Programski jezik Scratch i njegova primjena u osnovnoj školi

Budiša, Diana

SVEUČILIŠTE U ZAGREBU

UČITELJSKI FAKULTET

ODSJEK ZA UČITELJSKE STUDIJE

DIANA BUDIŠA

DIPLOMSKI RAD

PROGRAMSKI JEZIK *SCRATCH* I NJEGOVA PRIMJENA U OSNOVNOJ ŠKOLI

Čakovec, srpanj 2016.

SVEUČILIŠTE U ZAGREBU

UČITELJSKI FAKULTET

ODSJEK ZA UČITELJSKE STUDIJE

(Čakovec)

PREDMET: Programiranje

DIPLOMSKI RAD

Ime i prezime pristupnika: Diana Budiša

TEMA DIPLOMSKOGA RADA: Programski jezik *Scratch* i njegova primjena u osnovnoj školi

MENTOR: doc. dr. sc. Predrag Oreški

Čakovec, 2016.

SADRŽAJ:

SAŽETAK

Računalna tehnologija nezaustavljivom brzinom prodire u svaki djelić ljudskog života i rada. Djeca nove tehnologije primaju na drugačiji način od odraslih, jer odrastaju s novim izazovima. Uloga učitelja je usmjeriti učenike kako kreativno koristiti informacijsku tehnologiju te njome podignuti razinu obrazovanja u cjelini.

Danas je nastava informatike izborni predmet u onim osnovnim školama koje za to ispunjavaju tehničke i kadrovske uvjete. Teži se tome da se sljedećih godina informatika uvede kao obavezni predmet od 1. razreda osnovne škole. Informatika je idealno područje kroz koje se kod djece mogu razvijati vještine koje su nužne za cjeloživotno učenje, poput logičkog, stvaralačkog i apstraktnog mišljenja, kreativnosti, učenja metodom pokušaja i pogrešaka, stalnim dopunjavanjem i ispravljanjem vlastitog rješenja te korištenjem različitih izvora znanja.

Korištenje osnovnih alata za rad na računalu je svakako korisno, kao što je korisno i poznavati osnove programiranja. I dok danas djeca, sutra odrasli, možda neće kroz život morati programirati, dobro će im doći praktično znanje o načinu na koji funkcioniraju tehnologije koje koriste i okvirno shvaćanje toga što s njima mogu postići. Jedan od alata za početno učenje programiranja je *Scratch*, programski jezik i razvojno okruženje za stvaranje priča, igara, animacija, glazbe i drugih interaktivnih sadržaja na računalu, posebno osmišljen kako bi bio pristupačan i zanimljiv djeci. U ovom radu bit će objašnjen način programiranja aplikacija u *Scratchu,* te će na jednostavnim primjerima biti prikazana struktura programa sastavljena od grafičkih blokova koji predstavljaju naredbe. Također, na primjerima će biti prikazane mogućnosti korištenja ovog programa u osnovnoj školi u pojedinim predmetima.

KLJUČNE RIJEČI: *Scratch, programiranje, apstraktno mišljenje*

SUMMARY

Computer technology is spreading at an unstoppable pace into every part of human life and work. Today, children are getting familiar with new technologies in a different way than adults, because they are growing up with the new challenges. The role of the teacher is to direct students to use information technology creatively and, thus, raise the level of education in general.

Today's computer science classes are offered to students as an elective subject in primary schools that meet the technical and personnel requirements. The main goal is to introduce information technology in the coming years as a compulsory subject in the first grade of primary school. Informatics is an ideal area through which children can develop the skills necessary for lifelong learning, such as logical, creative and abstract thinking, creativity, learning by trial and error, upgrading and perfecting their own solutions continuously and using different sources of knowledge.

Knowledge and ability to use essential tools and related technology efficiently, are today's must have skills, as well as possessing a range of skills covering levels from basic programming techniques to advanced problem solving. While today's children are tomorrow's adults and may not have to be programmers for a living, they will get practical knowledge on how the technology they use actually works and the indicative understanding of what can be achieved with it. One of the tools for initial learning of programming is *Scratch*, a programming language and development environment for the creation of stories, games, animation, music and other interactive content on the computer, specifically designed to be accessible and interesting to children. This paper will explain how to build applications by programming in *Scratch*, and the simple examples will be shown through the program structure which consists mainly of graphical blocks that represent commands. Also, the examples included in this paper demonstrate the possibility for use of *Scratch* with primary schools in certain subjects.

KEY WORDS: *Scratch, programming, abstract thinking*

1. UVOD

Podučavanje informatike za učenike od prvog do četvrtog razreda u Hrvatskoj se još uvijek ne odvija u okviru obveznog predmeta, već se u školama koje imaju tehničkih mogućnosti i obrazovanog kadra održava u sklopu izvannastavnih aktivnosti.

Nastavni sadržaji predmeta *Informatika* podijeljeni su u devet programskih cjelina s radnim naslovima: *1. Osnove informacijske i komunikacijske tehnologije, 2. Strojna i programska oprema računala, 3. Multimediji, 4. Obrada teksta, 5. Proračunske tablice i baze podataka, 6. Izrada prezentacija, 7. Izrada Web stranica, 8. Rješavanje problema i programiranje* i *9. Internet*. Od tih devet programskih cjelina u prva četiri razreda obrađuju se postupno pet cjelina tako da se postigne kružno širenje znanja u vertikali školovanja (HNOS).

Prema novom *Nacionalnom okvirnom kurikulumu* slijedi uvođenje redovitog obveznog predmeta *Informatika* u osnovno i srednje obrazovanje s postignućima koja, s jedne strane osiguravaju naprednu razinu digitalne pismenosti kako bi se učenicima omogućila djelotvorna uporaba informacijske i komunikacijske tehnologije, a s druge strane osposobljavaju učenike za algoritamski način razmišljanja, rješavanje problema računalom kroz izradu računalnih programa i kreativno stvaranje novih rješenja u području informacijske i komunikacijske tehnologije. Informatika se, bez izuzetka, primjenjuje u svim područjima ljudske djelatnosti, zbog toga je u europskom kompetencijskom okviru ovladavanje njome svrstano u jednu od osam ključnih kompetencija, tzv. digitalne kompetencije (MZOŠ, 2011, str. 160).

Za razumijevanje tehnologije, potrebno je razumjeti jezik računala, odnosno ovladati principom rada računala. Najbolji način za to je učenje programiranja od najranije dobi, ali na način koji je djeci „prirodan“. Činjenicu da djeca mnogo svog vremena provode za računalom potrebno je iskoristiti kako bi to vrijeme mogli provesti učeći (Bubica, N., Mladenović, M., Boljat, I., 2014). Ideja o tome da učenici nižih razreda osnovne škole uče osnovne koncepte programiranja nije nova, a edukacijska vrijednost učenja programiranja opisana je u raznim istraživanjima (Đurđević, 2013).

Programiranje nije samo pisanje računalnih programa. Programiranje je rješavanje problema, otklanjanje grešaka, razvijanje logičkog razmišljanja i računalnog razmišljanja, a to podrazumijeva razvoj strategija za rješavanje problema koji se mogu odnositi i na neprogramerska područja. Zbog toga se može reći da programiranje mijenja način razmišljanja. Kada uzmemo u obzir da živimo u digitalnom dobu i da je djeci tehnologija „prirodno“ okruženje, jasno je da bi takve promjene trebali uzeti u obzir u školovanju, s obzirom na sveprisutnost informatike u svakodnevnom životu i s pretpostavkom da će ta prisutnost postajati sve izraženija (Bubica, N., Mladenović, M., Boljat, I., 2014).

Programiranje se ne promatra samo kao vještina koju je danas poželjno svladati, već kao alat pomoću kojeg kod učenika razvijamo metakognitivne sposobnosti koje su temelj uspješnosti u daljem školovanju. Ostaje pitanje kako danas poučavati ovakve vještine?

Programi koji se uče u hrvatskim školama, kao što su *QBasic*, *Logo* i *Pascal*, pomažu djeci da nauče osnove programiranja i logičkog razmišljanja. Međutim, zbog svoje složene sintakse nisu prilagođeni učenicima razredne nastave. Upravo zbog toga razvio se programski jezik *Scratch* za poučavanje programiranja, prilagođen djeci već od predškolske dobi. Programiranje u *Scratchu* temelji se na grafičkom povezivanju prethodno nacrtanih blokova (kao LEGO kockice). Različiti tipovi podataka su prikazani različitim oblicima blokova gdje se dijelovi slažu tako da se spoje na sintaktički pravilan način. Takav pristup radu onemogućuje sintaktičke greške (greške nastale ne poštivanjem propisanih pravila pisanja programa- greške pri korištenju naredbi, tipfeleri, itd.) i fokusira pažnju učenika na problem koji bi treba riješiti, a ne na samu mehaniku programiranja. Ova bi se vrsta programa u hrvatskim školama mogla primijeniti u nastavi informatike od drugog do četvrtog razreda. Pomoću njega učenici bi mogli raditi na raznim projektima i učiti širok spektar aktivnosti iz različitih predmeta te tako proširivati svoje znanje.

Struktura programskog jezika *Scratch*, kao i mogućnosti te način primjene u osnovnoj školi bit će predstavljene u sljedećim poglavljima rada.

2. PROGRAMIRANJE I APSTRAKTNO MIŠLJENJE

Prema Piagetu, postoje četiri faze kognitivnog razvoja čovjeka: senzomotorna (0-2g.), predoperacijska (2.-7. g), faza konkretnih operacija (7.-11.g) i formalnih operacija (>12 g). Odlazak u školu uobičajeno se događa u fazi konkretnih operacija gdje je dijete u stanju logički pristupiti rješavanju problema. Vrijeme ulaska u fazu formalnih operacija varira, dobar dio ljudi nikada ne dođe do ove faze koja je, zapravo, sposobnost apstraktnog mišljenja. Očito je da je ljudima prirodno razmišljati od konkretnog prema apstraktnom (Bubica i sur., 2014, str. 2).

Kroz svoju knjigu *Mindstorms,* Papert (1980) pokazuje kako računalo može doprinijeti mentalnim procesima ne samo instrumentalno već esencijalno, na konceptualnoj razini i to na način na koji ljudi razmišljaju i kada nisu u interakciji s računalima. Odrastanjem se razvija sposobnost apstraktnog mišljenja, zaključivanje, generaliziranje i sl. Ove sposobnosti se razvijaju sporo i potrebno ih je redovito vježbati. Konkretna iskustva su najučinkovitiji način učenja kada se pojavljuju u kontekstu relevantne konceptualne strukture (Bubica i sur., 2014, str. 2-3).

Programiranje je samo po sebi potpuno apstraktno i samim time teško za razumijevanje, ali je u isto vrijeme dobar alat za vježbu i razvoj apstraktnog razmišljanja. Kako djeca na početku osnovne škole, barem većina, nemaju mogućnost apstraktnog mišljenja, učenje programiranja čini im se teškim. Ako govorimo o djeci u prvom razredu osnovne škole, problem je još veći jer djeca u prvom razredu još ne znaju čitati ni pisati. Kako ih onda učiti programirati? Kao odgovor na ovaj prepoznati problem pojavljuju se vizualni programski jezici. Oni, osim uklanjanja problema sintakse, omogućuju učenje programiranja u konkretnom okruženju, gdje apstraktni pojmovi poput varijabli, petlji i sl. u vizualnom okruženju pružaju konkretno iskustvo. (Bubica i sur., 2014). Programiranje u grafičkim okruženjima u kojima ne postoje sintaktičke pogreške može ublažiti stresan početak učenja programiranja. (Bubica, 2014, str. 8). Vizualno programiranje danas se koristi za početno učenje koncepata računalne znanosti i programiranja jer podupire učenje putem istraživanja (Bubica i sur., 2014, str. 5).

Učenje i poučavanje programiranja je izazov za nastavnike i učenike, bez obzira na uzrast učenika (radi li se o osnovnoj, srednjoj školi ili fakultetu), a odabir programskog jezika za poučavanje početnika nije jednostavan problem. Početni programski jezik bi morao imati jednostavnu sintaksu, brzu povratnu informaciju i strukturirani način pisanja. (Krpan, D., Mladenović, S., Zaharija, G., 2014). Naglasak u programiranju kod djece treba biti na logici i računalnom razmišljanju. Kod programiranja se problem gotovo nikada ne riješi od prve. „Učiti" za programera znači ovladati višim vještinama izoliranja i ispravljanja pogrešaka *(„bug-ova"*) zbog kojih program ne radi. Kod novih alata za vizualno programiranje, kao što je *Scratch,* naglasak je na rješavanju problema gdje učenici mogu programirati u kontekstu (realni/stvarni svijet). Ovakva okruženja, osim pristupa licem u lice (*eng. face-to-face*), pružaju i mogućnost udaljene komunikacije u obliku „galerija", internetskih stranica na kojima učenici mogu postaviti i dijeliti svoje radove, što djeluje poticajno. (Bubica i sur., 2014, str. 5).

Istraživanje provedeno na predškolskom uzrastu djece starosti od 5 do 6 godina vezano za aspekte rješavanja problema uz korištenje računalnog programiranja pokazuje da su djeca aktivno sudjelovala i uživala u zanimljivim aktivnostima u kojima su imala mogućnost razvijanja matematičkih koncepata, strategija rješavanja problema i socijalnih vještina (komunikacijskih i kolaboracijskih vještina) (Fessakis, G., Gouli, E., Mavroudi, E., 2013).

3. PROGRAMSKI JEZIK *SCRATCH*

Scratch je novi programski jezik otvorenog koda razvijen 2003. godine na Tehnološkom institutu države Massachusetts (*Massachusetts Institute of Technology* – MIT) kao projekt *Lifelong Kindergarten* grupe, a omogućuje lako kreiranje interaktivnih priča, igrica, animacija i projekata te njihovo dijeljenje s drugim korisnicima preko weba. (Mujačić, 2015). Grupa *Lifelong Kindergarten* surađivala je s tvrtkom Lego na izradi robota Lego Mindstorms koji se, također, koriste za poučavanje programiranja. Uočilo se da djeci u radu sa kockicama odmah počinju navirati ideje, mašta i kreativnost pa su napravili vizualni programski jezik koji podsjeća na slaganje kockica. Naredbe su napravljene u obliku slagalica, tako da je vizualno jasno koje se naredbe mogu složiti. Grupirane su tematski, a razlikuju se po obliku i bojama (Bubica i sur., 2014).

