rejlik. Hierarchikus felépítést alkalmaz és követi az alapértelmezett futtatási sorrendjét az utasításoknak (fentről-lefele) az adott eseményen belül. Első lépésként meg kell határoznunk egy eseményt (event), amely teljesülése esetén az ahhoz hozzárendelt utasítások lefutnak. Egy eseményen belül több feltételt is megadhatunk, így kialakítva AND vagy OR kapcsolatokat. Ezt követően tudunk parancsokat adni a programnak. Minden esemény és utasítás valamilyen objektumhoz kell, hogy kapcsolódjon. Példa: Amennyiben a bal egérgombbal kattintottunk,

akkor a változó értéke növekedjen meg 10-zel. Ehhez a példához szükségünk van az egér objektumra, illetve egy szám típusú változóra. Emellett az eseménylapok rugalmasan módosíthatóak. Az egyes utasítások vagy események objektumait kicserélhetjük, vagy a megadott értékeket átírhatjuk a teljes utasítás szerkesztése nélkül. Az egyes elemeket „fogd és vidd" módszerrel könnyedén mozgathatjuk.

A Construct 2-vel történő vizuális programozás során megvalósulnak a programozás szekvencialitás, szelekció és iteráció alapelvei. Az utasítások végrehajtása egymást követő, szekvenciális sorrendben történik. Az eseményblokkok szelekció révén futtatják le a beléjük foglalt utasításokat. Valamint lehetőségünk van a hagyományos programozási nyelvekből ismert feltételes ciklusok, iterációk használatára.

A programfejlesztés során is fontos elem, a Construct 2 esetében pedig kiemelkedően fontos a készülő projekt előnézete és tesztelése. Így nemcsak a hibásan működő elemekre kapunk rálátást, de szembesülünk eddigi munkánk eredményével is, amely (különösképp a fiatal fejlesztők esetében) további lendületet adhat a munka folytatásához. Az előnézet minden esetben az egyik telepített böngészőprogramban történik.

Amikor a kész projektünket exportálni szeretnénk, létrehozunk egy a környezet használata nélkül is futtatható alkalmazást a kiválasztott platformra. Sok esetben amikor egy fejlesztő több platformot céloz meg, akár a teljes forráskódot át kell írnia, hogy mindegyikre meg tudja azt jelentetni. Szerencsére a HTML5 rugalmassága miatt a Construct 2 esetében erről nincs szó, mivel elég néhány apróbb módosítást elvégeznünk a programunkon, hogy alkalmazkodni tudjon a platformok közti differenciálódáshoz.

A program kiemelkedően jól kidolgozott dokumentációval (10, 2014) rendelkezik, melyet nem csupán tartalombeli bősége és hasznossága miatt értékelhetünk, hanem elérhetőség és nyitottság szempontjából is. Emellett a szoftver oktatásbeli használatát is támogatják. A hivatalos közösségi fórum külön részleggel rendelkezik, ahol a tanárok az ezzel kapcsolatos tapasztalataikat megbeszélhetik, megoszthatják egymással (12, 2014).

Vizuális programozás előnyei

A bevezetőben felsorolt problémák egyik megoldása lehet a vizuális programnyelvek bevezetése középiskolai oktatásba. A tanulók egy egyszerűen elsajátítható és látványos (a munkájuk eredményét azonnal visszatükröző) módszert tanulnának, amely nemcsak alkalmas az algoritmikus készségek fejlesztésére, de visszatükrözi a hagyományos programozási nyelvek esetében tanult elemeket is (13, 2012), így a későbbiekben erre a tudásra támaszkodva szintén könnyebb más nyelvek elsajátítása.

A vizuális programozás a hagyományos programozási nyelvek kategóriáját egészítené ki, a Construct 2 szoftver oktatásával. Egy innovatív és új módszerről van szó, amely a programozást egy alternatív oldalról közelíti meg. Olyan szerkesztőfelületet alkalmaz, amelyben „fogd és vidd" módszerrel összeállíthatunk „bármilyen" 2D-s játékszoftvert forráskód hagyományos gépelése nélkül. Ezért oktatásbeli hatékonysága is magas, hiszen a tanulóknak nem kell bonyolult kódelemeket elsajátítaniuk ahhoz, hogy továbbfejlődjön az algoritmikus készségük, valamint kedvezőbb számukra a tény, hogy játékszoftvert fejlesztenek a nem túl népszerű iskolához között problémák megoldása helyett. Emellett nem elhanyagolható a HTML5 technológia modernitása sem, hiszen egy dinamikusan fejlődő, nyílt és cross-platform rendszerről van szó, valamint, hogy a Construct 2 ingyenesen letölthető és használható az oktatásban.

