Prvi koraki v programiranje – številne poti in možnosti

Matija Lokar
Univerza v Ljubljani, Fakulteta za matematiko in fiziko, Jadranska 19, 1000 Ljubljana
Matija.Lokar@fmf.uni-lj.si

Izvleček

V zadnjih letih so v večini razvitih družb prišli do spoznanja, da je nujno, da vsi učenci usvojijo algoritmično razmišljanje in znanje osnov programiranja. S spoznavanjem računalniških konceptov in razvijanjem postopkovnega načina razmišljanja učenci pridobivajo znanje, spretnosti in veščine, ki so veliko bolj trajni kot obvladovanje konkretnih postopkov pri hitro razvijajočih se tehnologijah.

Seveda je treba zelo skrbno razmisliti, kako začeti s poučevanjem programiranja – kateri so ustrezni prvi koraki. V prispevku bodo predstavljeni številni načini, ki jih lahko uporabimo v ta namen. Tako kot so številni načini uvajanja, tako se odpirajo različne možnosti, pri kateri starosti je primerno vpeljati osnovne koncepte. Različne študije kažejo, da je mogoče začeti že z najmlajšimi, celo v vrtcu.

Ključne besede: algoritmično razmišljanje, osnove, programiranje, poučevanje.

Abstract

First Steps into Programming – Numerous Paths and Possibilities

In recent years, most developed societies have realized that it is vital for students to acquire the skill of algorithmic thinking and a basic knowledge of computer programming. Through getting to know computer-related concepts and developing the step-by-step thought process, the students acquire knowledge and skills that are much more permanent than the knowledge of performing concrete procedures in the rapidly developing technologies of today.

We will show which programming concepts can be taught in this manner. Due to the numerous ways and means of teaching, the appropriate timing of introducing basic programming concepts to students varies, too. Several studies show it is possible to start with very young learners, even at a preschool age.

Keywords: Algorithmic thinking, Basics, Programming, Teaching.

1 UVOD

Ko beseda nanese na učenje programiranja, je pogosta reakcija mnenje, izraženo kot »to je v današnjem času povsem nepotrebno znanje oziroma znanje, ki je potrebno le ozki skupini ljudi. Da bi vse učili programiranja je prav tako, kot da bi poskusili, da bi vse naučili, kako se popravlja in sestavlja avtomobil. Hočemo imeti voznike in ne mehanikov.«

Vsi, ki smo kljub tovrstnim pripombam trdili, da osnovno poznavanje programiranje spada v splošno izobrazbo, v osnovno pismenost sodobnega človeka, smo bili pri svojem mnenju osamljeni (ali pa ne dovolj glasni in prepričljivi (Lokar, 2005)).

Žal je prevladal pogled na to, da je »računalnik zgolj orodje« in da je vedenje o računalniških principih povsem nepotrebno. To je že pred časom povzročilo, da je programiranje, ki je včasih še kako bilo prisotno v šoli in je bilo nujna sestavina vseh predmetov, ki so bili »računalniški«, čedalje bolj izginjalo iz šolskih programov, računalništvo pa je postalo učenje uporabe računalniških orodij. Ta proces je bil prisoten v večini držav. Čeprav so računalniki povsod, računalništvo uči manj šol kot pred desetimi leti (Code.org, 2014).

Pred nekaj leti je prišlo v strokovnih krogih do precejšnjih sprememb v aktivnostih in javnem izražanju zahtev po potrebnih spremembah v konceptu poučevanja računalništva. Prišlo je do sprememb v pogledih na vlogo računalništva v šolskih programih in tudi v družbi sploh. Tako je na primer angleška Royal Society januarja 2012 objavila poročilo (Royal Society, 2012), ki pravi, da je treba korenito spremeniti pristop k poučevanju računalništva ali pa bo angleška družba začela zaostajati v razvoju. V organizaciji Code.org je decembra 2013 petnajst

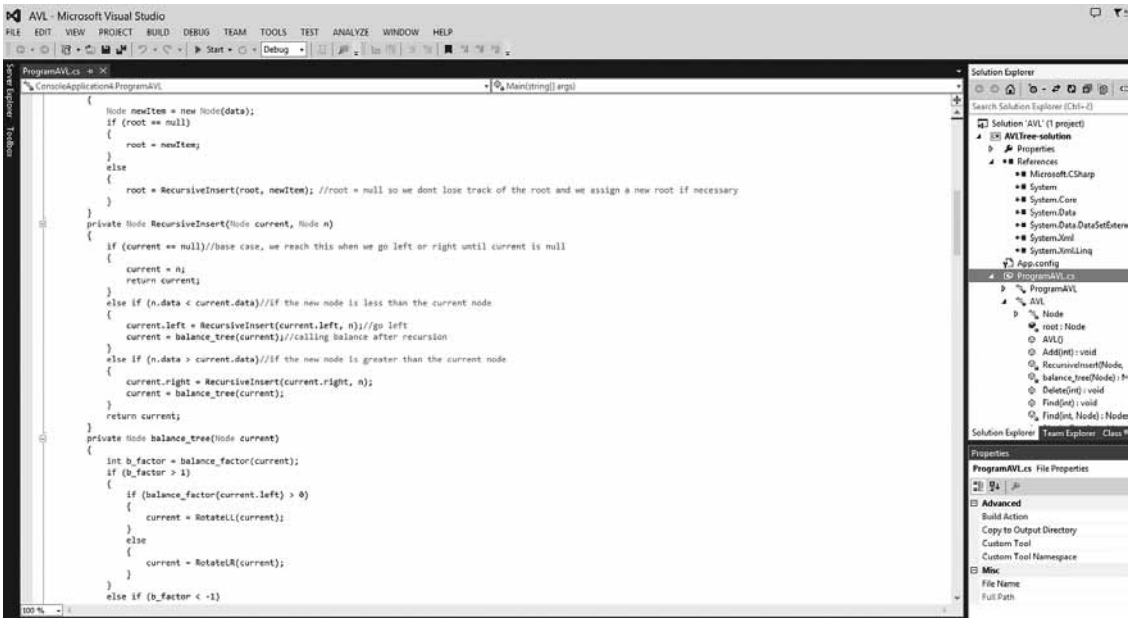
milijonov učečih se sodelovalo v aktivnosti Hour of Code, ki je bila namenjena temu, da »vse« spozna s tem, kaj je programiranje in zakaj je to pomembno znanje sodobnega človeka (Code.org – Hour of code, 2013). Letošnja ponovitev je na skoraj 80.000 dogodkih v več kot 180 državah število sodelujočih učencev dvignila na skoraj 81 milijonov (Code.org, 2014).

Slika 1: **Mnenja s spletne strani http://hourofcode.com/**

Ker namen prispevka ni poglabljanje v določena teoretična izhodišča ali analiza stanja, recimo le, da po svetu vedno bolj ugotavljajo, da je računalniška pismenost nekaj povsem drugega kot vedenje o tem, kako uporabljati urejevalnike besedil, se »sprehajati« po omrežju, pošiljati e-pošto ipd. Nujno je, da učenci postanejo aktivni ustvarjalci na področju računalništva in ne pasivni odjemalci računalniške tehnologije. Da bi to lahko postali, jih je treba naučiti algoritmičnega razmišljanja. Spoznavanje osnovnih programerskih korakov je ena od najboljših poti k usvajanju tega.

Kako se lotiti tega? Kakšna pot je ustrezna, kateri programski jezik uporabiti? Katera starost je primerna?

Bodo potem naši učenci sedeli pred računalnikom in se »mučili« s pisanjem programov v okoljih, kot jih prikazuje slika 2?



Slika 2: **Naj bo programiranje takšno?**

Nikakor ne. Danes imamo številne druge možnosti, ki bistveno olajšajo prve korake v svet programiranja. V nadaljevanju si bomo ogledali nekaj različnih pristopov, kako lahko začnemo uvajati programiranje.