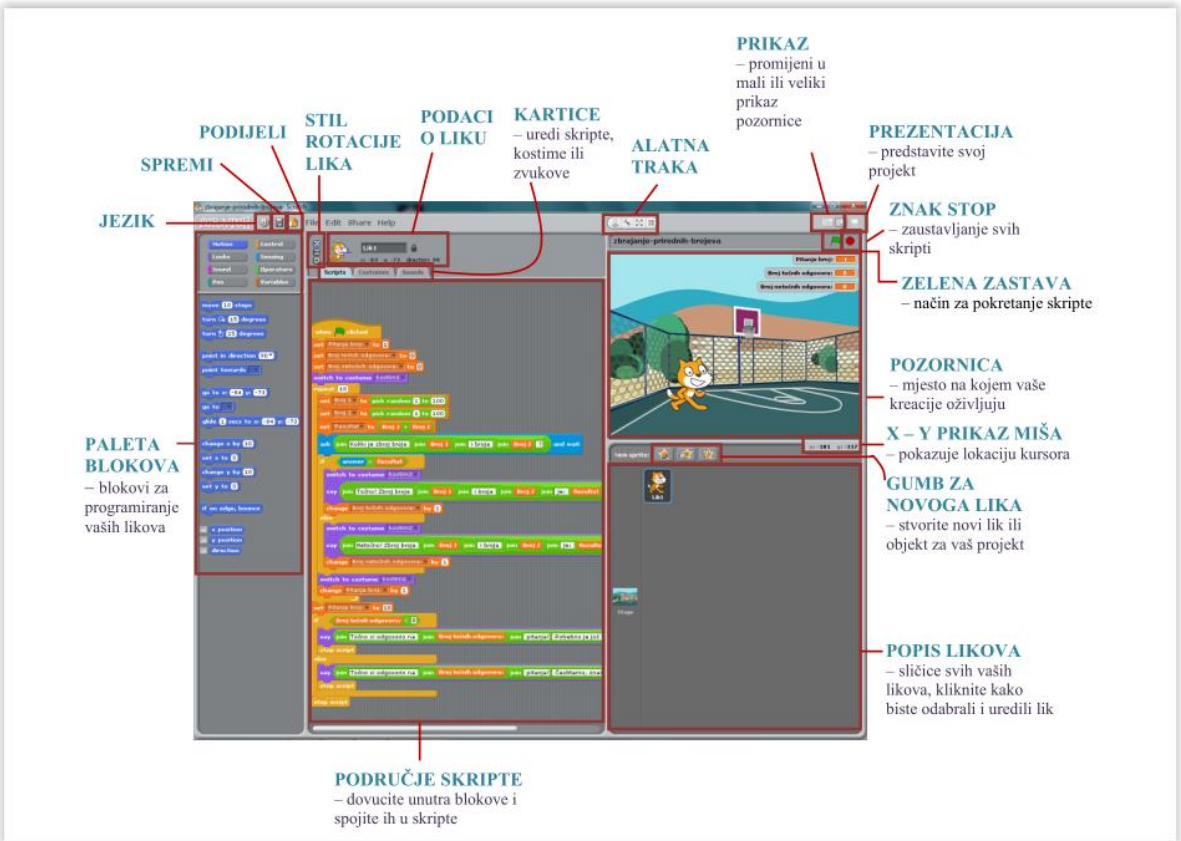
Projekt je razvijen u duhu proširenja vidika djece i poticaja djece na istraživanje, osmišljavanje i ostvarivanje svojih zamisli, koje zatim mogu pokazati drugima putem interneta, usput usvajajući osnove programiranja, matematike te vizualnog i interaktivnog dizajna (Valić, B., Radovan, A., Pavlović, D., 2013).

Scratch omogućuje rad u višejezičnom okruženju, uključujući sučelje i na hrvatskom jeziku, a može se instalirati na operacijske sustave *Windows, MAC i Linux*. Ima dobro razvijenu zajednicu korisnika koji postavljaju svoje uratke (projekte) na njegove službene mrežne stranice te omogućuju njihovo besplatno preuzimanje i korištenje. Iako je nastao na idejama programa *LOGO*, programiranje u *Scratchu* razlikuje se od programiranja u drugim vizualnim programskim okolinama, jer se u njemu rabe naredbene strukture u obliku grafičkih programskih blokova, pa se na taj način eliminira mogućnost sintaksnih pogrešaka (Peppler, Kafai, 2005). Program omogućuje programiranje mišem povlačenjem i uklapanjem blokova koji se mogu spojiti samo ako to odgovara u određenom sintaksnom smislu i na taj način omogućuje učenicima da se fokusiraju na probleme koje oni žele riješiti. Nastao je kao pomoć za razvijanje osnovnih vještina koje autori programa smatraju iznimno bitnim za 21. stoljeće, kao što su kreativno razmišljanje, jasna komunikacija, sistematična analiza, efikasna kolaboracija i kontinuirano učenje (Đurđević, 2013).

Autori *Scratcha* napominju da je taj program namijenjen početnicima u programiranju od navršenih osam godina nadalje i da se s radom u programu započinje lako, premda program omogućuje kreiranje složenijih projekata, a ima široku korisničku podršku za različite projekte (Đurđević, 2013).

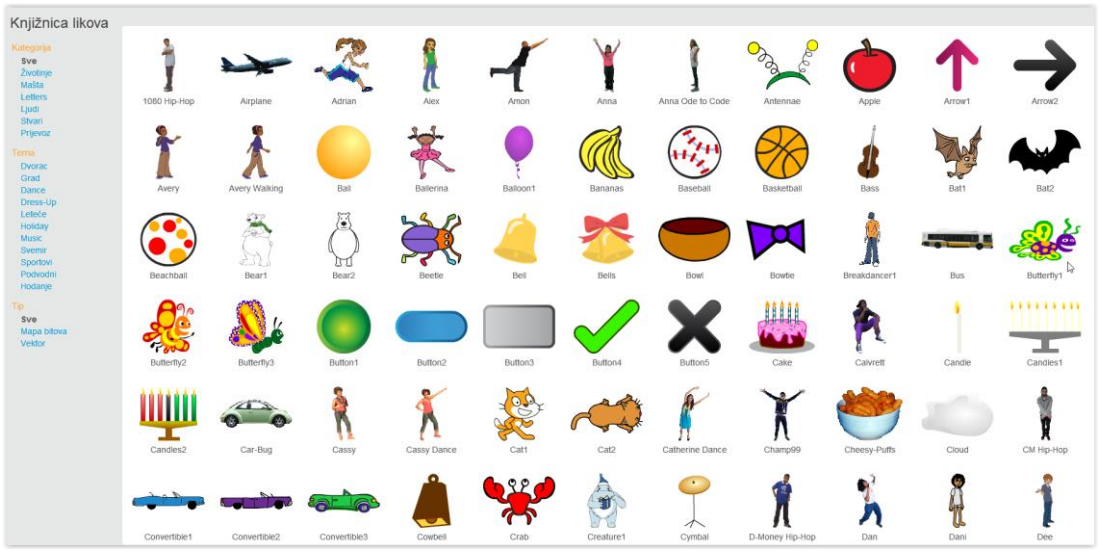
3.1. Osnovni elementi *Scratch* projekta

Scratch razvojno okruženje nalazi se na jednom prozoru i elementi su fiksni s više pojedinih dijelova. Na desnoj strani postoji mjesto gdje se izvodi program, ispod toga se nalazi popis objekata u programu, na lijevoj strani se nalazi dio s pomoćnim blokovima i opcijama, a u sredini je prostor za slaganje skripti (raspored dijelova može biti i obrnut, tako da se mjesto gdje se izvodi program nalazi na lijevoj strani) (Valić, B., Radovan, A., Pavlović, D., 2013).



Slika 1. Sučelje *Scratch* programa (ODRAZI, 2011)

Scratch projekt čine objekti koji se zovu likovi (*sprites*). Njihov se izgled može mijenjati dodavanjem različitih kostima (*costumes*). Lik može izgledati poput osobe, vlaka, leptira ili bilo čega drugoga. Za kostim je moguće koristiti bilo koju sliku; izraditi crtež u u Paint Editor-u, može se učitati slika s računala ili metodom *uhvati-povuci-pusti ('drag and drop')* preuzeti slika s internetske stranice. Liku se mogu dati različite upute: da se kreće, razgovara, reproducira glazbu ili ostvaruje interakciju s drugim likovima. Za izdavanje naredbi potrebno je složiti grafičke blokove u cjeline, nazvane skripte. (Otvoreno društvo za razmjenu ideja (ODRAZI), 2011).

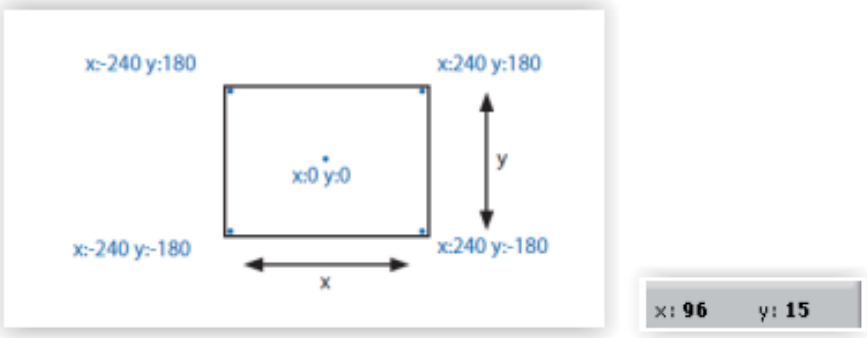


Slika 2. Knjižnica likova (*Scratch 2*)

Scratch blokovi organizirani su u osam kategorija različitih boja: *Kretanje, Izgled, Zvuk, Olovka, Upravljanje, Očitanja, Operacije i Varijable*. Svaka vrsta blokova ima jedinstven izgled kojim pokazuje koji blok ide ispod, iznad ili unutar na način da se blokovi spajaju kao slagalice. Svojim izgledom blokovi sugeriraju koja su ograničenja i mogućnosti te na taj način onemogućavaju javljanje pogrešaka. Iako nema dojave pogrešaka, sam program može funkcionirati na „krivi način“, ali eksperimentiranjem i iskustvom može se doći do željenih rezultata. Klikom na skriptu, *Scratch* prolazi kroz blokove od vrha do dna. Ako korisnik samo želi provjeriti kako djeluje određeni blok, dovoljno je kliknuti na njega i lik na ekranu će naredbu odmah izvesti. Skripte koje se grade uvijek se mogu izvesti te pojedini blokovi neće smetati ako se odvoje van skripte. (Valić i sur., 2013).

Slika 3. Paleta blokova (Valić i sur., 2013).

Likovi kojima se daju naredbe nalaze se na pozornici. Na pozornici se likovi pomiču i ostvaruju interakciju, a sam izgled pozornice može se mijenjati. Pozornica je široka 480, a dugačka 360 jedinica. Podijeljena je na x–y os. Središte Pozornice ima x-koordinatu 0 i y-koordinatu 0. Kako bi se otkrile x–y koordinate na Pozornici, potrebno je pomicati miš (kursor) i pratiti x–y prikaz koji se nalazi ispod Pozornice (ODRAZI, 2011).



Slika 4. Prikaz širine i *x-y* koordinata Pozornice (ODRAZI, 2011)

3.2. Kostimi

Klikom na karticu *Kostimi* moguće je vidjeti kostime likova i urediti ih.

Slika 5. Kostimi likova (Scratch)

Ovaj početni lik na slici 5. ima dva kostima. Osvijetljen je njegov trenutačni kostim (kostim1). Kako bi se promijenio kostim potrebno je kliknuti na sličicu željenoga kostima.

Četiri su načina stvaranja novih kostima:

- 

Klikom na „nacrtaj“ može se nacrtati novi kostim u Paint Editoru.
- 

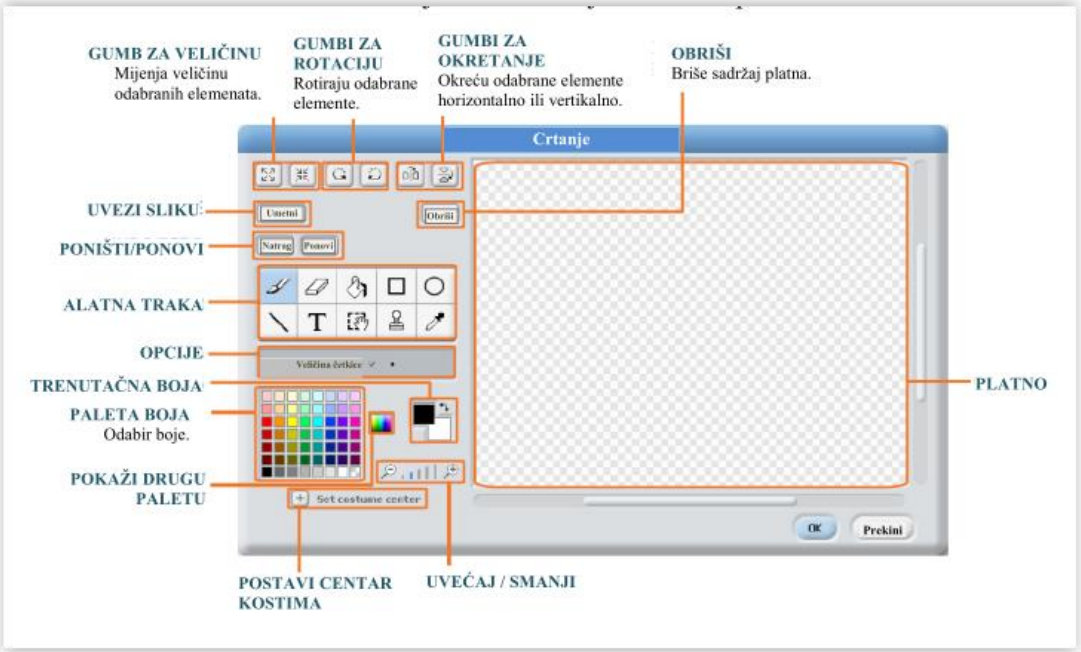
Klikom na „umetni“ može se umetnuti slika s tvrdog diska.
- 

Klikom na „kameru“ može se fotografirati web kamerom.
- Četvrti način je dovući jednu ili više slika s Interneta ili radne površine (desktop).