A Scirra cég programjával szerzett tudást a későbbiekben is felhasználhatják a diákok. Akár az oktatott fejlesztőkörnyezetet használva is piacképes szoftverek készíthetőek több platformra is, de a szerzett algoritmikus készség mindenképp kamatoztatható (különösen napjainkban ahol szinte minden ember érintkezik valamilyen digitális eszközzel).

Oktatási terv

A Construct 2 környezethez kidolgozott oktatási terv fejezetekre bontva vezeti be a tanulókat a szoftver használatába és ezzel együtt a komplexebb projektek elkészítésébe is. Kialakításakor külön figyelmet fordítottam arra, hogy a tanórák szűk keretein belül minél több területet és programot lefedjen és látványos maradjon. A modulokon végighaladva a csoport először megismerkedik a szoftver használatával, majd már az első feladatnál grafikai felületet épít fel és alapvető mozgást valósít meg minimális beállításokkal. Ezt követi a mélyebb elmerülés a program lehetőségeiben és folyamatosan haladva egyre komplexebb projekteket valósítanak meg. Bár a csoport részéről elvárt az önálló munka, mégis ez gyakorlatban nem mindig kivitelezhető, ezért a tanár szolgál vezetőként a diákok számára.

A kezdeti bevezetést követik a már vizuális programozást is igénylő feladatok. Az egyszerűség kedvéért itt még csupán csak néhány soros kódot készítünk és az egyes utasítások vagy feltételek nem kerülnek kombinálásra. Emellett fokozatosan érintik a program újabb, kevésbé szembeötlő funkcióit (például az animációkezelés vagy grafikai szerkesztés).

Az ezt követő modul már alapértelmezettnek veszi az eseménylapok használatát és igyekszik komplexebb eseményblokkokat vagy utasításcsomagokat kialakítani további új lehetőségek bemutatásával.

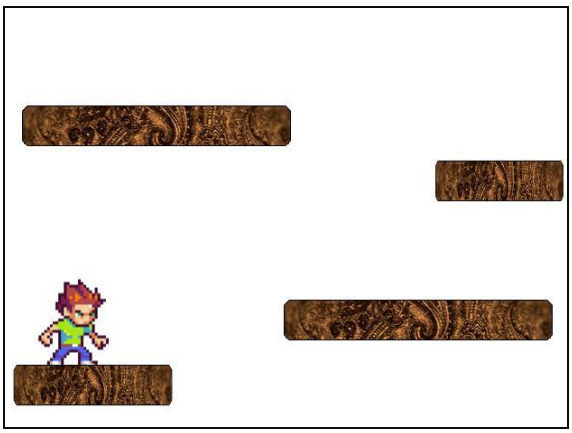
Az elmélyülés és közeledés a hagyományos programozási nyelvek utasításaihoz folyamatos. A következő részben a vizuális programozás további elmélyítése kapja a hangsúlyt. A tanulók kihasználják annak lehetőségeit: alseményeket hoznak létre, illetve megismerkednek a különböző kifejezések használatával (például véletlenszám generálás, vagy koordináta lekérdezés).

Ezek mellett figyelembe véve a játékfejlesztés kreatív oldalát is, a diákok egy önálló (vagy csoportos) beadandó programot is elkészítenek. Minimális irányadás mellett a tanulók fantáziájára van bízva, hogy hogyan építik fel az órákon (és egyénileg) szerzett tudásukat felhasználva.

A továbbiakban a kidolgozott oktatási tervezetből emelek ki néhány feladatot, azok általános bemutatásának céljából.

Feladat 3: Ugrálás fölfelé

Juttassuk fel a fiút a legmagasabb pontra. Hozzunk létre egy fiú objektumot, amelyhez platform mozgási viselkedést rendelünk, illetve egy talaj elemet, amelyen nem mehet keresztül a fiú (1. ábra).

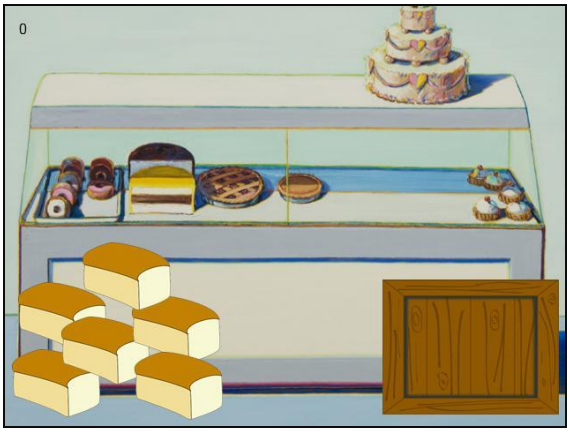


1. ábra

Feladat 6: Kenyerek pakolása

Pakoljuk be a kenyereket a ládába és közben számoljuk meg, hogy hányat pakoltunk be.