2 KAJ JE PROGRAMIRANJE

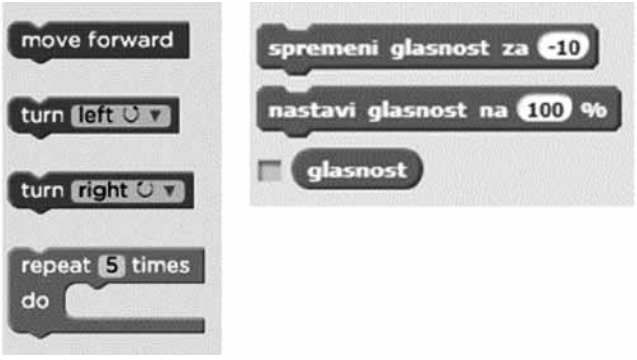
Preden si bomo ogledali nekaj možnosti, kako se lahko lotimo učenja programiranja, si oglejmo, kaj sploh je programiranje. Najkrajše lahko rečemo, da je to sestavljanje navodil za procesor. In kaj je procesor? Vsekakor najprej pomislimo na košček elektronike, ki se skriva v računalniku, na »srce računalnika«. Vendar lahko na procesor gledamo precej bolj splošno. Procesor je pravzaprav vsakdo, ki zna izvajati določen nabor opravil. Tudi človek. Ko na primer v Google Maps uporabimo ukaz, ki poišče pot od točke A do točke B, pravzaprav naročimo, da želimo dobiti program, namenjen procesorju – človeku. In če sledimo tem navodilom, izvajamo program.

Če na programiranje in temeljne programerske koncepte gledamo v tej luči, se odpirajo številne možnosti, kako učečim se razložiti pojem programa kot zaporedja ukazov, pogojnega stavka, zanke, funkcije itn. Nekaj teh možnosti si bomo ogledali v nadaljevanju.

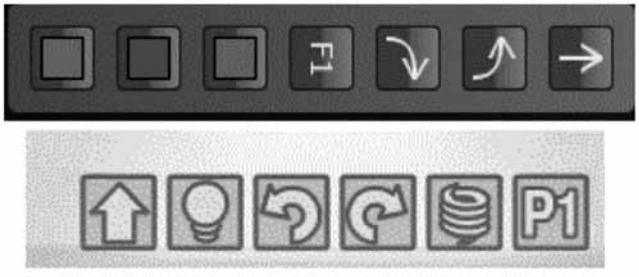
Zato tudi pojem programskega jezika lahko opredelimo bistveno bolj splošno. Na programski jezik lahko gledamo kot na sestavne bloke za pripravo navodil za določeno programsko okolje. Ti bloki so lahko »klasični programski«, kot nam jih prikazuje slika 3, a tudi taki, kot vidimo na slikah 5 in 6, ali celo v fizični obliki (slika 6).

```
random.getstate()
    Return an object capturing the currer

random.setstate(state)
    state should have been obtained fror

random.getrandbits(k)
    Returns a Python integer with k ranc
    When available, getrandbits() er
```

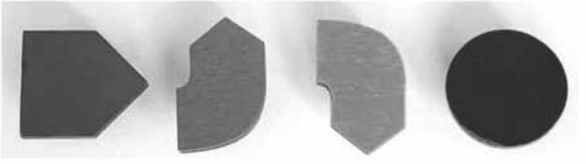
Slika 3: **Sestavni deli programskega jezika – »klasični«**



Slika 4: **Sestavni deli programskega jezika – »kocke«**



Slika 5: **Različne vrste programskih jezikov**



Slika 6: **Fizična oblika ukazov programskega jezika**

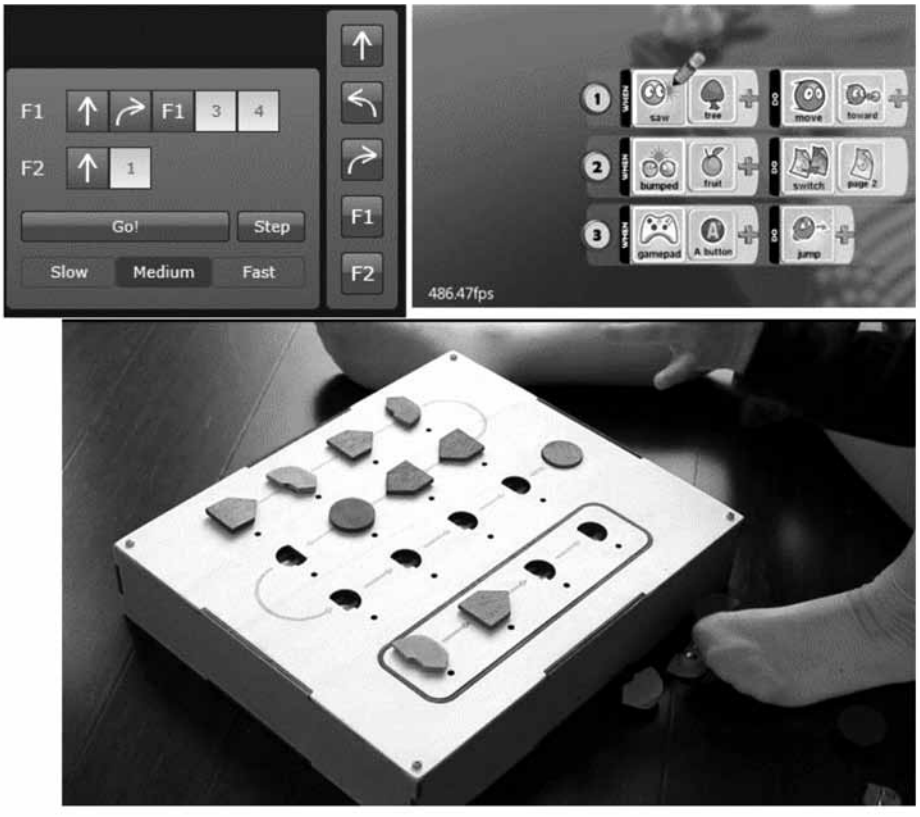
Okolja, v katerih programiramo, niso nujno taka, kot kaže slika 2, temveč so lahko tudi drugačna (slika 7).

3 KAJ NAJ BI SE NAUČILI

Sedaj, ko smo naš pojem programiranja zastavili tako splošno, premislimo, kateri so tisti pojmi in koncepti, s katerimi naj bi seznanili začetnika. Spisek je presenetljivo kratek:

- slediti zapisanemu programu (predvideti rezultat),
- spremeniti zapisani program,
- sestaviti svoj program.

Prvi dve točki sta izjemno pomembni in v klasičnem pristopu k poučevanju programiranja prepogosto zanemarjeni. »Pravo« programiranje je dejansko le zadnja točka. In kaj učiti tam? Tudi ta spisek lahko končamo pri tretji točki:



Slika 7: **Različna okolja za programiranje**

- zaporedje ukazov in pomembnost vrstnega reda,
- vejitev,
- zanka.

4 KAKO

Glede na zgoraj vpeljane pojme in naštete različne možnosti je tudi načinov, kako se vsega skupaj lotiti, presenetljivo veliko. Katere bomo uporabili, je odvisno od starosti, zanimanja, časa, okolja, skratka od konkretne pedagoške okoliščine. V prispevku se bomo omejili predvsem na tiste načine, ki so primerni za vse starostne skupine. Zato ne bomo omenili določenih gradiv, ki so namenjena starejšim učencem predvsem kot uvod v »resno programiranje« in predpriprava spoznavanja kompleksnejših jezikov, kot so Java, C# in podobni.

Naj naštejemo le nekaj možnosti, ki jih imamo, če želimo najmlajše vpeljati v svet programiranja:

- programiranje brez računalnika,
- ob računalniških igrah,
- z upravljanjem robotov,
- z uporabo programskih jezikov, zasnovanih na konceptu zlaganja kock.