Scratch prepoznaje brojne formate slika: JPG, BMP, PNG, GIF (uključujući i animirani GIF). Svaki kostim ima broj koji je prikazan s njegove lijeve strane. Redoslijed kostima moguće je promijeniti povlačenjem sličica, a njihov će se broj automatski ažurirati ovisno o položaju. (ODRAZI, 2011).

3.3. Paint Editor

Paint Editor koristi se za stvaranje ili uređivanje kostima i pozadine.



Slika 6. Paint editor (ODRAZI, 2011)

U Paint editoru mogu se koristiti sljedeći alati: *kist, gumica, ispuni, pravokutnik, elipsa, crta, tekst, odabir, žig, kapaljka* (slično kao i u samom programu *Paint*). Trenutačne boje (primarna boja i boja pozadine) prikazane su ispod područja opcija. Klikom na *Postavi centar kostima* odabire se mjesto koje će biti središte rotacije kada se kostim bude okretao na Pozornici. Klikom na gumb „Zoom“ može se smanjiti ili povećati uvećanje Platna. Kada je zoom veći od 100%, klizne trake služe za pomicanje oko platna. Zoom ne mijenja veličinu slike. Ako želimo zarotirati sadržaj platna ili samo odabrane elemente, potrebno je kliknuti na gumb „Rotiraj“ (vodoravno ili okomito), slično je i za gumb Zaokreni. Pritiskom na gumb „Očisti“ mičemo sve sadržaje s platna. Ako pogriješimo u radu, klikom na „Poništi“ možemo poništiti prethodnu radnju. Ako se predomislimo, uvijek možemo kliknuti na „Ponovi“ kako bismo vratili poništene radnje. (ODRAZI, 2011, str. 12).

3.4. Vrste blokova

Tri su glavne vrste blokova u *Paleti blokova*.

SLOG BLOKOVI: Ovi blokovi imaju ispupčenja na dnu i/ili udubljenja na vrhu poput 

. Ovi se blokovi mogu slagati u slogove. Neki slog-blokovi imaju predviđen prostor za unos podataka gdje se može unesti broj (npr. 10 u bloku 

) ili odabrati element iz padajućeg izbornika (npr. 

).

Neki slog blokovi, poput, 

, imaju „usta“ u obliku slova C, gdje je moguće umetnuti druge slog-blokove. (ODRAZI, 2011, str. 13).

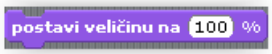
ŠEŠIRI: Ovakvi blokovi imaju zaobljene vrhove, poput 

. Smještaju se na vrh slogova gdje čekaju neku radnju, primjerice pritiskanje određene tipke, nakon čega izvršavaju radnju bloka ispod njih. (ODRAZI, 2011, str. 13).

REPORTERI: Ovakvi blokovi, kao 

i 

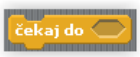
, kreirani su kako bi stali u upisni dio drugih blokova. Reporteri sa zaobljenim krajevima (kao 

) dojavljuju brojeve ili nizove i pristaju u zaobljene i pravokutne praznine (poput 

ili 

. (ODRAZI, 2011, str. 14).

Reporteri s oštrim rubovima (kao 

) dojavljuju boolean vrijednosti (točno ili netočno) i pristaju u blokove s oštrim ili pravokutnim prazninama (poput 

ili 

. Neki reporteri pored sebe imaju tzv. checkbox, odnosno kutiju za označivanje, nalik ovoj 

. Klikom na prazninu, na pozornici se pojavljuje monitor koji prikazuje trenutačnu vrijednost reportera. Monitor se automatski ažurira, ovisno o promjenama te vrijednosti (

).

Prikaz je moguć u više različitih formata:

mala znamenka s imenom reportera

velika znamenka, bez imena

klizni pokazivač koji omogućuje promjenu vrijednosti reportera (dostupno samo za varijable)

Format prikaza mijenja se dvostrukim klikom na monitor ili desnim klikom (Mac: Ctrl+klik). Klizni pokazivač dostupan je samo za varijable koje je stvorio korisnik. Klikom desne tipke miša (Mac: Ctrl+klik) na monitor u kliznom formatu može se prilagoditi minimalna i maksimalna vrijednost. (ODRAZI, 2011, str. 14).

ZELENA ZASTAVA

Zelena je zastava praktičan način istovremenog korištenja više skripti.

Klikom na zelenu zastavu  pokreću se sve skripte koje imaju ovaj znak 

na vrhu. Zelena zastava bit će osvijetljena sve dok se skripte izvode. U prezentacijskom izgledu ista opcija može se koristiti pritiskom tipke Enter. Na *Scratch* internetskoj stranici zelena se zastava automatski pali prilikom pregledavanja projekta. (ODRAZI, 2011, str. 10).

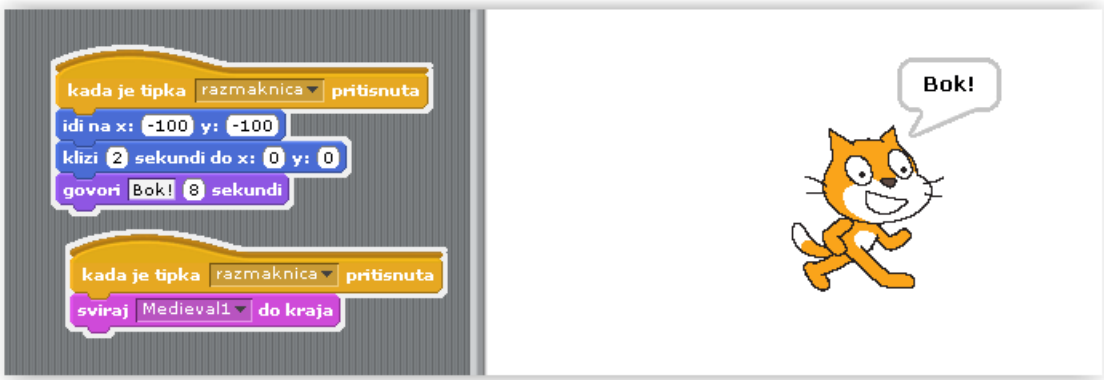
4. PROGRAMSKE STRUKTURE U *SCRATCHU*

Programska struktura je ustrojstvo programa, tj. način i redoslijed rješavanja pojedinih manjih radnji da bi se došlo do konačnog rješenja zadatka. Nekoliko je osnovnih programskih struktura koje se pojavljuju u mnogim računalnim programima pa tako i u *Scratchu*. To su:

- pravocrtna programska struktura (slijed/sekvencija)
- struktura grananja (odabir/selekcija)
- struktura petlje (ponavljanje/iteracija)

4.1. Pravocrtna programska struktura - Slijed ili sekvencija

Slijed ili sekvencija *(eng. Sequention)* je najjednostavnija logička struktura u kojoj se naredbe odvijaju jedna iza druge, onim redom kako su napisane, a formiraju se u obliku bloka naredbi (Lipljin, 2004, str. 111).



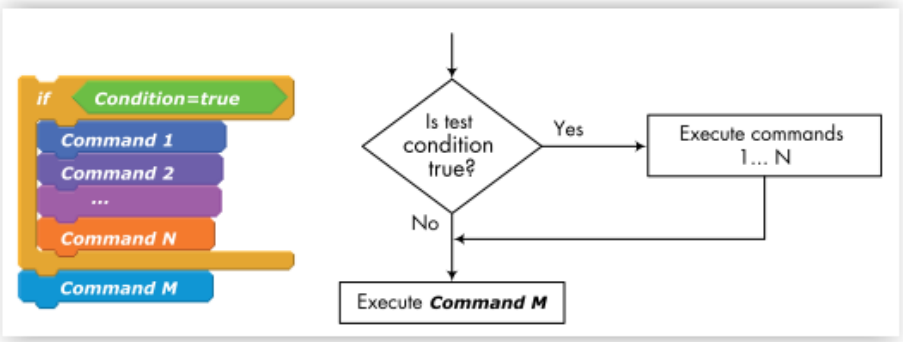
Slika 7. Primjer izvođenja programa slijedom naredbi

4.2. Struktura grananja – Odabir ili selekcija

Odabir ili selekcija *(eng. Selection)* je mjesto u programu na kojem se ispituje neka vrijednost ili logički izraz koji može biti jednostavan ili složen. Ovisno o rezultatu ispitivanja, dolazi do grananja ili do jednostrane selekcije (Lipljin, 2004, str. 111). Grananje je programska struktura koja omogućuje različit tijek programa, ovisan o rezultatu postavljenog uvjeta. To je važna struktura bez koje bi mogućnost rješavanja zadataka računalom bila vrlo ograničena. Rezultat postavljenog uvjeta mora biti jedno od dva stanja: true/false, da/ne, istina/laž, 1/0. Vrijednost rezultata uvjeta bit će 1 ako je uvjet zadovoljen i 0 ako uvjet nije zadovoljen.

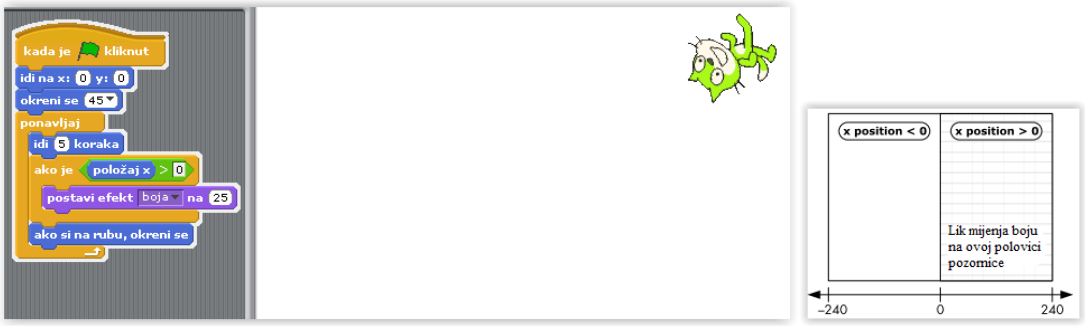
4.2.1. NAREDBA *IF*

If naredba daje mogućnost određivanja je li skup naredbi treba (ili ne treba) biti izvršen na temelju rezultata ispitivanog stanja. Naredba *If* radi tako da se ispita uvjet i u skladu sa istinitom ili lažnom vrijednošću uvjetnog izraza izvršavaju naredbe. Ukoliko je uvjet istinit, naredba ili blok naredbi koje slijede se izvršavaju, a ako je neistinit naredbe se preskaču.



Slika 8. Struktura *If* naredbe (Marji, Learn to program with Scratch, 2014., str. 128).

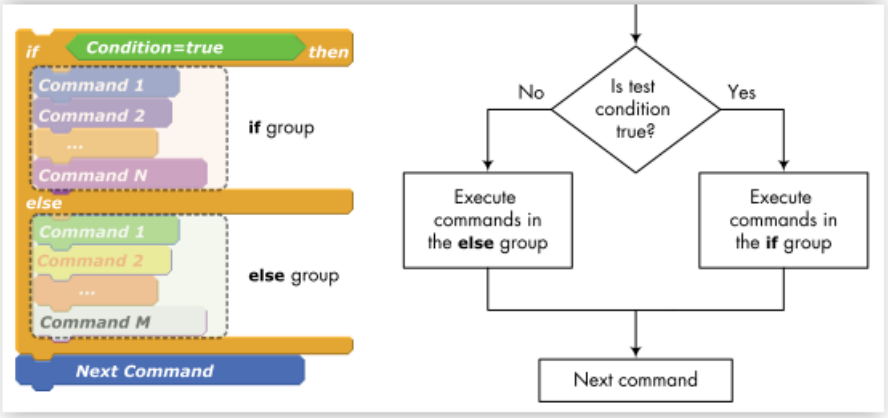
Ova skripta pokreće lik koji mijenja boju samo kad se nalazi na desnoj strani pozornice (kad je koordinata x veća od 0 = uvjet).



Slika 9. Primjer *If* naredbe - mijenjanje boje lika ovisno o položaju (koordinatama)

4.2.2. NAREDBA *IF/ELSE*

Naredba *If/else* odnosno dvostruko uvjetno grananje omogućava da se ovisno o ispunjenju postavljenog uvjeta izvodi jedan od dva neovisna bloka naredbi. Ako je vrijednost uvjeta istina, izvodi se prvi blok naredbi. Ako je vrijednost uvjeta neistina, preskače se prvi blok naredbi i izvodi se drugi blok (iza naredbe else).



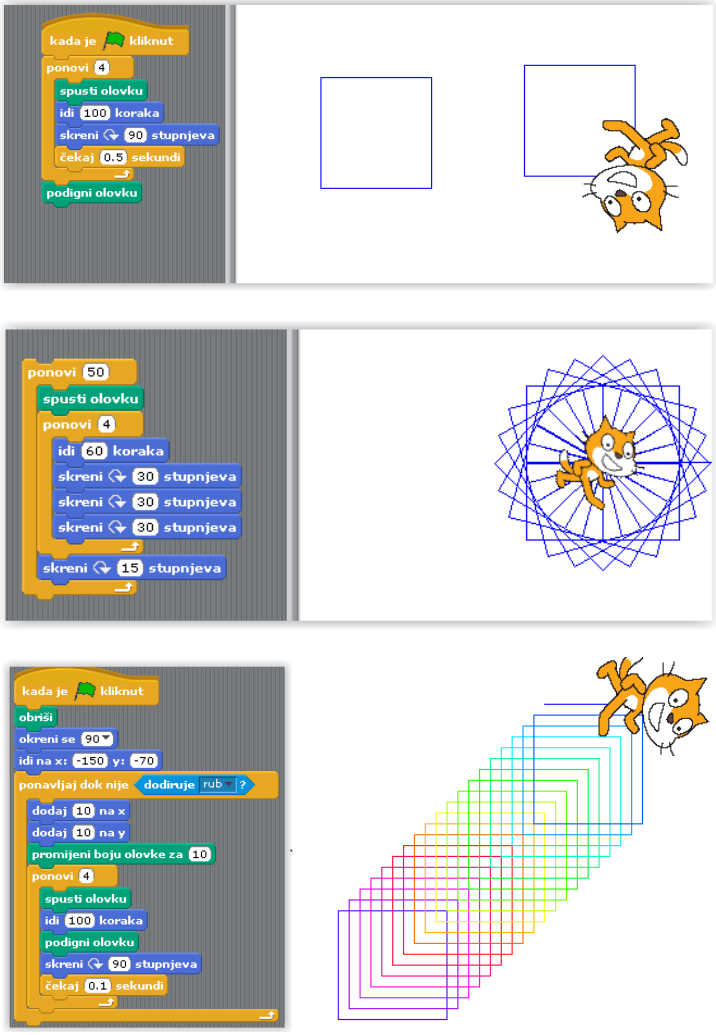
Slika 10. Struktura *If/else* naredbe (Marji, Learn to program with Scratch, 2014., str. 131).