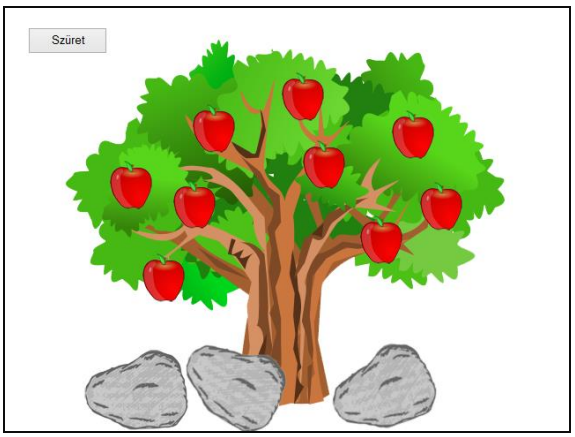
Hozzunk létre egy háttér, egy kenyér és egy láda objektumot, valamint egy szövegdobozt, amibe megszámoljuk az elpakolt kenyerek számát (2. ábra).



2. ábra

Feladat 7: Almaszedés

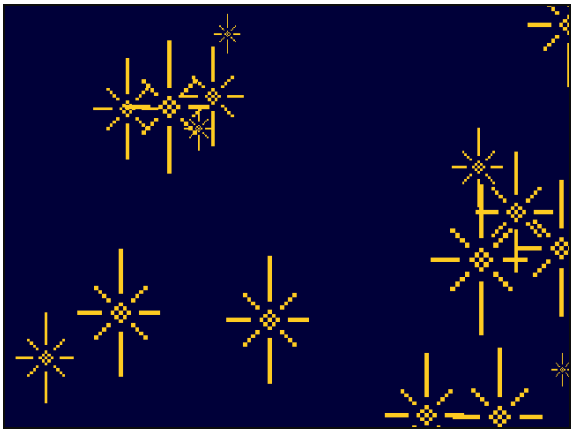
Szüreteljük le a kövek mellé ültetett almafát egy gombra kattintva. Hozzunk létre egy fa objektumot, a tövében kövekkel, illetve rajta elhelyezett almákkal. Adjunk hozzá egy gombot, amely megnyomása után az almák a beállított viselkedés miatt leesnek a kövekre, amelyeken fennakadnak vagy legurulnak róluk (3. ábra).



3. ábra

Feladat 11: Csillagos éjszaka

Töltsük fel az égboltot különböző méretű csillagokkal. A megoldáshoz szükségünk lesz egy csillag objektumra, amelyet másodpercenként véletlenszerű pozícióba és különböző méretben több példányban megjelenítünk a képernyőn (4. ábra). A csillagok létrehozását a billentyűzet bármelyik gombjának megnyomásával megszakíthatjuk. A feladat célja, hogy megismerkedjünk a hagyományos programozási nyelvekben már közismert while ciklussal a Construct 2 környezetében.

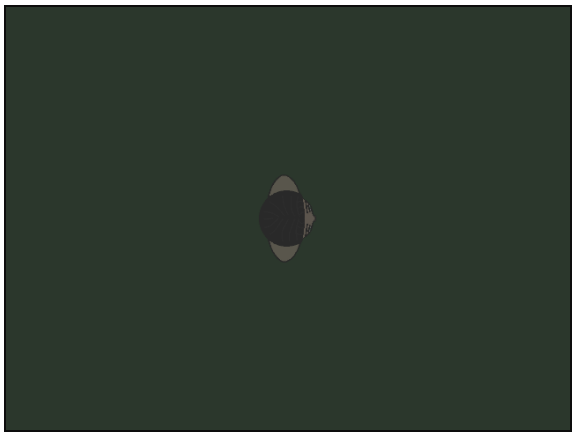


4. ábra

Feladat 13: Erre csörög a dió

Keressük meg a zene forrását! A feladat a tradicionális „Erre csörög a dió” játék számítógépes megvalósítása. Egy hangforrással rádió és egy gyerekkel játszunk. A rádió folyamatosan játssza a zenét, amely aszerint halkuljon vagy erősödjön, hogy a gyerek milyen messze van a rádiótól.

A megvalósításhoz szükségünk van a következő objektumokra: egy egyszínű háttérrel rendelkező rétegen egy gyerek és egy rádió objektumra. A karaktert mozgathassuk a kurzormozgató nyilak segítségével (5. ábra).