Večina aktivnosti je zasnovana na igri, sodelovalnem delu in pogosto na fizičnih aktivnostih kot sestavnih delih »programiranja«. Tako dejansko podiramo številne stereotipe, ki jih imamo o programiranju in so v glavnem posledica tega, da je zelo veliko uvodnih tečajev v programiranju zasnovanih na zgledih, ki kot prvo dejavnost prikažejo programe, kot so ti na sliki 8, in z vajami z navodili, kot so npr.:

- Sestavi program, ki prebere dve števili in izpiše njuno vsoto.
- Napiši program, ki izpiše vsa soda števila med 1 in 1000.
- Izračunaj vsoto recipročnih vrednosti kvadratov vseh števil med a in b.

```
.model tiny
.code
org 100h

main proc
   mov ah, 9
   mov dx, offset pozdrav
   int 21h

   retn

pozdrav db 'Pozdravljen svet!$'

main endp
end  main
```

```
public class Pozdrav {
    public static void main(String[] args) {
        System.out.println("Pozdravljen, svet!");
    }
}
```

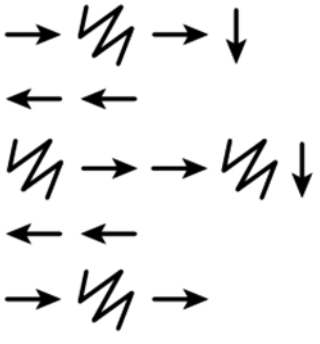
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication2
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Pozdravljen svet!");
        }
    }
}
```

Slika 8: **Prvi programi**

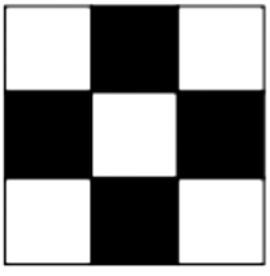
4.1 Brez računalnika

Presenetljivo veliko ciljev, ki jih želimo doseči z uvajanjem programiranja v pouk, lahko dosežemo brez uporabe računalnika. Tako obstajajo namizne igre, primer katerih je Robot Turtles (Shapiro, 2014). Čeprav to morda ni razvidno na prvi pogled, se v igri skrivajo številni programerski koncepti. Zato ni čudno, da je ob objavi igra požela izjemen odziv.

Kot smo omenili, je pomemben sestavni del učenja programiranja sledenje programom. Tudi za to ne potrebujemo računalnika. Dovolj bo karirast papir in barvice (Code.org, 2014). Učenci vidijo, da je tudi to, kar prikazuje slika 9, program, katerega rezultat je slika 10.



Slika 9: **Program? Program!**



Slika 10: **Rezultat programa**

Učenci ob tej in podobnih dejavnostih spoznajo pomembnost vrstnega reda ukazov, vidijo, da za rešitev določene naloge obstaja več postopkov (programov), vidijo, da so določene ideje njim povsem jasne, a jih procesor (drug učenec) lahko razume narobe, če niso pravilno izražene, in še številne druge koncepte.

Tovrstnih dejavnosti je še veliko. Odličen vir zanje so gradiva, nastala v okviru projekta Computer Science Unplugged (Computer Science Unplugged, 2014). Gradiva imamo po zaslugi prof. dr. Demšarja na voljo tudi v slovenskem jeziku (Demšar, 2014).

4.2 Ob računalniških igrah

O uporabnosti računalniških iger pri učenju je znanega že veliko (mdr. Prensky, 2005; Kafai, 1995; Aldrich, 1995; Marji, 2014). Zato ni čudno, da tudi za

učenje programiranja obstajajo številne primerne igre. Izkušnje kažejo, da z njimi lahko dosežemo odlične rezultate tako pri osnovnih konceptih, kot so zaporedje ukazov, vejitev, zanka, kot tudi pri vpeljavi zahtevnejših konceptov. Med njimi omenimo funkcijo (podprogram) in rekurzijo.

Pri številnih tovrstnih igrah gre za to, da imamo na voljo določen nabor zelo preprostih vnaprej pripravljenih ukazov. Z njimi upravljamo določeno figuro, ki mora rešiti različne naloge (prižgati žarnico, ujeti pujska, prevoziti vse poti ipd.). To storimo tako, da ukaze s pomočjo vlečenja postavimo na ustrezna »programska mesta« in s tem sestavimo program. Že z zelo omejenim naborom ukazov učence lahko seznanimo s prijemi, kot so pomembnost pravilnega zaporedja ukazov, odločitev (pogojni stavek), pojem funkcije, klic funkcije, učinkovitost programa, rekurzija idr.

Omenimo nekaj najbolj tipičnih predstavnikov.

Lightbot

Gre za igrico kanadskega programerja Coolia Niata, pri kateri moramo pripraviti navodila, s katerimi vodimo možička po njegovem svetu tako, da prižge vse žarnice. Igra je na voljo za različna okolja. Tako jo lahko uporabimo na Applovih napravah (Lightboot Inc., 2014), na napravah z operacijskimi sistemom Android (Google play, 2014) in v okolju Windows (Kongregate, 2014). Na voljo je tudi posebna različica, namenjena najmlajšim, LightBoot Jr.

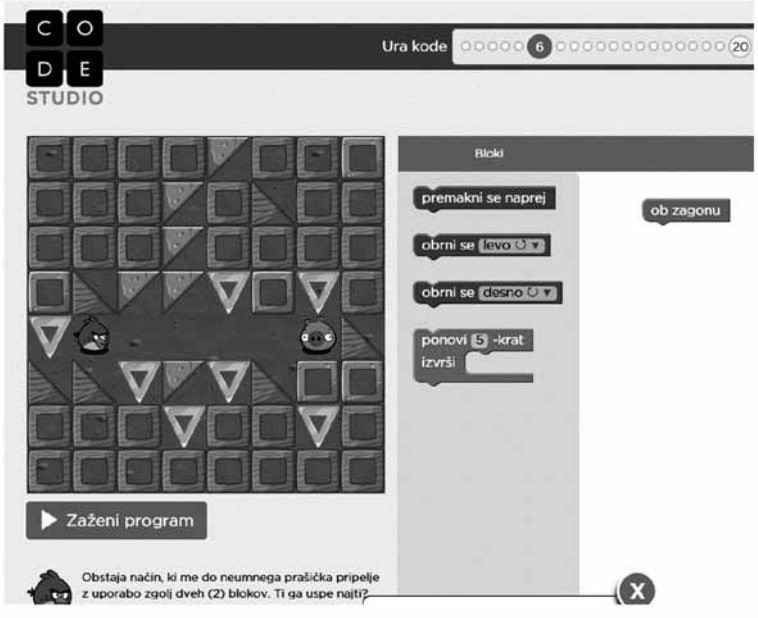
V sklopu prej omenjene Hour of code (Code.org – Hour of code, 2013) so pripravili spletno različico, ki je na voljo tudi v slovenskem jeziku.

RoboZZle

Igor Ostrovsky, programer pri Microsoftu, je leta 2009 objavil spletno igro, ki je hitro postala izjemno popularna, še posebno ko se je na YouTube pojavil demonstracijski video (Ostrovsky, 2009). Tudi tu gre za to, da sestavimo navodila za premikanje figure. Ta se mora premikati samo po pobarvanih kvadratkih in pri tem pobrati vse zvezdice. Kot je danes že običajno, je tudi ta igra na voljo za različne naprave in operacijske sisteme (RoboZZle, 2014; Google Play, 2012; Maxwell, 2013). Igra je odličen uvod v spoznavanje pojma funkcije (podprograma), saj je za rešitev večine nalog nujno, da glavna funkcija (F1) kliče druge.