U navedenom primjeru korišten je *mod operator* koji vraća cjelobrojni ostatak dijeljenja dva cijela broja, kako bi utvrdio je li unesen broj paran ili neparan.

4.3. Struktura petlje – Ponavljanje ili iteracija

Ako se neki dijelovi programa ponavljaju određen broj puta, govori se o logičkoj strukturi ponavljanja ili iteracije (*eng. Loop*) (Lipljin, 2004, str. 111). Često je u programu potrebno ponoviti neku radnju više puta. Da se ne bi ista naredba pisala više puta, postoji programska struktura petlje koja omogućava ponavljanje jedne ili više naredbi. Petlja može biti: bezuvjetna, pri čemu se izvršava unaprijed zadan broj puta, i uvjetna pri čemu broj ponavljanja petlje ovisi o postavljenom uvjetu. (Lipljin, 2004).

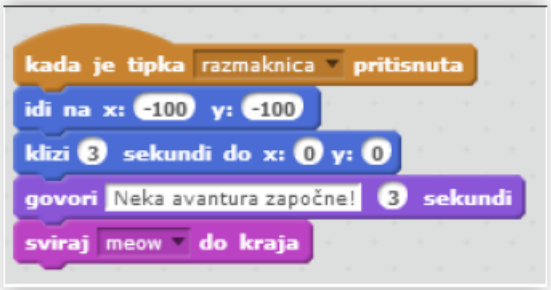
Kvadrat, na primjer, ima četiri stranice i može se nacrtati ponavljanjem sekvence: „idi za 100 koraka, okreni se za 90 stupnjeva i tako četiri puta“. Može se eksperimentirati s različitim kutovima, bojama i pomacima kako bi se dobili zanimljivi oblici.

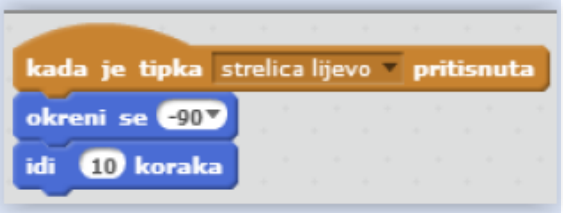


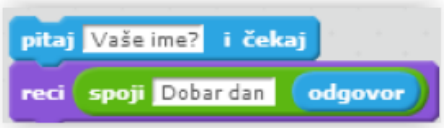
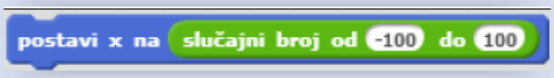
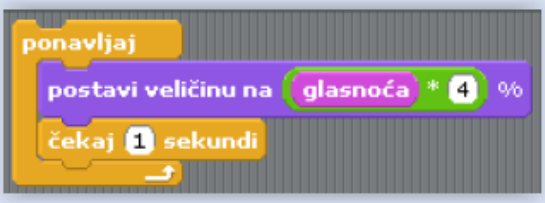
Slika 11. Primjeri petlje (ponavljanja) – kvadrat

4.4. Primjeri koncepata u *Scratchu*

U tablici ispod prikazani su primjeri sastavljanja osnovnih koncepata u programiranju i način njihovog oblikovanja u *Scratchu*.

KONCEPT	OBJAŠNJENJE	PRIMJER
Sekvenca	Za kreiranje programa u *Scratchu* treba sistematično razmišljati o poretku koraka.	
Iteracija	„Ponovi“ i „ponavljaj“ koriste se za iteraciju i ponavljanje serija instrukcija.	
Uvjetne naredbe	„Ako je“ i „ako je – inače“ grananja za provjeru stanja.	
Varijable	Blok „varijable“ omogućava kreiranje varijabli i njihovo korištenje u programu. Varijable mogu zapisati brojeve i linije teksta u simbolička imena.	

Liste	Blokovi „liste“ omogućuju spremanje i pristupanje listama brojeva i linijama teksta. Ovakav tip podatkovne strukture može se smatrati dinamičnim poljem.	
Upravljanje događajima	„Kada je tipka pritisnuta“ i „kada je lik pritisnut“ su primjeri upravljanja događajima – odgovori na događaje koji su korisnički aktivirani ili aktivirani drugim dijelom programa.	
Niti	Izvršavanje dva sloga u isto vrijeme kreira dvije nezavisne niti koje se izvršavaju paralelno.	

Koordinacija i sinkronizacija	Blokovi „razglasi“ i „kada primim“ mogu koordinirati akcije više likova. Sinkronizacija je moguća korištenjem bloka „razglasi i čekaj“.	
Unos s tipkovnice	Blok „pitaj i čekaj“ traži korisnika unos s tipkovnice, a blok „odgovor“ sprema taj unos.	
Nasumični brojevi	Blok „slučajni broj“ izabire cijeli broj zadanom opsegu.	
Boolean logika	Blokovi „i“, „ili“ i „ne“ su primjeri *Boolean* logike.	
Dinamična interakcija	Blokovi „miš_x“, „miš_y“ i „glasnoća“ mogu se koristiti kao dinamički unos podataka za interakciju u realnom vremenu.	

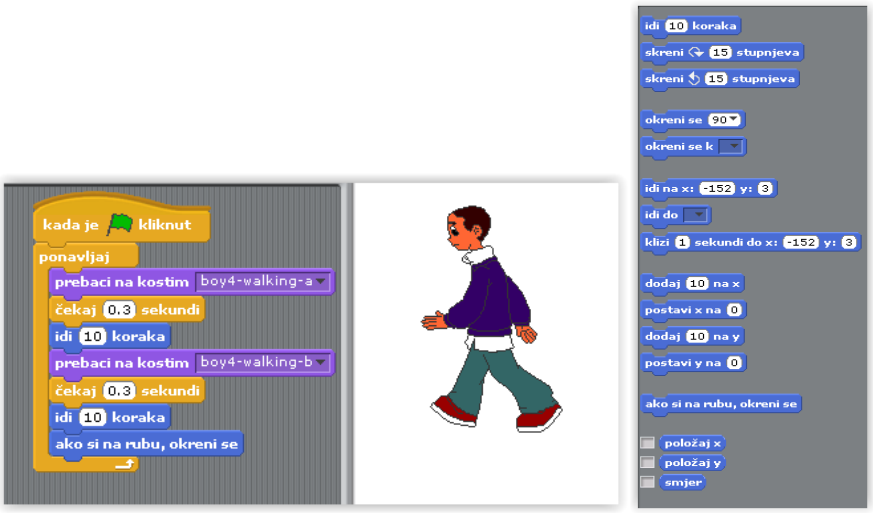
Dizajn korisničkog sučelja	U *Scratchu* se mogu dizajnirati interaktivna korisnička sučelja, npr. ako se koriste likovi na koje se može kliknuti kao na dugmeta.	

5. MULTIMEDIJSKE MOGUĆNOSTI

Kao što je već navedeno, *Scratch* je razvojno okruženje za stvaranje priča, igara, animacija, glazbe i drugih interaktivnih sadržaja na računalu. Neke od osnovnih multimedijskih mogućnosti koje program nudi su: kretanje, umetanje zvuka, promjena pozadine i boje likova, „razgovor“, pomicanje likova pomoću tipki, animiranje likova (hodanje, plesanje...), stvaranje priča i raznih igara.

5.1. Kretanje

Likovi se mogu pokrenuti pritiskom na blok *Kretanje* koji daje razne mogućnosti. Sve mogućnosti ovog bloka prikazane su na slici ispod. Kroz jednostavnu skriptu i mijenjanje kostima lika, „pokrenut će“ se lik te će hodati po pozornici i okretati se kad dođe do ruba.

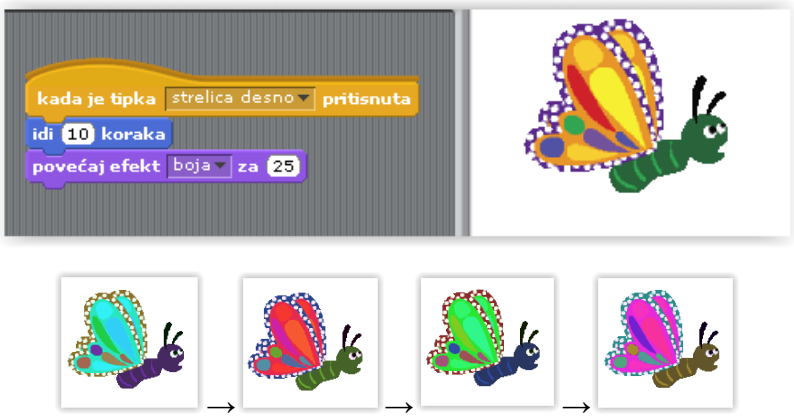


Slika 12. Kretanje lika po pozornici i mogućnosti bloka "kretanje"

5.2. Promjena boje lika i pozadine

PROMJENA BOJE LIKA

Kako bi likovi izgledali zanimljivije, moguće im je promijeniti boju. Za to je potrebno ići pod blok *Izgled* – „povećaj efekt boja za željeni broj“. Boja lika se mijenja ukoliko se njome upravlja (odabere se željena tipka), tako da se pod blokom Upravljanje odabere naredba „kada je određena tipka pritisnuta“. Što lik ima više boja, promjena će biti jače vidljiva.



Slika 13. Promjena boje lika

PROMJENA POZADINE

Kako bi likovi došli do izražaja te cijeli projekt ljepše izgledao, moguće je koristiti već ponuđene, raznolike pozadine raspoređene po kategorijama ili učitati/nacrtati/uslikati svoju te na nju dodavati razne likove.

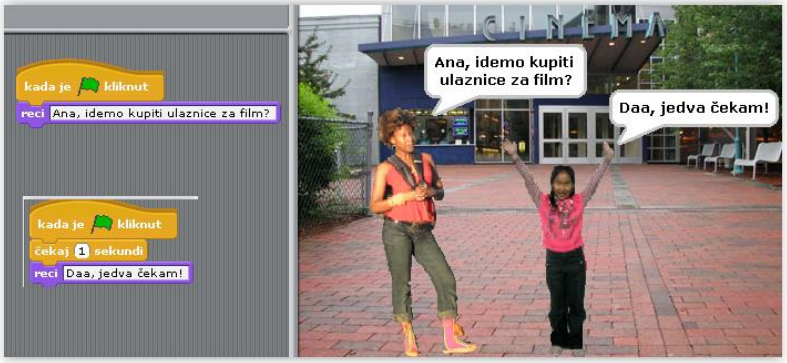
Slika 14. Odabir raznolikih pozadina

5.3. „Razgovor“

Ukoliko se želi da lik koji se nalazi na pozornici nešto govori, koristi se blok *Izgled* i naredbe „govori“ i „reci“. Klikom unutar ova dva bloka može se promijeniti/unesti željeni tekst.

Slika 15. Primjer govorenja

Ako na pozornicu stavimo nekoliko likova, koristeći naredbe „govori“, „reci“, „razmišljaj“ i „misli“ može se stvoriti razgovor između tih likova. U ovom slučaju treba imati dvije skripte, za svaki lik po jednu.



Slika 16. Primjer razgovora

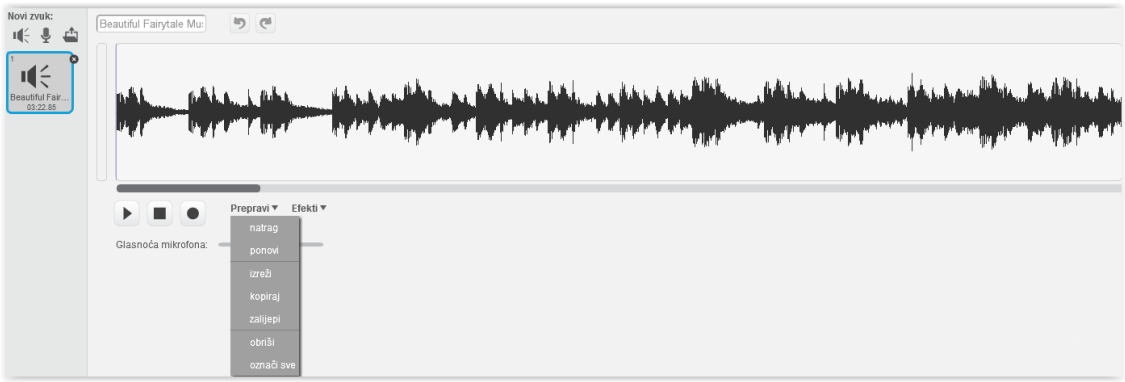
Lik koji se nalazi na pozornici može postaviti pitanje, a odgovara se tako što se tipkovnicom unosi odgovor. Ovime se pridonosi interakciji između programa i korisnika. (npr. *Kako se zoveš?* Unosi se odgovor: *„svoje ime“*).