5. ábra

Összegzés

Napjainkban számos elterjedt módja van az általános- és középiskolai intézményekben az algoritmizálás készségének kialakításának. Az általános iskolákban elterjedt módszerek közé tartoznak a Lego, illetve Logo programozási nyelvek oktatása, illetve középiskolákban a hagyományos programozási nyelvek mellett elterjedőben van a táblázatkezelő függvények segítségével oktatott algoritmizálás és a mobil programozás is. Ezek mellett sajnos azonban a tapasztalat szerint a tanulók nem rendelkeznek még közel sem azzal az algoritmizáló képességgel, amelyet a kerettanterv előír. Ezt a problémát szeretném megközelíteni a vizuális programozás irányából, azon belül is napjaink egyik legnépszerűbb vizuális környezetét használva: a Construct 2-t.

A Construct 2 tanulása közben elsajátított tudást a későbbiekben többszörösen is kamatoztathatják a diákok. A hasonló kategóriában elterjedt szoftverek száma folyamatosan nő és ezzel párhuzamosan csökken az űr a köztük és hagyományos programozási nyelvvel készített alkalmazások között. Így nemcsak az algoritmikus készség fejlődéséről beszélhetünk, hanem egy olyan módszerről is, amelyet felhasználva (akár a Construct 2 környezetnél maradva) piacképes szoftverek fejleszthetőek.

A Construct 2 oktatási tervnek nem célja, hogy kiszorítsa a hagyományos programozási nyelveket az oktatásból. Ehelyett annak újragondolását szorgalmazza. A programozási nyelvek előnyei vitathatatlanok, azonban nem megfelelőek és nem is szükségesek minden diák számára. Az alapvető informatikaoktatás egyik célja, hogy az algoritmikus készség kialakuljon, amelyre tökéletesen alkalmas a Construct 2 szoftver egy egyszerű és látványos módon tanítva azt. A programozási nyelvek ellenben az informatika iránt mélyebben érdeklődőknek, illetve a későbbiekben ilyen irányba elinduló tanulók számára hasznosak, így azoknak csupán fakultációs lebontásban való tanítását javasolnám. Az otthonában nagyon kevés tanuló próbálja ki, amit az iskolában tanult, nem írnak parancssori ablakban futó programokat. Ellenben képzeljük el, hogy ez a szám

mennyivel nőne ha játékokat fejleszthetne nyelvi akadályok nélkül. A játékszoftverek szerves részei különböző generációk életének és messze túlmutatnak a hagyományos „játék” fogalmán. Egy többdimenziós médiumról beszélhetünk, amely jelenleg nagyobb piacot tudhat magáénak, mint a filmipar. Ha a látványosság, egyszerűség és a siker élménye párosítható a fejlesztési folyamattal, úgy a tanulók nagyobb eséllyel kezdenek majd a későbbiekben is érdeklődni további programozási nyelvek iránt.

Irodalomjegyzék

1. Katona Endre (2000) Bevezetés az informatikába, Panem Könyvkiadó, 155–159
2. Rozgonyi-Borus Ferenc, Kokas Károly (2002) Informatika középiskolásoknak, Mozaik Kiadó, 168–172
3. Scirra (2011) Create Games with Construct 2 - Scirra.com https://www.scirra.com/ Letöltés dátuma: 2014. április 18.
4. YoYo Games (2007) Welcome to YoYo Games http://www.yoyogames.com/ Letöltés dátuma: 2014. április 14.
5. Stencyl (2011) Create iOS, Android and Flash Games with Stencyl http://stencyl.com/ Letöltés dátuma: 2014. április 14.
6. GameSalad (2009) Game Design Engine, Make Games for iPhone & Android - GameSalad http://gamesalad.com/ Letöltés dátuma: 2014. április 14.
7. Game Editor (2011) Game Edior http://game-editor.com/ Letöltés dátuma: 2014. április 20.
8. Oktatáskutató és Fejlesztő Intézet (2012) Kerettanterv - Oktatáskutató és Fejlesztő Intézet, OFI http://kerettanterv.ofi.hu/ Letöltés dátuma: 2014. április 27.
9. Oktatáskutató és Fejlesztő Intézet (2013) Kerettanterv - Oktatáskutató és Fejlesztő Intézet, OFI http://kerettanterv.ofi.hu/08_melleklet_szakiskola/index_szi.html Letöltés dátuma: 2014. november 14.
10. Scirra (2014) Official Construct 2 Manual - Construct 2 Manual https://www.scirra.com/manual/1/construct-2 Letöltés dátuma: 2014. november 14.
11. Scirra (2014) Construct 2 Releases - Scirra.com https://www.scirra.com/construct2/releases Letöltés dátuma: 2014. november 15.
12. Scirra (2014) Education and Construct 2 - Scirra Forums https://www.scirra.com/forum/education-and-construct-2_f149 Letöltés dátuma: 2014. november 15.
13. Scirra (2012) A new way to learn how to program - Scirra.com https://www.scirra.com/blog/82/a-new-way-to-learn-how-to-program Feltöltés dátuma: 2012. május 3. Letöltés dátuma: 2014. november 15.