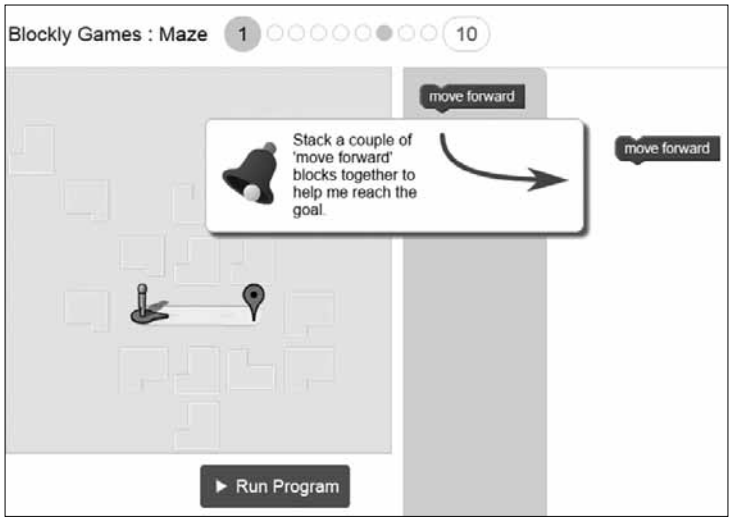
The Maze

V sklopu Hour of code (Code.org – Hour of code, 2013) so pripravili igrico The Maze (Code.org – The Maze, 2014), ki uporablja znane like iz popularne igre Angry Birds. Tudi pri tej moramo s sestavljanjem vnaprej pripravljenih ukazov opraviti določene naloge. Ukazi so oblikovani po vzoru ukazov, ki jih uporablja programski jezik Scratch (Scratch Group – MIT , 2014), zato je ta igra lahko zanimiv uvod v spoznavanje jezika Scratch.



Slika 11: **The Maze – http://studio.code.org/hoc**

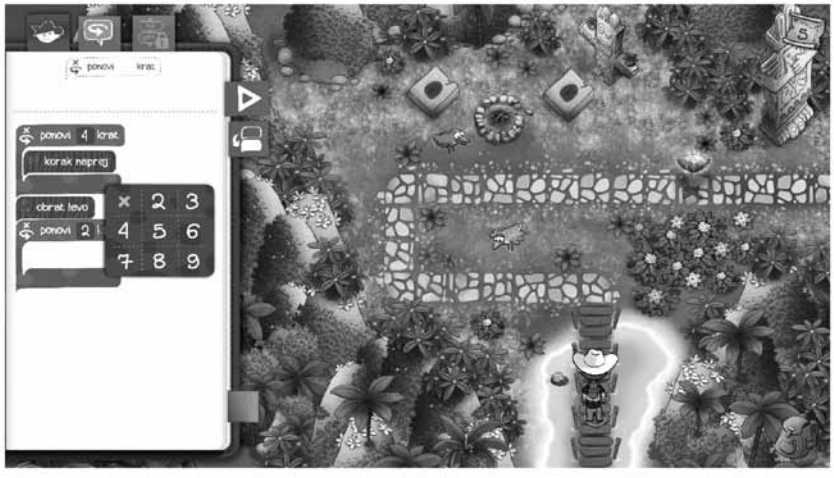
Zelo podobna igra je Googlova demonstracija uporabe jezika Blockly (Google Projects – Blockly, 2014), ki se prav tako imenuje Maze (Google Projects – Blockly/Maze, 2014).



Slika 12: **Blockly – Maze**

Teci, Marko!

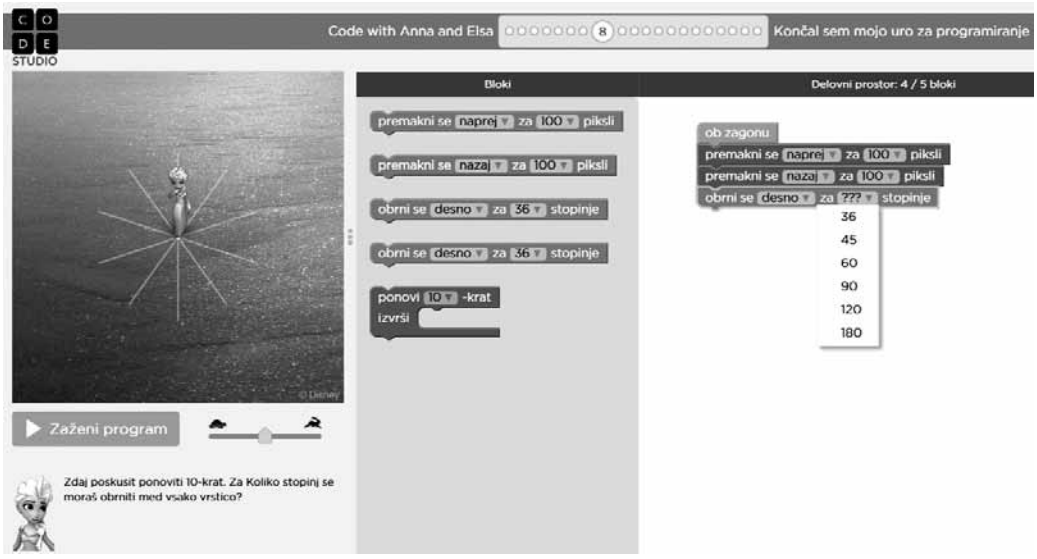
Teci, Marko! je še ena od iger, ki sledijo konceptu »programiranja s sestavljanjem blokov«. Igra je še v nastajanju, saj je na voljo le nekaj stopenj. Pustolovca Marka moramo voditi po džungli, kjer išče drage kamne in druge zaklade. Je precej nezahtevna, zelo barvita in ker je po zaslugi sodelovanja Fakultete za matematiko in fiziko na voljo tudi v slovenskem jeziku, bo pritegnila že najmlajše. Obstajajo tako spletna različica (Allcancode, 2014) kot tudi različice za mobilne telefone.



Slika 13: **Igra Teci, Marko! (http://www.allcancode.com/)**

The Frozen

Za letošnjo (2014) izvedbo Hour of Code so pri Code.Org uspeli pridobiti pravice, da uporabijo like iz priljubljenega risanega filma Frozen. Tako je nastala igrica, pri kateri moramo z zlaganjem ukazov sestaviti navodila za drsanje, da sledovi drsalke opišejo predpisani vzorec. Pri igrici se zelo pozna eden od ciljev letošnjega dogodka – še posebno pozornost posvetiti temu, da v svet programiranja privabijo čim več deklet. Tako imamo možnost izbire barv, objavljanja svojih izdelkov v oblaku, lahko si jih natisnemo itn.



Slika 14: **The Frozen – http://studio.code.org/s/frozen**

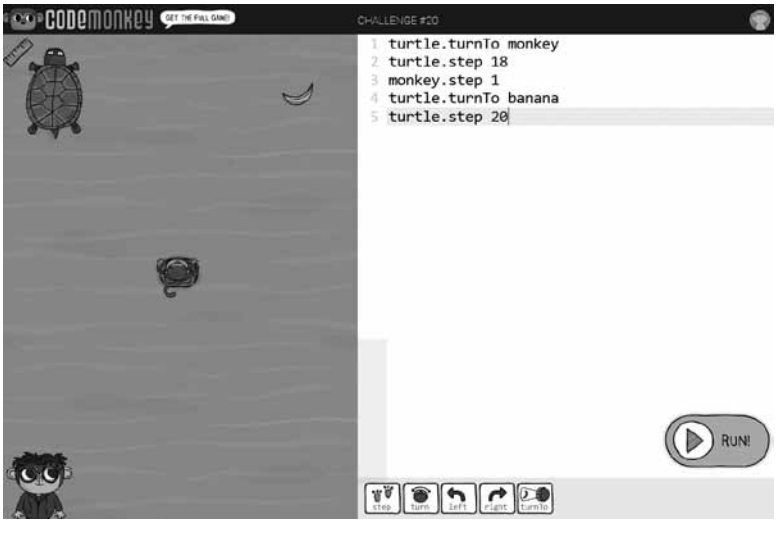
The Flappy Code

Zanimivo nadaljevanje poučevanja je na voljo v sklopu v prejšnjem razdelku omenjenega spletnega mesta (Code.org – Hour of code, 2013). Če so učenci pri prej omenjenih igrah »zgolj« reševali naloge, pa morajo v sklopu te igre (code.org – The Flappy Code, 2014) početi dvoje: najprej sestaviti igrico, potem pa jo še odigrati.

CodeMonkey

Pri izraelskem podjetju, avtorju igre CodeMonkey, menijo, da lahko uporaba programiranja s tehniko povleci in spusti (torej tako kot upravljamo vse prej naštete igre) pomeni določeno oviro pri nadaljnjih korakih v svet programiranja. Zato je v njihovem okolju treba ukaze (vsaj deloma) tipkati. Še vedno pa so zasnovani na principu upravljanja figure, ki mora opraviti določene naloge.

Samo okolje je precej dodelano, razlikuje med učencem in učiteljem. Igra vsebuje kar nekaj pedagoško zanimivih prijemov (možnost merjenja razdalje, naloge s popravljanjem kode ipd.). Učencu omogoča, da spremlja svoj napredek, učitelju pa, da nadzoruje dogajanje v razredu; vse skupaj lahko upravljamo in analiziramo na ravni šole. Ves nadzor je plačljiv, samo igro pa lahko v večjem delu igramo brezplačno.



Slika 15: **Codemonkey – http://www.codemonkey.co.il/**

CodeCombat in CodeSpells

CodeCombat in CodeSpells sta tipična predstavnika skupine iger, ki poskušajo biti čim bolj »igralne«. Organizirane so podobno kot »navadne« računalniške igre, le da moramo za premikanje junaka ali reševanje določenih nalog napisati ustrezno kodo. Žal je pri tovrstnih igrah pogosto veliko večji poudarek na igri in je samo učenje programiranja bolj »postranska« zadeva. No, obe tukaj omenjeni igri sta po mojem mnenju našli dobro ravnovesje med igro in učenjem. Prva (ICTMagic, 2014) je predstavnica tako imenovane platformske igre, druga, sicer še v razvoju (ThoughtSTEM, 2014), pa je igra igranja vlog z zelo bogato grafiko in bo verjetno navdušila predvsem nekoliko starejše učence.

Slika16: **CodeCombat – http://codecombat.com/play**

CodeHunt

To je igra, ki prihaja iz Microsoftovih raziskovalnih oddelkov. Namenjena je učenju jezikov JavaScript in C#. Omenjamo jo zaradi inovativnega pristopa. Učenje poteka tako, da dobimo že napisane delčke kode in rezultate testnih primerov. Kodo moramo spremeniti tako, da bodo vsi testni primeri dali predvideni rezultat. V igri se skriva vrsta raziskovalnih dosežkov, ki so med drugim dokumentirani na Microsoft Research (2014).

Slika17: **Code Hunt – https://www.codehunt.com/**