5.4. Umetanje zvuka

Kako bi *Scratch* projekt bio potpun i zanimljiv, zvuk je njegov neizostavan dio. Svi zvukovi u programu podijeljeni su u dvije vrste: zvukovi i note, a mogu biti reproducirani tijekom izvršavanja programa, bilo kao pozadinska glazba, buka ili kao zvučni efekti prilikom igranja. Uz ponuđene, moguće je snimiti ili uvesti nove audio datoteke. Scratch podržava format MP3 i nekomprimirane formate poput WAV, AIF i AU datoteka. Za pregled zvukova pojedinog lika, potrebno je kliknuti na taj lik i zatim na karticu *Zvukovi*, gdje su ponuđene opcije umetni već postojeći zvuk (zvuk iz programa ili bilo koji zvuk s računala) i snimi zvuk. Svi zvukovi mogu se preslušavati, pauzirati ili po potrebi obrisati.

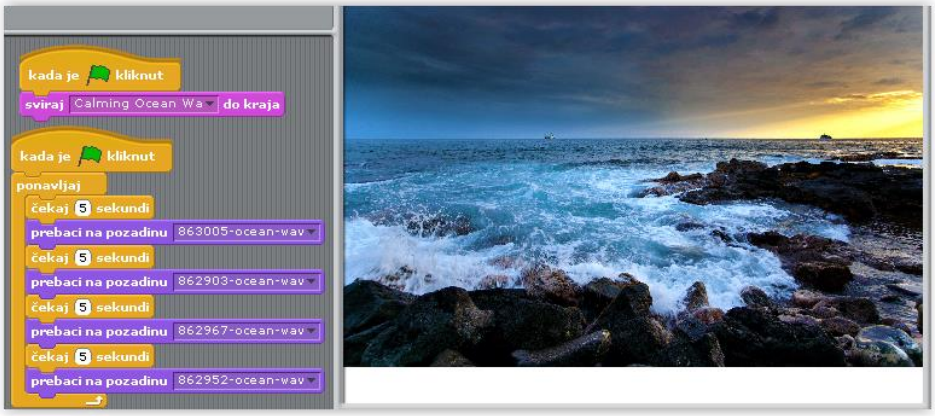
Slika 17. Opcije: umetni ili snimi zvuk

Scratch 2.0 nudi dodatne opcije uređivanja zvuka (kopiranje, izrezivanje...) i dodavanja efekata (postupno pojavljivanje/nestajanje, tišina, glasnije, blaže...).



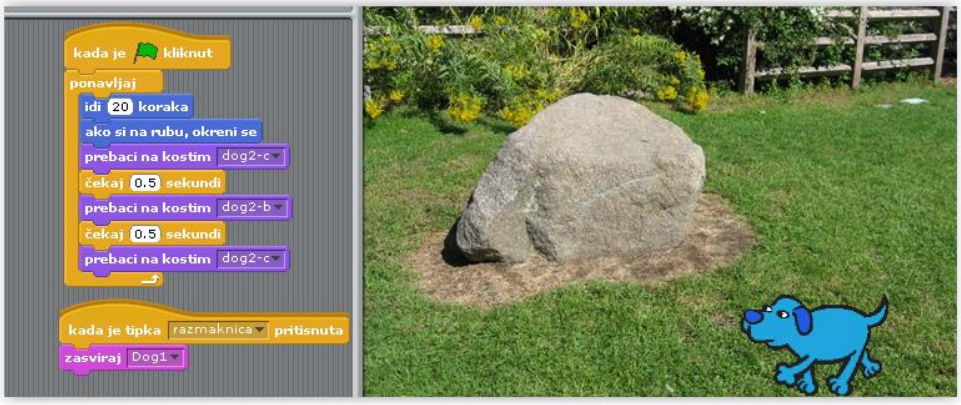
Slika 18. Uređivanje zvučnog zapisa - *Scratch 2.0*

U navedenom primjeru svira umetnuta pozadinska glazba „*Ocean waves*“ te se svakih 5 sekundi mijenjaju pozadine stvarajući opuštajući ugođaj.



Slika 19. Primjer pozadinske glazbe

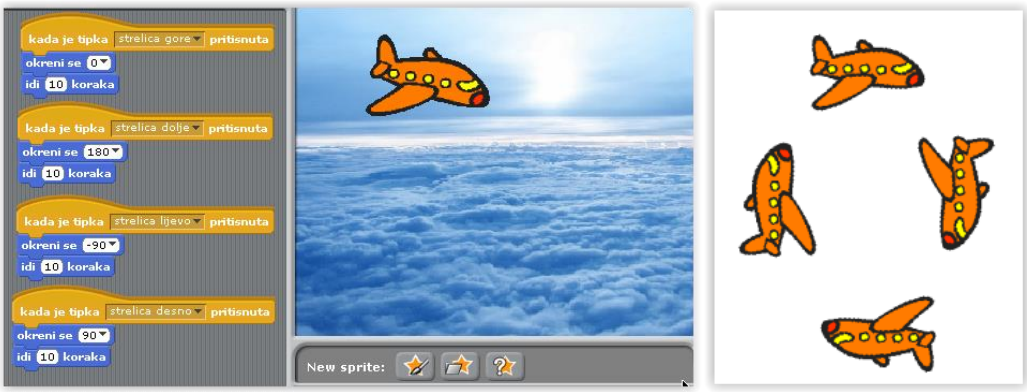
Sljedeći primjer prikazuje lik psa koji šeće po travi i svaki put kad je tipka razmaknica pritisnuta, on zalaje. Zvuk glasanja može se upotrijebiti za bilo koju drugu životinju.



Slika 20. Primjer glasanja psa

5.5. Pomicanje likova pomoću tipki

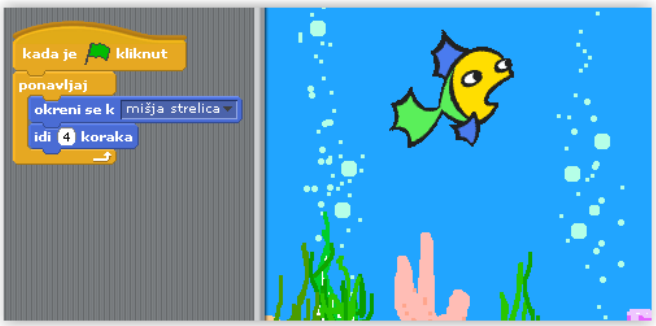
Likovi u *Scratchu* mogu se pomicati tipkama (strelicama, slovima, brojkama, razmaknicom) i time učiniti projekt zanimljivijim. Također, likovi se mogu rotirati/okretati u različitim stupnjevima, biti postavljeni da klize do određenih koordinata i sl. Takve naredbe nalaze se pod blokom *Kretanje.* U navedenom primjeru strelicama gore-dolje-lijevo-desno pomiče se i okreće zrakoplov. Potrebne su četiri skripte, jedna za svaku tipku.



Slika 21. Primjer pomicanja/okretanja likova pomoću tipki

5.6. Slijeđenje pokazivača miša

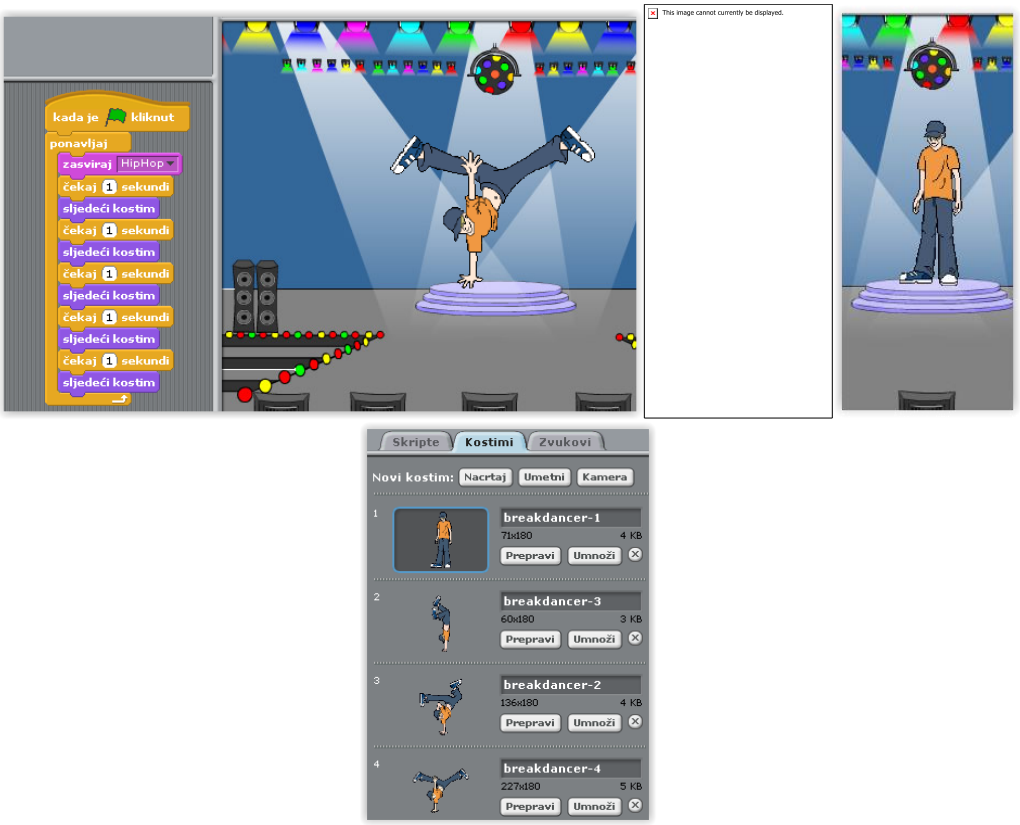
Likovi se mogu programirati tako da slijede pokazivač miša. Pod blokom *Kretanje* treba odabrati „*Okreni se k mišja strelica*“.



Slika 22. Primjer slijeđenja pokazivača miša

5.7. Animiranje likova – plesanje

Animiranje likova postiže se mijenjajući njihove kostime/izgled. Kad se tome dodaju različite opcije kretanja (klizi, hodaj i dr.), zvuk i razni efekti (boje, vrtlog, mozaik...), dobiva se zaokruženu i zanimljiva cjelina.



Slika 23. Primjer animacije (ples) uz glazbu

Slika 24. Primjer animacije slova (imena)

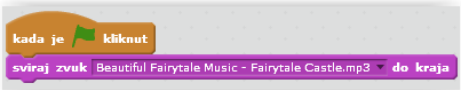
5.8. Stvaranje priče

Stvarajući i gledajući priče u Scratchu, djeca se upoznaju s različitim likovima, pobuđuje im se mašta, razvija smisao za lijepo, prenose se doživljaji i sadržaji te razvijaju psihomotoričke sposobnosti i kreativne potrebe. Učenici mogu razviti multimedijske vještine crtanja likova za njihove priče, preuzimati i uređivati slike koje pronađu na internetu te uvesti ili snimiti zvukove/glazbu za svoje priče. *Scratch* nudi i više nego dovoljno mogućnosti za stvaranje zanimljive interaktivne priče.

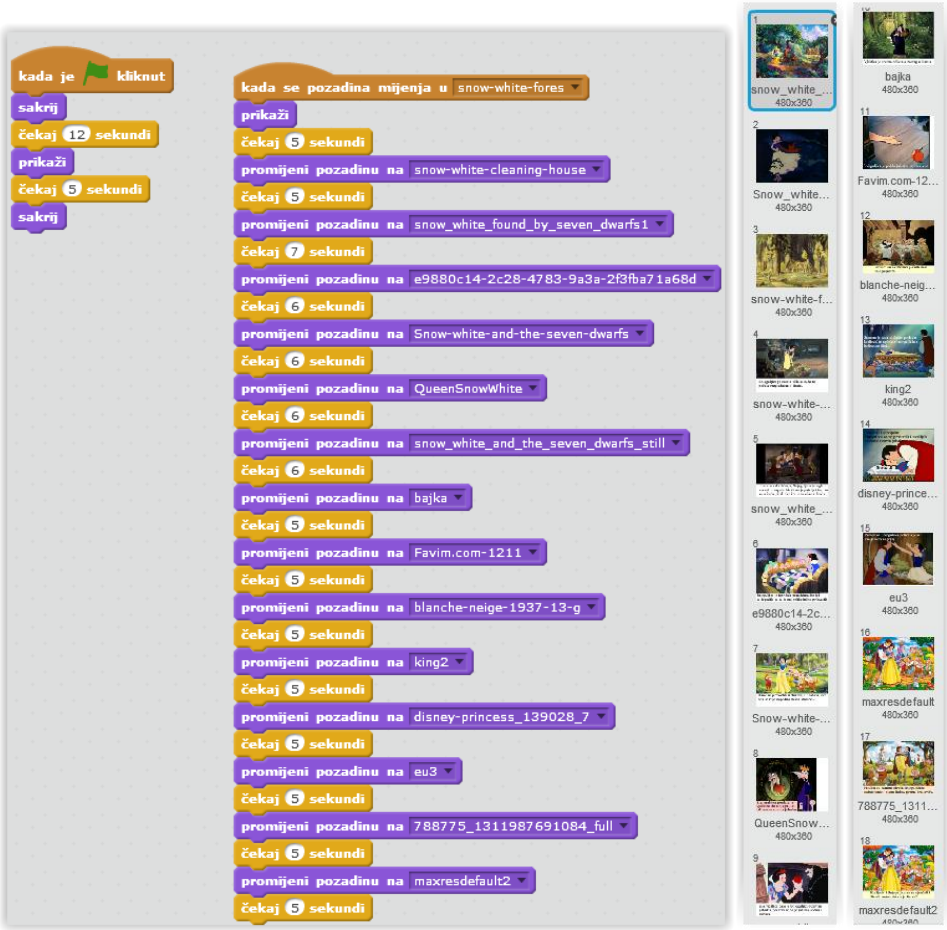
Kao primjer napravljena je bajka *Snjeguljica i sedam patuljaka*. Priča je napravljena izmjenom različitih pozadina s natpisima/tekstom. Na početku se nalazi prva pozadinska slika s natpisom *Snjeguljica i sedam patuljaka*, koji se prikazuje tri sekunde te zatim nestaje promjenom efekta „*duh*“. Za prva tri natpisa korištena su tri lika, a ostali natpisi dodani su direktno na pozadinu pomoću opcije *tekst* u Paint editoru.

Slika 25. Snjeguljica i sedam patuljaka - skripta za prvu pozadinu

Kako bi priča bila efektnija, umetnuta je pozadinska glazba *Fairytale music*.