4.3 Programiranje robotov, sestavljanje računalnikov

Določene raziskave kažejo, da je zelo primeren način vpeljave otrok v svet programiranja in algoritmičnega razmišljanja prek programiranja robotkov. S tem pride do povezave med fizičnim svetom in miselnimi vzorci. Ker bi to področje zahtevalo svoj prispevek, tu le naštejemo nekaj možnosti in ustreznih gradiv.

LEGO Mindstorms NXT (LEGO, 2014) je verjetno ena od prvih idej, na katere pomislimo ob omembi otrok in robotov. Za našo zgodbo o programiranju je ta izdelek zanimiv, ker se je z leti programski jezik zelo »izbrusil« in postal preprost ter nadvse primeren za prve korake v svet programiranja (Fisher, 2014; Cliburn, 2006; Lawhead, 2003).

Primo (Solid Labs, 2014) je projekt, namenjen res že najmlajšim. Tu otroci upravljajo robota (leseno kocko s kolesi) tako, da nekaj vnaprej pripravljenih programskih kock vstavljajo v »programsko desko«. Tu gre res za elementarno programiranje, vendar so programski koncepti, ki jih lahko usvojijo otroci, vseeno zelo bogati.

Play I (Play-i, 2014) – gre za še en projekt, ki je po objavi na platformi Kickstater (kot sta sredstva pridobila tudi prej omenjena projekta RoboTurtles (Shapiro, 2014) in Primo (Solid Labs, 2014)) hitro nabral želena sredstva in za katerega še ne vemo, ali bo res uspešen. Vsekakor so vsi trije projekti videti izjemno obetavni. Tu s pomočjo različnih programskih jezikov upravljamo dva robota, ki sta videti kot žogici. Projekt je zanimiv, saj nakazuje možnost, da pri upravljanju robota prehajamo od neposrednega upravljanja prek enostavnega uporabniškega vmesnika do uporabe programskih jezikov, zasnovanih na Scratchu in v obliki »klasične programske kode«.

Kano (Kano, 2014) – kaj pa če bi, preden se lotimo programiranja, najprej sami sestavili svoj računalnik. Ideja, ki je skupini avtorjev prinesla izjemen odziv na že prej omenjeni platformi Kickstarter in katerih največja trenutna težava je izpolniti vsa naročila. In ob tem nekateri še vedno trdijo, da je programiranje za otroke pretežko in preveč abstraktno.

4.4 »KODIRANJE«

Obstajajo številni jeziki, ki so primerni, da začetnika popeljejo v svet programiranja. Kateri je najprimernejši, je odvisno tako od starosti učečega se kot od številnih drugih okoliščin.

Logo

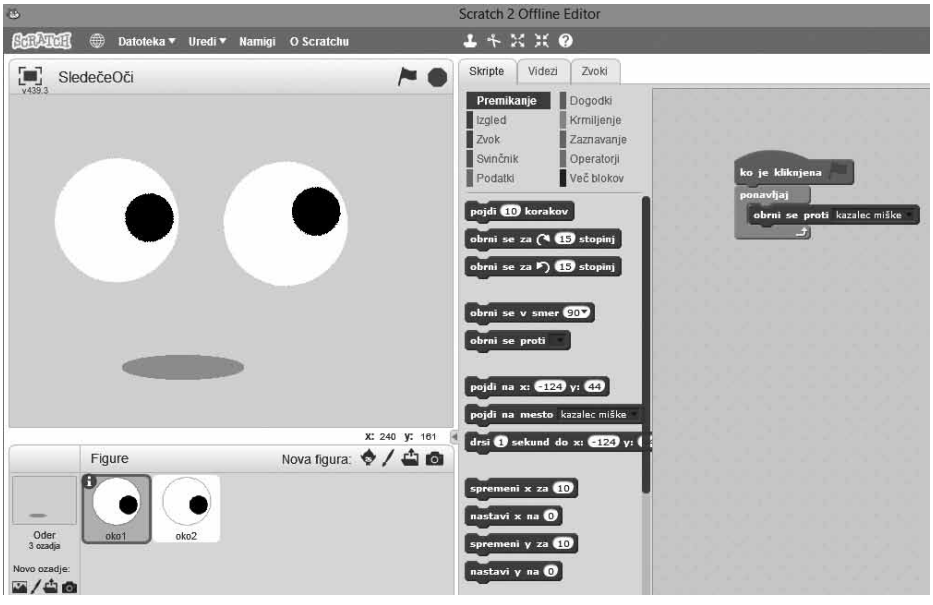
Tradicionalno se je kot prvi jezik pri najmlajših večinoma uporabljal Logo. Ta nastopa v različnih izvedbah in je na voljo v različnih okoljih (Softronix, 2014; Berkeley, 2014; FMSLogo, 2014). O uporabi tega je-

zika obstaja veliko literature (med drugim Mayer, 1988; Pardamean, 2014; McDougall, Murnane, & Wills, 2014; Batagelj, 1991).

Scratch

Leta 2003 je MIT Media LAB Lifelong Kindergarten Group pod vodstvom Mitcha Resnicka razvil programski jezik Scratch, katerega glavni namen je bil ustvariti programski jezik, v katerem bodo otroci na kar se da preprost način ustvarjali računalniške igre, animirane zgodbe, računalniško grafiko ipd. Ko je bila leta 2006 objavljena spletna stran, ki je omogočala programiranje v Scratchu, je jezik postal izjemno popularen. Popularnost mu je še povečal odmeven nastop Mitcha Resnicka na enem od dogodkov TED (Resnick, 2012).

Trenutno je Scratch eden od jezikov, ki ga največ uporabljajo pri prvih korakih v programiranje. Njegovo uporabo in primernost so v zadnjih letih izjemno veliko raziskovali (Resnick, 2009; Marji, 2014; (Medlock-Walton, 2014; Gruenbaum, 2014; Su, 2014; Joshi, 2013).



Slika 18: **Scratch 2.0**

Na temeljih tega jezika je nastalo še veliko izpeljank, kot so na primer Snap (Berkeley, 2014), PocketCode (Catrobat, 2014), Design Blocks (MIT Media Lab, 2014), Gamefroot (Gamefroot, 2014) idr.

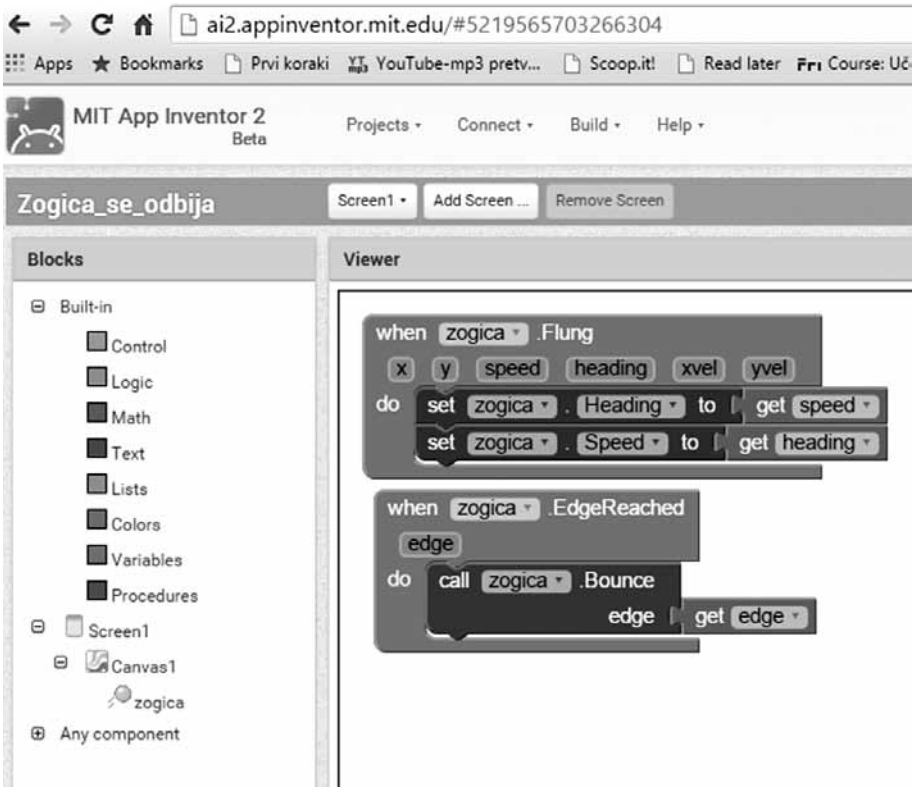
Scratch je zanimiv tudi zaradi tega, ker je dostopen v slovenščini. Poleg tega je v slovenščini na voljo tudi kar nekaj gradiv. Eno od takih zbirk najdemo na Lokar, Scratch – gradiva in prevod (2014).

MIT AppInventor

Glede na to, da otroci in mladostniki zelo veliko časa preživijo skupaj s svojimi mobilnimi telefoni, je bila »naravna« ideja, da bi jih poskusili v svet programiranja pripeljati prek ustvarjanja programov za pametne telefone. V sodelovanju podjetja Google in MIT Media Lab, avtorjev jezika in okolja Scratch je nastal MIT App Inventor (MIT Media Lab, 2014) oziroma Android App Inventor, programsko okolje in jezik, ki omogoča, da ustvarimo programe, ki tečejo na napravah z operacijskim sistemom Android. Za sam razvoj ni treba uporabljati pametnih telefonov, vendar je pravi učinek pri učenju dosežen takrat, ko lahko učenec na svoj telefon namesti program, ki ga je sam napisal.

Kot prikazuje slika 19, je samo programiranje precej podobno programiranju v Scratchu. Ker pa je razvoj samega uporabniškega vmesnika nekoliko bolj zapleten in zahteva določena naprednejša znanja, bo bolj primeren za starejše učence v zadnjem triletju osnovne šole.

Žal samo okolje ni na voljo v slovenščini.



Slika 19: **MIT App Inventor**

Kodu

Seveda je moral tudi velikan na področju računalništva, Microsoft, ponuditi ustrezno programsko okolje. Projekt je bil zasnovan v sklopu Microsoft Research (Microsoft Research, 2014). Okolje teče tako na namiznih računalnikih kot tudi na konzoli Xbox. Kodu je v prvi vrsti namenjen ustvarjanju grafično bogatih iger. Tudi tu je dan velik poudarek na spletni skupnosti (Kodu Game Lab, 2014), ki si izmenjuje projekte, izkušnje, scenarije idr. Tudi o uporabi tega jezika obstaja več raziskav, npr. Fowler & Cusak (2011).

Blockly

V sklopu Googlovih projektov je nastal projekt Blockly. Ideja projekta je ponuditi grafični urejevalnik, ki po vzoru jezika Scratch in podobnih ponuja vnaprej pripravljene bloke, ki jih z miško zlagamo skupaj. Sam urejevalnik potem lahko uporabimo v različnih drugih projektih. Tako Blockly uporabljajo že prej omenjeni MIT AppInventor (MIT AppInventor, 2014), Play-I (Play-i, 2014) idr. S stališča poučevanja programiranja je projekt zanimiv, ker omogoča pretvorbo programa, napisanega z Blocklyem, v Python ali JavaScript (Google Projects – Blockly, 2014).

Tako lahko naredimo prehod iz programiranja v jezikih, kot so Scratch in podobni, v bolj »tradicionalne« jezike, npr. Python, JavaScript idr.

5 SPLETNA OKOLJA

V zadnjem času je nastalo več spletnih okolij, v katerih so zbrane različne dejavnosti, namenjene seznanjanju s prvimi koraki v svet programiranja. Določene so namenjene samostojnemu učenju, spet druge ponujajo gradiva, namenjena tako učencem kot učiteljem.

RoboMind.net (RoboMind, 2014) je sklop dejavnosti (spletna stran, programski jezik, tečaji, učni načrti), s katerimi po besedah avtorjev otroke že od devetega leta starosti lahko vpeljujemo v svet programiranja. S pomočjo jezika lahko upravljamo tudi robote LEGO Mindstorms NXT, lahko pa vse izvajamo na računalniku, brez fizičnih robotov. Jezik in dokumentacija sta delno poslovenjena, žal pa je program po enomesečnem preizkusnem obdobju plačljiv.

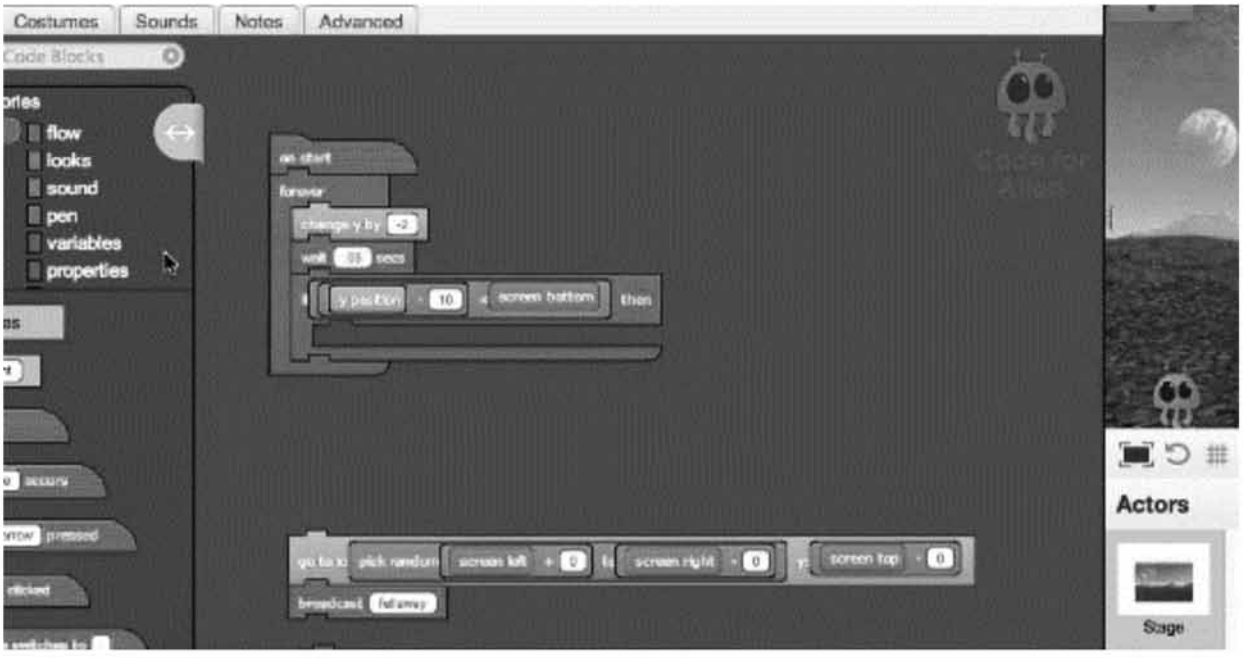
Code.org je neprofitna organizacija (Code.org, 2014), katere glavni cilj je pripeljati pouk računalništva v vse razrede od vrtca do konca srednje šole v ZDA. Decembra 2013 so bili med glavnimi nosilci akcije Hour of code (Code.org – Hour of code, 2013), ki je bila namenjena temu, da »vse učeče se« spozna s tem, kaj je programiranje in zakaj je to pomembno znanje sodobnega človeka. Izhodišče akcije je bilo »Na Kitajskem vsak učenec spozna programiranje, v ZDA pa le pet odstotkov. Popravimo to.«

Slika 20: **Code.org**

Na njihovi spletni strani je zbrano obilje gradiv, namenjenih tako učečim kot tudi učečim se. Tu bomo srečali številne v tem članku opisane dejavnosti in jezike ter še mnogo več.

Tynker (Tynker, 2014) je spletni učni sistem, ki z različnimi pristopi omogoča kar se da preprosto in učinkovito učenje programiranja. Ponujajo učna okolja, namenjena tako posameznikom kot šolam, organizirajo poletne tabore ipd. Njihova gradiva naj bi že uporabljalo več kot osem in pol milijona učečih se. Poudarek je na aktivnem delu, programiranju iger, bogatih grafičnih vsebinah, nagrajevanju z značkami ipd. Programsko okolje je zelo podobno Scratchu, a je še bolj bogato, z več možnostmi.

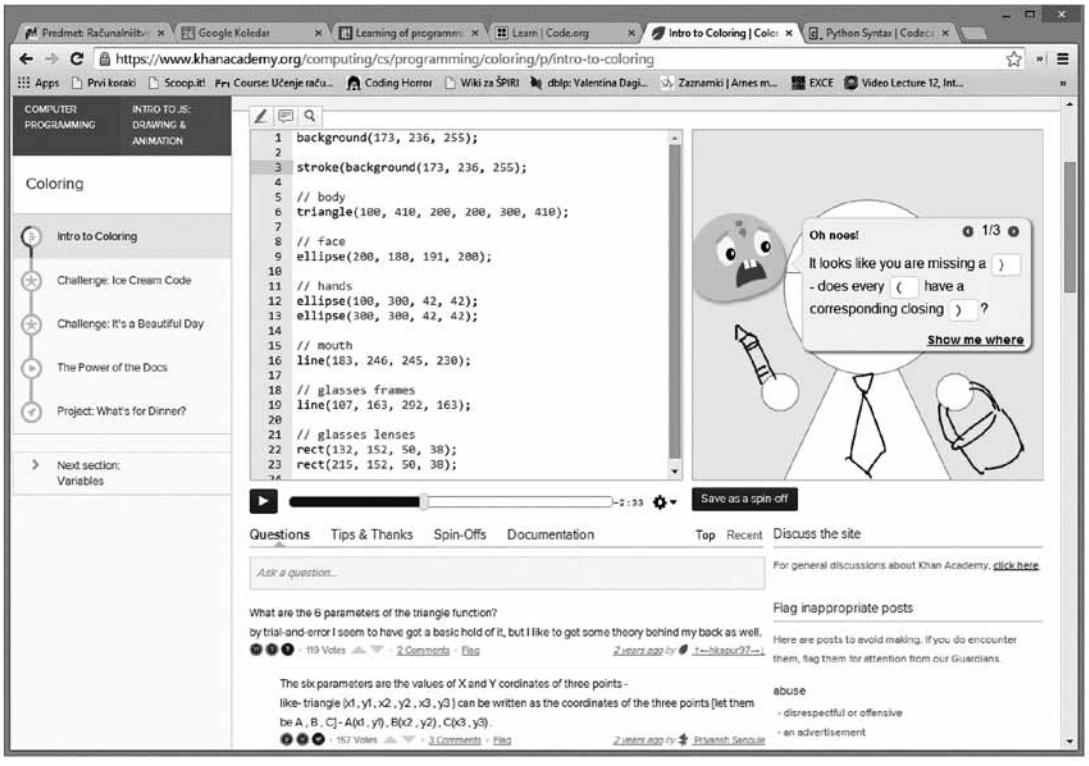
Codecademy (Codecademy, 2014) v nasprotju s prej omenjenimi spletnimi mesti, ki gradijo v glav-



Slika 21: **Tynker**

nem na programiranju v okoljih, podobnih Scratchu, »vztraja« na bolj »tradicionalnem« pristopu in jezikih. Tako se lahko učimo PHP, JavaScript, Python idr. Vsa koda se izvaja v nadzorovanem okolju, s polno namigi, pomoči. Tu so seveda nagrade v obliki značk, ki kažejo naš napredek. Spletno mesto, ki je vsekakor dobrodošlo za nekoliko starejše učence, ki jim jezik (angleščina ali španščina) ne bo povzročal težav in niso navdušeni nad »igračkastim« pristopom drugih okolij.

KhanAcademy (Khan Academy, 2014) je spletno mesto, ki bo vsekakor prišlo prav, če želimo učiti ali se naučiti programiranja prek uporabe JavaScripta. Zgledi so povezani z grafiko in zato vizualno privlačni. Vse je zasnovano na postopnem napredku, pri čemer v delu okna vidimo kodo, v delu pa njen učinek. Vse je podprto z video posnetki in razlago (žal le v angleščini).



Slika 22: **Khan Academy**

6 SKLEP