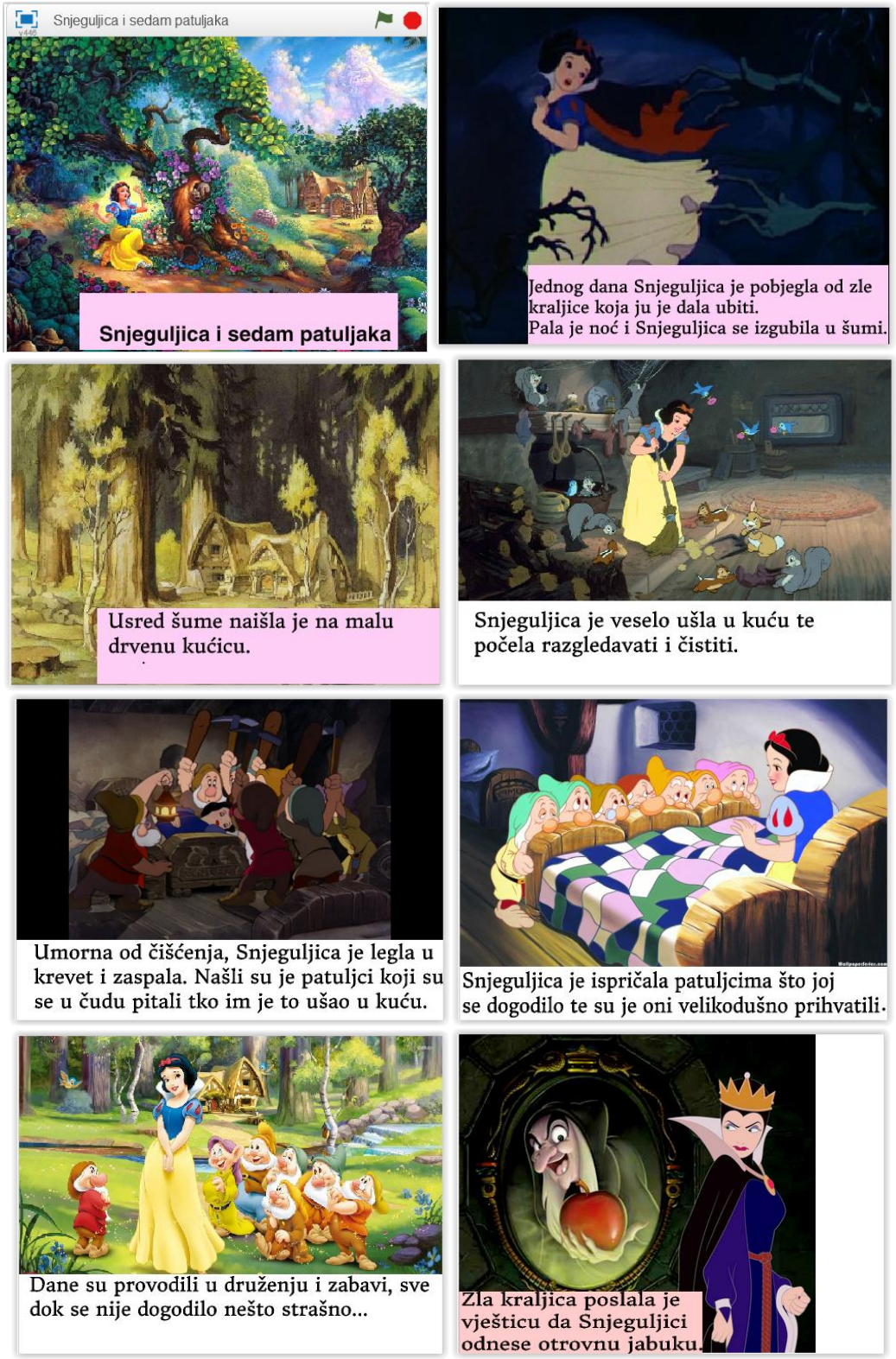
Kako bi se stvorila priča, potrebne su različite pozadine koje se mijenjaju svakih nekoliko sekundi (po želji). Pozadine se mijenjaju pod blokom *Izgled* – *„promijeni pozadinu na“*, a prijelaz između pozadina tempira se pomoću bloka *Upravljanje* – *„čekaj _ sekundi“*. Drugi i treći lik/natpis potrebno je sakriti kako se ne bi prikazali odmah pri pritisku zelene zastavice, nego tek kad se pojavi odgovarajuća pozadina.



Slika 26. Skripta za mijenjanje pozadina i sve upotrebljene pozadine

Jednostavnom izmjenom pozadina i dodavanjem teksta stvorena je priča o *Snjeguljici i sedam patuljaka*. Kako bi se dobilo na interaktivnosti, potrebno je dodati različite likove poput Snjeguljice, zle kraljice, patuljaka, kraljevića i vještice te stvoriti razgovor između njih (pomoću blokova *Izgled* i *Upravljanje*).

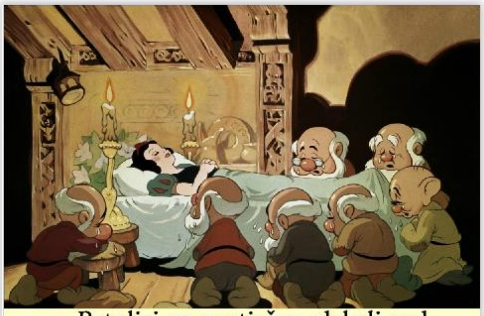
Ovako izgleda napravljena priča – ***Snjeguljica i sedam patuljaka***



Zla vještica dala je Snjeguljici otrovnu jabuku, praveći se da je jabuka sočna i zdrava.

Vještica je sretno odšetala natrag u šumu.

Snjeguljica je pojela jabuku i otrovala se.



Patuljci su neutješno plakali nad Snjeguljicom.

Šumom je tada slučajno prolazio kraljević te ugledao Snjeguljičino beživotno tijelo.

Odlučio ju je poljubiti... Snjeguljica se tad probudila i iskašljala komadić otrovne jabuke.

Presretna Snjeguljica pohrlila je u kraljevićev zagrljaj.

Patuljci su danima slavili Snjeguljičino ozdravljenje i njenu ljubav prema kraljeviću.

Kraljević i Snjeguljica su se vjenčali i živjeli sretno do kraja života!

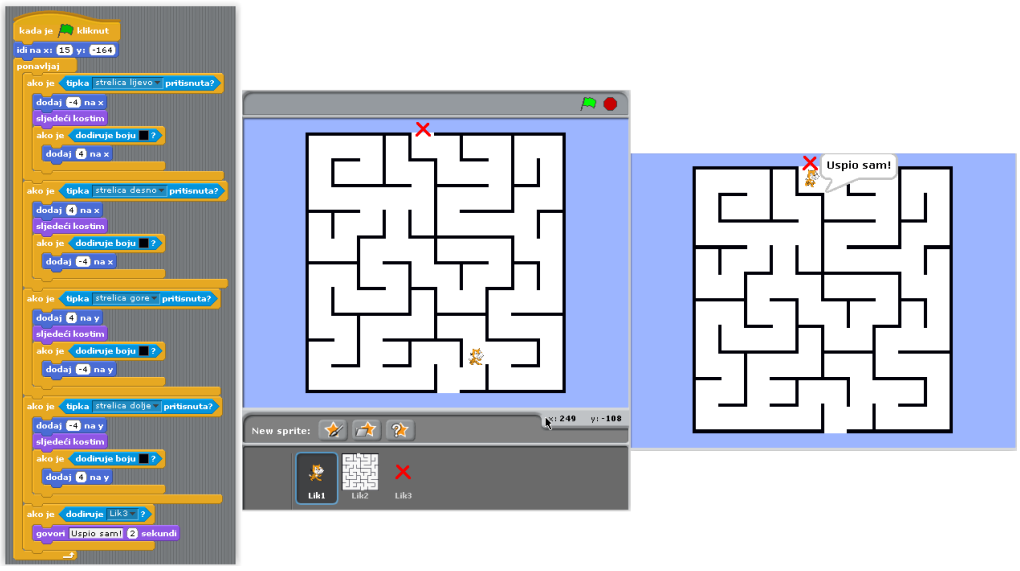
5.9. Igre

Računalne igre često od igrača zahtijevaju brzu reakciju, rješenje nekog problema te mogu pozitivno utjecati na razvoj mnogih sposobnosti (psihomotorne, mentalne i perceptivne) djeteta. Igre mogu imati pozitivan utjecaj na koordinaciju oko-ruka, finu motoriku i razumijevanje prostornih odnosa. Također, mogu pozitivno utjecati na formiranje mišljenja, zaključivanja, inteligenciju i kreativnost.

Pomoću *Scratcha* mogu se stvoriti razne zanimljive igre - akcijske, sportske, edukativne... *Scratch* je pravo mjesto za kreiranje vlastitih junaka, njihovih neprijatelja i novih avantura. Objašnjena su dva primjera jednostavnih igara – labirint i veslo lopta.

LABIRINT

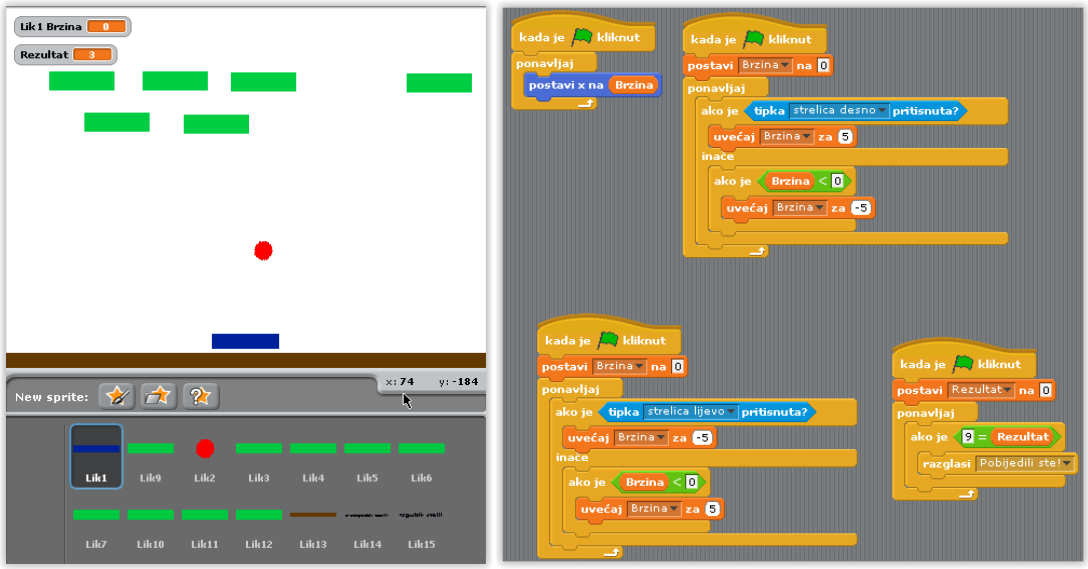
Za ovu igru potrebni su labirint, lik koji ide do cilja te oznaka za cilj. U igri se lik pomiče pomoću tipki sa strelicama (gore-dolje-lijevo-desno), kako bi došao do cilja (crvenog znaka X). Lik mačke ima dva kostima kako bi se dobio privid kretanja. Kako bi se lik svaki put nalazio na početku labirinta, treba mu prvo zadati naredbu „idi na koordinate x 15 i y -164. Pod blokom *Upravljanje* odabiremo naredbe „ponavljaj“ i „ako je“. Zatim u bloku *Očitanja* treba odabrati „tipka ____ pritisnuta“, kako bi se upravljalo kretanjem lika pomoću željenih tipki. Cilj je da lik ne prolazi kroz crte/labirint i tako ga se navodi u pravom smjeru. Kad lik dodiruje cilj, zadana je naredba da izgovori „uspio sam!“.



Slika 27. Igra labirint

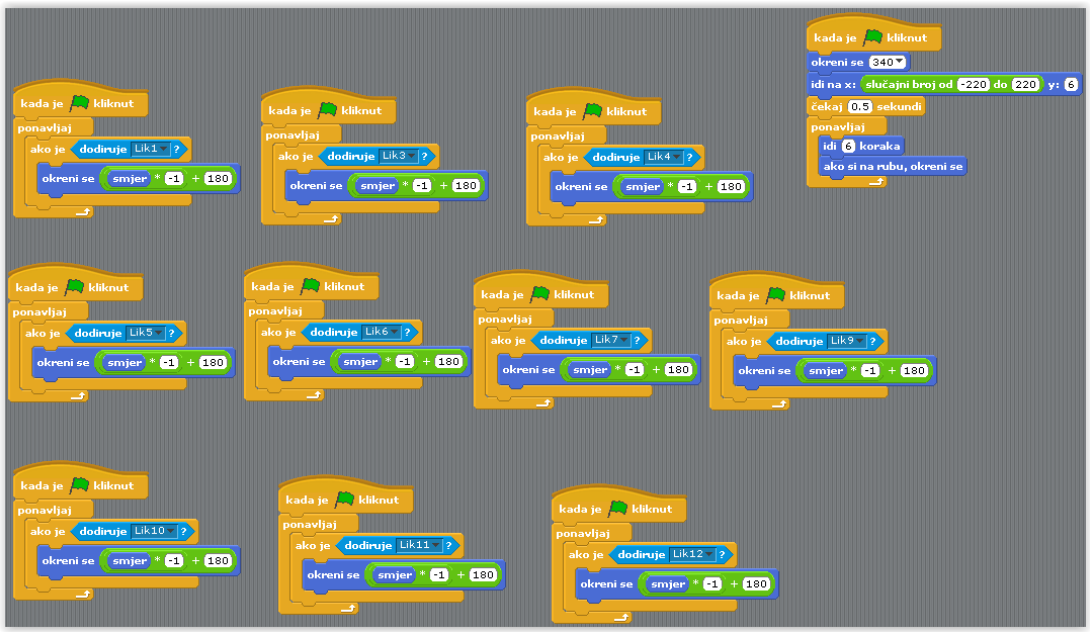
VESLO - LOPTA

Za ovu igru potrebna su tri glavna elementa: lopta, veslo i cigle. Cilj igre je da odguravajući lopticu o veslo njome pogodimo cigle koje nestaju u trenutku dodira s lopticom. Pomoću varijabli „brzina“ i „rezultat“, prati se brzina i rezultat igre (svaka pogođena cigla nosi 1 bod, a ukupno je 9 bodova jer ima 9 cigli). Svaki lik ima po nekoliko skripti. Prvo treba stvoriti/nacrtati lik „veslo“ u Paint editoru, a zatim pomoću petlje/ponavljanja i naredbi *ako je* i *ako je – inače* isprogramirati da se veslo pomiče željenom brzinom pomoću lijeve i desne strelice. Na početku igre brzina i rezultat trebaju biti postavljeni na 0.