V zadnjem času se je na področju uvajanja programiranja zgodilo res veliko. Raziskave so pokazale koristnost učenja vsaj osnovnih programerskih korakov za vsakega učenca, razvoj pa je ponudil številne možnosti za vpeljavo teh korakov. Tako učenje in poučevanje programiranja lahko potekata na zanimiv način in omogočata ustrezne rezultate.

V prispevku so predstavljene številne možnosti za prve korake v svet programiranja. In katera je najprimernejša? Po mnenju avtorja odgovora na to ni. Vsak učeči se je svet zase. To, kar bo ustrezno za del učencev nekega razreda, že v tem razredu ne bo primerno za druge. Zato je vloga učitelja v tem procesu ob vseh tehnoloških možnostih in obilici gradiv še toliko bolj pomembna.

Potrebujemo veliko več vedenja o tem, kako učence voditi skozi proces spoznavanja, kako jih ustrezno usmerjati, kakšna gradiva in dejavnosti so primerne za posameznika z določenimi značilnostmi, kakšna je ustrezna povratna informacija, ki naj jo dajejo orodja ipd.

7 LITERATURA

[1] Aldrich, C. (2005). *Learning by doing: A comprehensive guide to simulations, computer games, and pedagogy in e-learning and other educational experiences.* John Wiley & Sons.

[2] Allcancode. (2014). *Run Marco!* Pridobljeno iz http://www.allcancode.com/.

[3] Batagelj, V. (1991). *Ferdinand in LOGO.* Ljubljana: Zavod RS za šolstvo in šport.

[4] Berkeley. (2014). *Berkeley Logo (UCBLogo)*. Pridobljeno s http://www.cs.berkeley.edu/~bh/logo.html.

[5] Berkeley. (2014). *Snap! Build Your Own Blocks*. Pridobljeno s http://snap.berkeley.edu/.

[6] Catrobat. (2014). *Pocket Code*. Pridobljeno s https://pocketcode.org/.

[7] Cliburn, D. (2006). Experiences with the LEGO Mindstorms throughout the Undergraduate Computer Science Curriculum. *Frontiers in Education Conference, 36th Annual* (str. 1–6). IEEE.

[8] Code.org – Hour of code. (2013). *Hour of code*. Retrieved from http://code.org/.

[9] code.org – The Flappy Code. (6 2014). *The Flappy Code*. Pridobljeno s http://learn.code.org/flappy/1.

[10] Code.org – The Maze. (2014). *The Maze*. Pridobljeno s http://learn.code.org/s/1/level/2.

[11] Code.org. (2014). *Code.org*. Pridobljeno s http://code.org/.

[12] Code.org. (2014). *Graph Paper Programming.* Pridobljeno s http://learn.code.org/unplugged/unplug3.pdf.

[13] Code.org. (2014). *Hour of Code, Dejstva in uporabna statistika.* Pridobljeno s http://hourofcode.com: http://hourofcode.com/si/resources/stats.

[14] Codecademy. (2014). *Codecademy – Learn to code interactively, for free*. Pridobljeno s http://www.codecademy.com/.

[15] Computer Science Unplugged. (2014). Pridobljeno s http://csunplugged.org/.

[16] Demšar, J. (2014). *VIDRA.* Pridobljeno s http://vidra.fri.uni-lj.si/.

[17] Fisher, C. R. (2014). Key-stage computing: Evaluating the suitability of Lego Mindstorms NXT 2.0 for use in early computer science education. *Discovery, Invention & Application, University of Derby*.

[18] FMSLogo. (2014). *FMSLogo: An Educational Programming Environment*. Pridobljeno s http://fmslogo.sourceforge.net/.

[19] Fowler, A., & Cusak, B. (2011). Enhancing Introductory Programming with Kodu Game Lab: An Exploratory Study. *2nd annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ2011)* (str. 69–79).

[20] Gamefroot. (2014). *Gamefroot*. Pridobljeno s http://gamefroot.com/.

[21] Google Play. (december 2012). *RoboZZle Droid*. Pridobljeno s https://play.google.com/store/apps/details?id=com.team242.robozzle.

[22] Google play. (2014). *Lightbot – Programming Puzzles*. Pridobljeno s https://play.google.com/store/apps/details?id=com.lightbot.lightbot.

[23] Google Projects – Blockly. (2014). *Blockly – A visual programming editor*. Pridobljeno s https://code.google.com/p/blockly/.

[24] Google Projects – Blockly. (2014). *Blockly – Code*. Pridobljeno s https://blockly-demo.appspot.com/static/apps/code/index.html.

[25] Google Projects – Blockly/Maze. (2014). *Blockly – Maze*. Pridobljeno s https://blockly-demo.appspot.com/static/apps/maze/index.html.

[26] Gruenbaum, P. (2014). Undergraduates Teach Game Programming Using Scratch. *IEEE Computer*.

[27] ICTMagic. (2014). *CodeCombat.* Pridobljeno s http://codecombat.com/play.

[28] Joshi, A. B. (2013). How Kids Learn When They Do Scratch Programming. *Tech Seva Conference 2013 in Pune India 1.1* (str. 1–6).

[29] Kafai, Y. (1995). *Minds in play: Computer game design as a context for children's learning.* Routledge.

[30] Kano. (2014). Pridobljeno s http://www.kano.me/.

[31] Khan Academy. (2014). *Khan Academy – Computer programming*. Pridobljeno s https://www.khanacademy.org/computing/cs.

[32] Kodu Game Lab. (2014). *Kodu Game Lab Community*. Pridobljeno s http://www.kodugamelab.com/.

[33] Kongregate. (2014). *Lightboot*. Pridobljeno s http://www.kongregate.com/games/coolio_niato/light-bot.