Slika 28. LIJEVO - Igra "*Veslo - lopta*" (svi likovi), DESNO - skripte za veslo

Zatim treba složiti da se loptica na početku igre uvijek stvori na drugom mjestu (slučajne koordinate) te ako je na rubu da se odbije od njega. Nakon što su stvorene cigle, treba napisati 10 skripti (1 za veslo i 9 za zelene cigle) kako bi se promijenio smjer loptice kad dotakne ciglu. To se radi pomoću naredbi „ponavljaj“ i „ako je“ te bloka *Kretanje* („okreni se“) i *Operacije* (množenje i zbrajanje).



Slika 29. Skripte za lopticu

Pošto je cilj igre pogoditi/srušiti lopticom sve cigle, treba biti napisana skripta za ciglu u kojoj kad loptica dotakne ciglu, ona nestane. To se radi pomoću naredbi „ponavljaj“ i „ako je“. Također, kad cigla nestane, rezultat se povećava za 1.

Slika 30. Skripta za svaku ciglu

Sljedeće što je potrebno je granična linija ispod vesla koja, ukoliko ju loptica dodirne, razglašava poruku: „Izgubili ste!“. U ovom slučaju granična linija je smeđe boje.

Slika 31. Skripta za graničnu liniju

Zatim trebamo natpise „Izgubili ste!“ i „Pobijedili ste!“ koji se prikazuju kad je primljena poruka da je loptica dodirnula graničnu liniju ili kad su pogođene sve cigle. Tad je kraj igre te se ona zaustavlja.

Slika 32. Skripta za natpise "Pobijedili ste!" i "Izgubili ste!"

Ovisno o završetku igre, javljaju se natpisi (likovi 14. i 15.) *„Pobijedili ste!“* ili *„Izgubili ste!“*

Slika 33. Povratna informacija o završetku igre

6. VARIJABLE, LISTE I NIZOVI (POLJA)

VARIJABLE

Mjesto u memoriji rezervirano za pohranu podatka naziva se varijabla. Programi upravljaju podacima spremljenim u memoriji. Varijabla nije ime samog podatka nego mjesta u memoriji koje čuva podatke. O varijabli treba razmišljati kao o kutiji u koju se spremaju podaci za kasnije korištenje. Varijabla se odnosi izravno na kutiju i neizravno na podatak. Budući da se podaci u kutiji mogu mijenjati za vrijeme izvršavanja programa, varijabla će ukazivati na različite vrijednosti, ali uvijek na istu kutiju.

Kao primjer objašnjen je „*Matematički kviz*“ u kojem imamo lik koji se nalazi na nekoj pozadini. Prikazane su dvije varijable, prvi broj i drugi broj. Lik pita igrača koliki je zbroj ta dva broja. Ako igrač odgovori točno, ocjena mu se poveća za 1 te igrač na kraju dobiva svoju ocjenu. Prvo treba dodati lik na pozornicu, odabrati lik po izboru i postaviti ga na željenu poziciju blokom "idi na". Lik treba pitati koliki je zbroj dva broja. Brojevi će biti slučajno generirani, pa se koristi blok "slučajan broj". Brojeve će računalo spremiti u varijable pa treba stvoriti nove varijable "prvi broj" i "drugi broj". Zatim postaviti njihove vrijednosti na slučajne brojeve u željenom rasponu (npr. 1 do 100). Lik će pitati koliki je zbroj pomoću bloka "pitaj i čekaj". Ako igrač odgovori točno (odgovor je jednak zbroju dva zamišljena broja), lik će reći "Točno!". Ako odgovor nije točan, lik će reći "Netočno!". Zatim se uvodi blok zbrajanja "+". Kviz je moguće nadograditi tako da bude mali test iz matematike. Lik će radnje zamišljanja brojeva, ispitivanja igrača i evaluacije odgovora ponoviti 5 puta i na kraju dati ocjenu igraču. Stvori se novu varijablu „ocjena“ koja je inicijalno 0, a povećava se sa svakim točnim odgovorom igrača. Na kraju igre lik govori igraču koju je ocjenu dobio.

Slika 34. Primjer varijabli - Matematički kviz

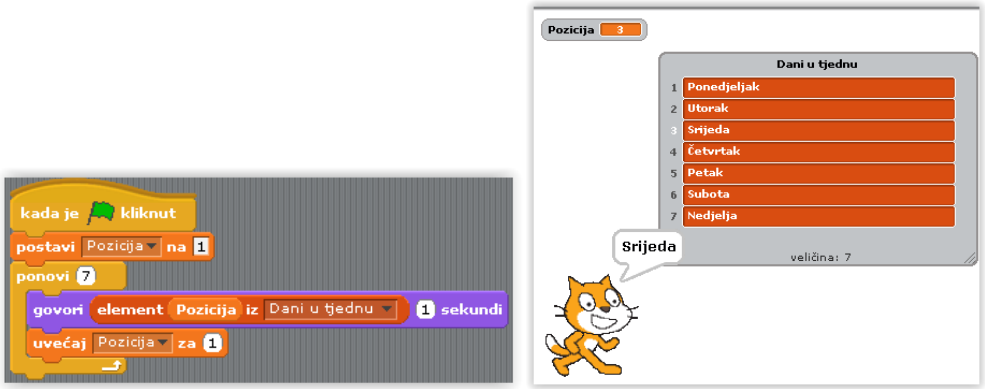
LISTE

U *Scratchu* se mogu stvarati i uređivati liste. U osnovi, lista je niz povezanih elemenata koji se u memoriji ne moraju nalaziti u nizu; suprotno njima, npr. polje je niz elemenata koji se u memoriji nalaze u nizu. Liste mogu sadržavati brojeve, nizove slova i drugih znakova. Lista je kao spremnik u koji možemo pohraniti i pristupiti višestrukim vrijednostima. Taj spremnik sadrži brojne ladice, a u svakoj ladici nalazi se jedan predmet. Za stvaranje liste potrebno je otvoriti kategoriju blokova *Varijable* i kliknuti „Nova lista“. Kada se stvori popis, na pozornici će se pojaviti monitor popisa. Monitor s popisom pokazuje sve elemente danoga popisa. Elementi popisa mogu se unijeti izravno u monitor popisa (klikom na znak +). U listama se može dodavati, izbacivati ili zamjenjivati elemente.

Na početku će popis biti prazan, s duljinom 0. Za dodavanje elementa treba kliknuti na gumb + na dnu monitora popisa. Duljina će se uvećati za 1. Također, moguće je dodavati elemente koristeći blokove popisa (primjerice 

).

Veličinu monitora popisa moguće je promijeniti u donjem desnom kutu. U navedenom primjeru program nam redom „govori“ dane u tjednu, prikazuje njihovu poziciju (npr. srijeda, pozicija 3) te ukupnu veličinu liste (7 elemenata).

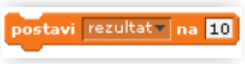


Slika 35. Primjer liste - Dani u tjednu

NIZOVI (POLJA)

U programima se ponekad rabe skupovi podataka istog tipa koji predstavljaju cjelinu. Skup podataka istog tipa može se pohraniti pod zajedničkim imenom, a redni broj podatka u tom skupu označiti brojem (indeksom). Takav način pohrane omogućavaju polja (nizovi) podataka. Polje je konačni niz podataka istog tipa koji predstavljaju cjelinu, a sastoji se od članova polja.

Nizovi su sačinjeni od slova, riječi ili drugih znakova (npr.: jabuka, Listopad 2015, Čestitam!). Nizove je moguće spremati u varijable ili kao popise (poput 

ili 

).

Združuju se korištenjem opcije 

, a usporediti se mogu primjenom sljedećih blokova: 
, 
ili 
. Nizovi se vrednuju kao nula u blokovima matematičkih operacija (poput: 
) i u blokovima koji predviđaju broj (poput: 

ili 

). (ODRAZI, 2011, str. 16).

7. PRIMJENA *SCRATCH*-a U OSNOVNOJ ŠKOLI

Scratch pridonosi razvoju mnogih vještina: učenja, kritičkog razmišljanja, rješavanja problema, komunikacije, suradnje, kreativnosti i inovacija. Te vještine uključuju jasno komuniciranje, analiziranje, sustavno i učinkovito surađivanje, kreativno razmišljanje te kontinuirano učenje. Ovakav pristup učenju i poučavanju inspiriran je konstruktivističkom teorijom učenja i obrazovanja (učenje temeljeno na iskustvu) (Screawn, 2014.)

Scratch učiteljima nudi priliku korištenja računala u svakodnevnim školskim aktivnostima, uzimajući u obzir razvojne osobine učenika (razreda) i nastavne sadržaje koji to svojom metodičkom posebnošću dopuštaju. Učenje pomoću računala i *Scratcha* ne može u potpunosti zamijeniti komunikaciju učenika i učitelja, ali mnoge aktivnosti može učiniti jednostavnijima i zanimljivijima. Ponavljanje uz pomoć edukativnih igara čini nastavnu aktivnost privlačnijom, stvaranje i prezentiranje vlastitih uradaka dobiva na novim mogućnostima izražavanja, a samostalno istraživanje potiče radoznalost i upornost kod učenika.

7.1. *HRVATSKI JEZIK*

Hrvatski jezik najopsežniji je predmet osnovnoškolskoga obrazovanja te je usko povezan sa svim ostalim predmetima. Upravo zbog toga postoje brojne mogućnosti uporabe informatičke tehnologije i programa *Scratch,* kako u obradi novih nastavnih sadržaja, tako i za ponavljanje naučenog (kvizovi, igre i sl.).

Za ponavljanje nastavnog sadržaja *Vrste riječi – imenice, glagoli, pridjevi* može poslužiti neobičan kviz sa životinjama. Na svakoj životinji nalazi se natpis (sretan, rak, letjeti, smeđ...) te životinju treba povući do odgovarajućeg stupca ovisno o njenom natpisu (imenica, glagol, pridjev). Ako je učenik povukao životinju u točan stupac, životinja nestaje, stupac mijenja naziv u *TOČNO* i javlja se pripadajući zvuk (pljesak i sl.), a ako je učenik pogriješio, životinja se vraća na početno mjesto/koordinate i igra se nastavlja. Na ovaj način igrač (učenik) odmah dobiva povratnu informaciju o riješenosti.

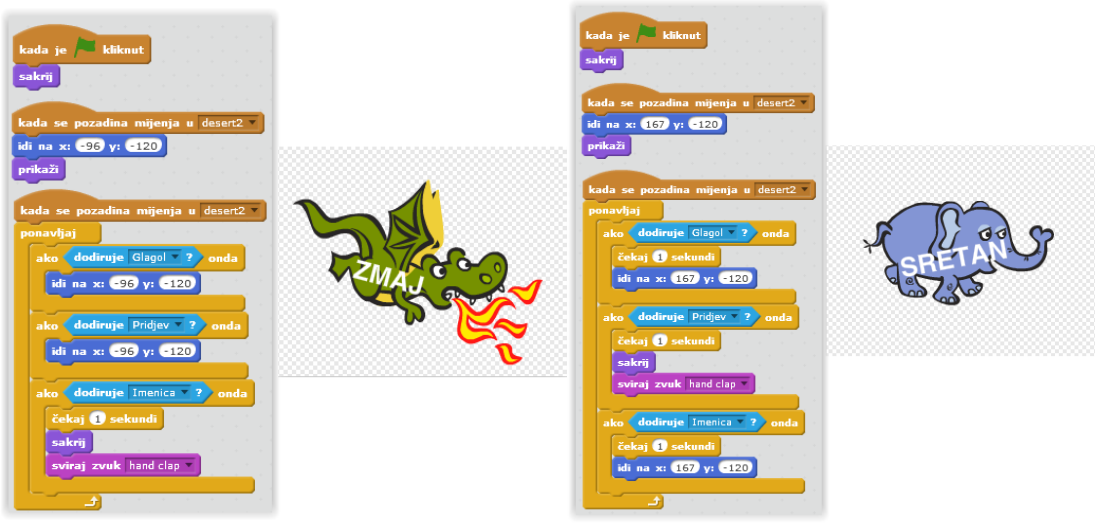
Kviz se sastoji od tri stupca (imenica, glagol, pridjev) i devet životinja s natpisima. Životinju treba povući do odgovarajućeg stupca ovisno o njenom natpisu, odnosno vrsti riječi koju „nosi“. Ako životinja dodiruje odgovarajući stupac, tad ona nestaje, a stupac mijenja kostim/naziv u *TOČNO* na dvije sekunde. Kad životinja dodiruje „krivi“ stupac, vraća se na početnu poziciju/koordinate i to znači da je igrač pogriješio u odabiru stupca. Prva pozadina uvodi nas u kviz, dok se na drugoj (*pozadina2*) nalaze životinje.

Slika 36. Izgled programa *Vrste riječi - imenice, glagoli, pridjevi*

Stupac *Imenica* povezan je sa svim životinjama/likovima koje nose „imenicu“. To znači da kad lik *imenica2, imenica3 i imenica4* dodiruju stupac *imenica*, on prepoznaje da je to točan odgovor te mijenja kostim/natpis u *TOČNO*. Kad likovi *glagol2, glagol3* i *glagol4* dodiruju stupac *glagol*, on mijenja kostim u *TOČNO*. Isto tako kad lik *pridjev2, pridjev3* i *pridjev4* dodiruju stupac *pridjev*, on također mijenja kostim u TOČNO. Naredbe za dodir nalaze se pod blokom *Očitanja*.