[34] Lawhead, P. e. (2003). A road map for teaching introductory programming using LEGO© mindstorms robots. *ACM SIGCSE Bulletin*, 191–201.

[35] LEGO. (2014). *Lego Mindstorms*. Pridobljeno s http://www.lego.com/en-us/mindstorms?icmp=COUSFRMindstorms.

[36] Lightboot Inc. (6 2014). *Lightbot – Programming Puzzles*. Pridobljeno s https://itunes.apple.com/us/app/light-bot/id657638474.

[37] Lokar, M. (2005). *Osnove programiranja – zakaj in vsaj kaj.* Zavod RS za šolstvo.

[38] Lokar, M. (2014). *Scratch – gradiva in prevod*. Pridobljeno s http://lokar.fmf.uni-lj.si/moodle/course/view.php?id=22.

[39] Marji, M. (2014). *Learn to Program with Scratch: A Visual Introduction to Programming with Games, Art, Science, and Math.* No Starch Press.

[40] Maxwell, B. (2013). *Robozzle*. Pridobljeno z iTunes: https://itunes.apple.com/us/app/robozzle/id350729261?mt=8.

[41] Mayer, R. E. (1988). *Teaching and Learning Computer Programming: Multiple Research Perspectives.* Routledge, dostopno v Google Books.

[42] McDougall, A., Murnane, J. S. & Wills, S. (2014). The Educational Programming Language Logo: Its Nature and Its Use in Australia. V A. Tatnall & B. Davey, *Reflections on the History of Computers in Education* (str. 394–407). Springer Berlin Heidelberg.

[43] Medlock-Walton, P. (2014). Blocks-based programming languages: simplifying programming for different audiences with different goals. *SIGCSE '14 Proceedings of the 45th ACM technical symposium on Computer science education* (str. 545–546). ACM.

[44] Microsoft Research. (2014). *Kodu*. Pridobljeno s http://research.microsoft.com/en-us/projects/kodu/.

[45] Microsoft Research. (2014). *PeX4Fun.* Pridobljeno s http://research.microsoft.com/en-us/projects/pex4fun/.

[46] MIT AppInventor. (2014). *MIT AppInventor*. Pridobljeno s http://appinventor.mit.edu/explore/.

[47] MIT Media Lab. (2014). *Design Blocks*. Pridobljeno s http://www.designblocks.net/.

[48] MIT Media Lab. (2014). *MIT App Inventor*. Pridobljeno s http://appinventor.mit.edu/.

[49] Ostrovsky, I. (2009). *RoboZZle Game*. Pridobljeno z YouTube: http://www.youtube.com/watch?v=MmqBVWi_Pc0.

[50] Pardamean, B. a. (2014). Enhancement of creativity through logo programming. *Am. J. Applied Sci.*, 528–533.

[51] Play-i. (6 2014). Pridobljeno s Play-I: https://www.play-i.com/.

[52] Prensky, M. (2005). Computer games and learning: Digital game-based learning. *Handbook of computer game studies*, 97–122.

[53] Resnick, M. (2009). Scratch: programming for all. *Communications of the ACM*, str. 60–67.

[54] Resnick, M. (2012). *Let's teach kids to code*. Pridobljeno s http://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code.

[55] RoboMind. (2014). *RoboMind.net – the new introduction to programming*. Pridobljeno s http://www.robomind.net/en/index.html.
[56] RoboZZle. (6 2014). *RoboZZle*. Pridobljeno s http://www.robozzle.com/.
[57] Royal Society. (2012). *Computing in Schools: Shut down or restart?* Pridobljeno s https://royalsociety.org/education/policy/computing-in-schools/report/.
[58] Scratch Group – MIT. (2014). *Scratch*. Pridobljeno iz http://scratch.mit.edu/.
[59] Shapiro, D. (2014). *The game for little programmers*. Pridobljeno z Robot Turtles: http://www.robotturtles.com/.
[60] Softronix. (2014). *MSW Logo*. Pridobljeno s http://www.softronix.com/logo.html.
[61] Solid Labs. (2014). *Primo*. Pridobljeno s http://primo.io/.
[62] Su, A. Y. (2014). Investigating the role of computer-supported annotation in problem-solving-based teaching: An empirical study of a Scratch programming pedagogy. *British Journal of Educational Technology*, 647–665.
[63] ThoughtSTEM. (2014). *CodeSpells: Express Yourself With Magic.* Pridobljeno s https://www.kickstarter.com/projects/thoughtstem/codespells-express-yourself-with-magic?ref=email.
[64] Tynker. (2014). *Tynker – Programming is the New Literacy*. Pridobljeno s http://www.tynker.com/.

▪

Matija Lokar je zaposlen na Fakulteti za matematiko in fiziko Univerze v Ljubljani kot višji predavatelj. Raziskovalno se ukvarja predvsem s problematiko uvajanja računalniške tehnologije v pouk matematike in s tehnikami poučevanja programiranja. Je avtor več knjig in člankov s področja računalništva in uvajanja računalniške tehnologije v pouk, prav tako pa številnih gradiv s tega področja. Sodeloval je pri številnih domačih in mednarodnih projektih na temo uporabe informacijsko-komunikacijske tehnologije v izobraževanju. Na matični fakulteti je več let sodeloval pri dopolnilnem izobraževanju učiteljev računalništva v osnovnih in srednjih šolah ali ga vodil ter predaval predvsem pri predmetih in tečajih s področja poučevanja programiranja.