Slika 37. Skripte za stupce: *imenica, glagol i pridjev*

Svaki lik/životinja ima svoju skriptu, a životinje se pojavljuju kad se pozadina mijenja u *pozadinu 2*.




kada je kliknut
sakrij
kada se pozadina mijenja u desert2
idi na x: 72 y: -110
prikaži
kada se pozadina mijenja u desert2
ponavljaj
ako dodiruje Glagol ? onda
čekaj 1 sekundi
sviraj zvuk hand clap
sakrij
ako dodiruje Imenica ? onda
idi na x: 72 y: -110
ako dodiruje Pridjev ? onda
idi na x: 72 y: -110


LETJETI


kada je kliknut
sakrij
kada se pozadina mijenja u desert2
idi na x: -61 y: -35
prikaži
kada se pozadina mijenja u desert2
ponavljaj
ako dodiruje Imenica ? onda
idi na x: -61 y: -35
ako dodiruje Glagol ? onda
čekaj 1 sekundi
sviraj zvuk cricket
sakrij
ako dodiruje Pridjev ? onda
idi na x: -61 y: -35


PUZATI


kada je kliknut
sakrij
kada se pozadina mijenja u desert2
idi na x: 60 y: 60
prikaži
kada se pozadina mijenja u desert2
ponavljaj
ako dodiruje Imenica ? onda
čekaj 1 sekundi
sviraj zvuk pop
sakrij
ako dodiruje Glagol ? onda
idi na x: 60 y: 60
ako dodiruje Pridjev ? onda
idi na x: 60 y: 60


ŠIŠMIŠ

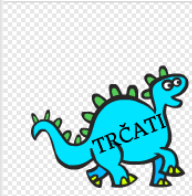

kada je kliknut
sakrij
kada se pozadina mijenja u desert2
idi na x: 181 y: -42
prikaži
kada se pozadina mijenja u desert2
ponavljaj
ako dodiruje Imenica ? onda
idi na x: 181 y: -42
ako dodiruje Glagol ? onda
idi na x: 181 y: -42
ako dodiruje Pridjev ? onda
čekaj 1 sekundi
sviraj zvuk meow2
sakrij

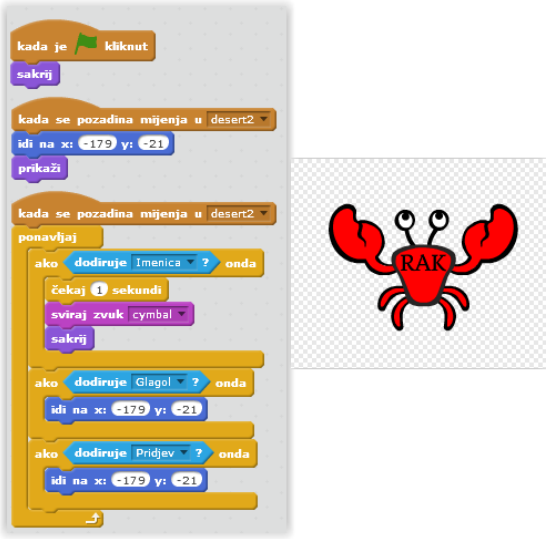

VELIK


kada je kliknut
sakrij
kada se pozadina mijenja u desert2
idi na x: -187 y: -114
prikaži
kada se pozadina mijenja u desert2
ponavljaj
ako dodiruje Imenica ? onda
idi na x: -187 y: -114
ako dodiruje Glagol ? onda
idi na x: -187 y: -114
ako dodiruje Pridjev ? onda
čekaj 1 sekundi
sviraj zvuk pop
sakrij


SMEĐ


kada je kliknut
sakrij
kada se pozadina mijenja u desert2
idi na x: 167 y: 55
prikaži
kada se pozadina mijenja u desert2
ponavljaj
ako dodiruje Imenica ? onda
idi na x: 167 y: 55
ako dodiruje Glagol ? onda
čekaj 1 sekundi
sviraj zvuk chomp
sakrij
ako dodiruje Pridjev ? onda
idi na x: 167 y: 55


TRČATI

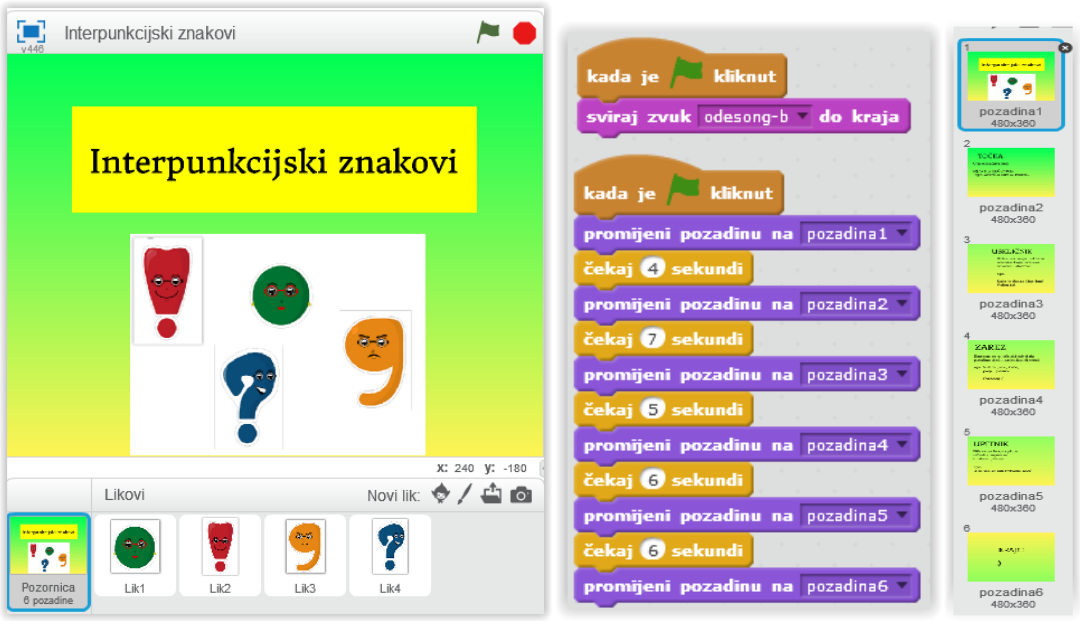


Slika 38. Skripta i slika za svaku životinju/lik

Ako je igrač uspješan u razvrstavanju natpisa na životinjama, onda one nestaju s pozornice i prilikom svakog nestajanja pojedini stupac mijenja naziv u *TOČNO* na 2 sekunde.

Slika 39. Primjer nestajanja životinje u donjem lijevom kutu s natpisom "*smeđ*" koji pripada stupcu *pridjev*

Sljedeći primjer prikazuje *Interpunkcijske znakove* i može poslužiti u obradi ili ponavljaju ovog nastavnog sadržaja. Program sadrži šest pozadina i četiri lika koji predstavljaju osnovne interpunkcijske znakove (točka, zarez, upitnik i uskličnik) te svaki lik ima svoju skriptu. Izmjenom pozadina i likova dobiva se *slideshow* koji ukratko objašnjava definiciju pojedinog znaka i pokazuje primjer. Kako bi program bio zanimljiviji dodana je glazba koja svira od početka do kraja izvođenja programa te efekt „klizanja“ od jedne točke do druge kod pojedinih likova.



Slika 40. Program, skripta i pozadine za *Interpunkcijske znakove*

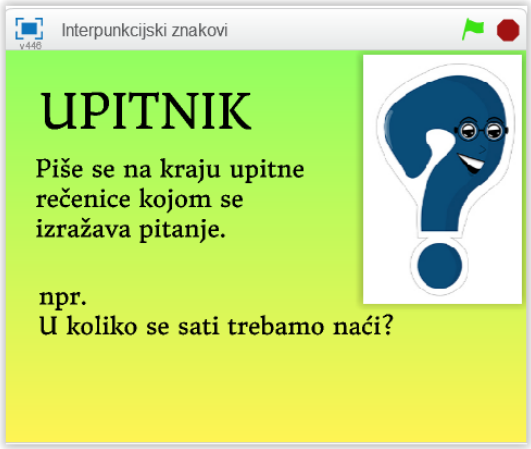
Izgled programa *Interpunkcijski znakovi*


Interpunkcijski znakovi
USKLIČNIK
Dolazi na kraju usklične rečenice koja izražava emocije i stavove.
npr.
Kako je danas lijep dan!
Volim te!

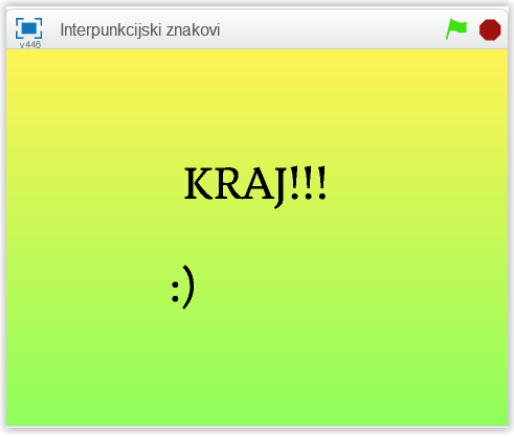

kada je kliknut
sakrij
kada se pozadina mijenja u pozadina3
idi na x: -165 y: 67
prikaži
klizi 3 sekundi do x: -166 y: -61
čekaj 2 sekundi
sakrij


Interpunkcijski znakovi
ZAREZ
Zarezom se u rečenici odvajaju pojedine riječi i rečenični dijelovi.
npr. Vidim ljude, kuće, polja, planine.
(NIZANJE)


kada je kliknut
sakrij
kada se pozadina mijenja u pozadina4
prikaži
čekaj 6 sekundi
sakrij


Interpunkcijski znakovi
UPITNIK
Piše se na kraju upitne rečenice kojom se izražava pitanje.
npr.
U koliko se sati trebamo naći?


kada je kliknut
sakrij
kada se pozadina mijenja u pozadina4
prikaži
čekaj 6 sekundi
sakrij

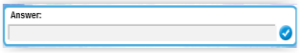

Interpunkcijski znakovi
KRAJ!!!
:)

Sljedeći primjer može poslužiti kao kratka provjera znanja iz lektire, a radi se o književnom djelu *Šegrt Hlapić* koje se obrađuje u 3 razredu. Kviz se pokreće likom na gumb „*start*“ i sastoji se od pet pitanja. Svako pitanje ima četiri ponuđena odgovora (*a, b, c, d*), a samo je jedan točan odgovor. Igrač pomoću tipkovnice unosi svoj odgovor (slovo) u kućicu „*Answer*“. Za svaki točan odgovor, bodovi se povećavaju za 1, a za svaki netočan se smanjuju za 1.

Slika 41. Izgled programa i pozadine za kviz *Čudnovate zgode šegrta Hlapića*

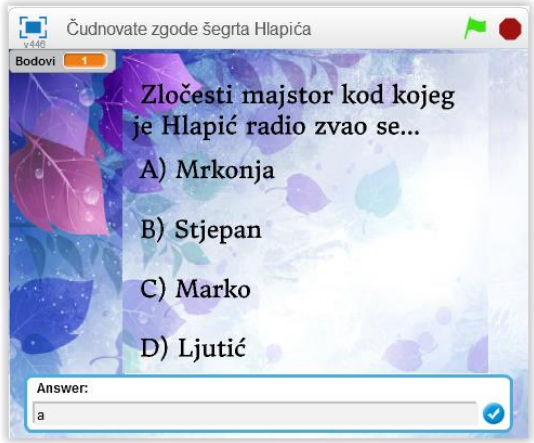
Na početku kviza se pojavljuje pozadina *Čudnovate zgode šegrta Hlapića*, gumb *start* i varijabla *Bodovi* koja je postavljena na 0. Kviz se sastoji od sedam pozadina i dva lika (gumb start i strelica). Promjenom pozadine mijenjaju se pitanja u kvizu.

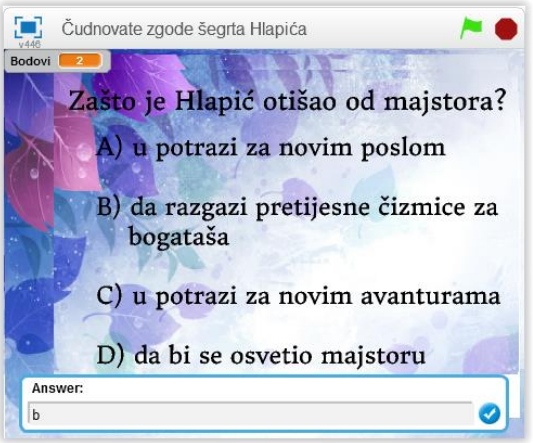
Slika 42. Gumb *start* i njegova skripta te skripte za naslovnu pozadinu i varijablu "*bodovi*"

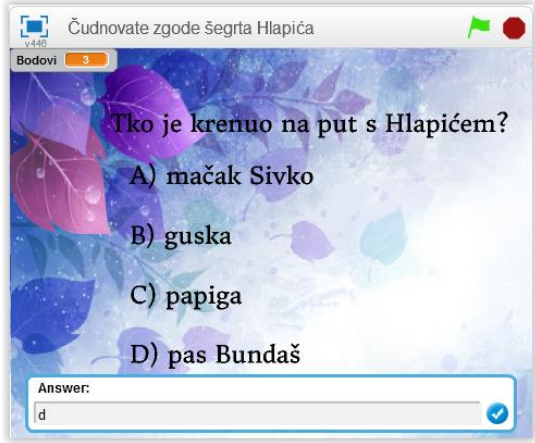
Pritiskom na gumb start pojavljuje se prvo pitanje (pozadina se mijenja u pozadinu1), na dnu pozornice pojavljuje se kućica „*Answer*“ 

u koju igrač upisuje svoj odgovor te ga potvrđuje pritiskom tipke enter na tipkovnici. Ovisno o tome pod kojim se slovom nalazi točan odgovor (*A/a, B/b, C/c, D/d*), mijenja se varijabla „*bodovi*“ (+ 1 ili -1). Ako igrač odabere netočan odgovor, igra se nastavlja dalje i nije ponuđena mogućnost ispravka odgovora. Ovime se igrača upućuje na pozorno čitanje pitanja i promišljanje o odgovoru. Svako pitanje ima svoju skriptu.

Izgled kviza *Čudnovate zgode šegrta Hlapića*





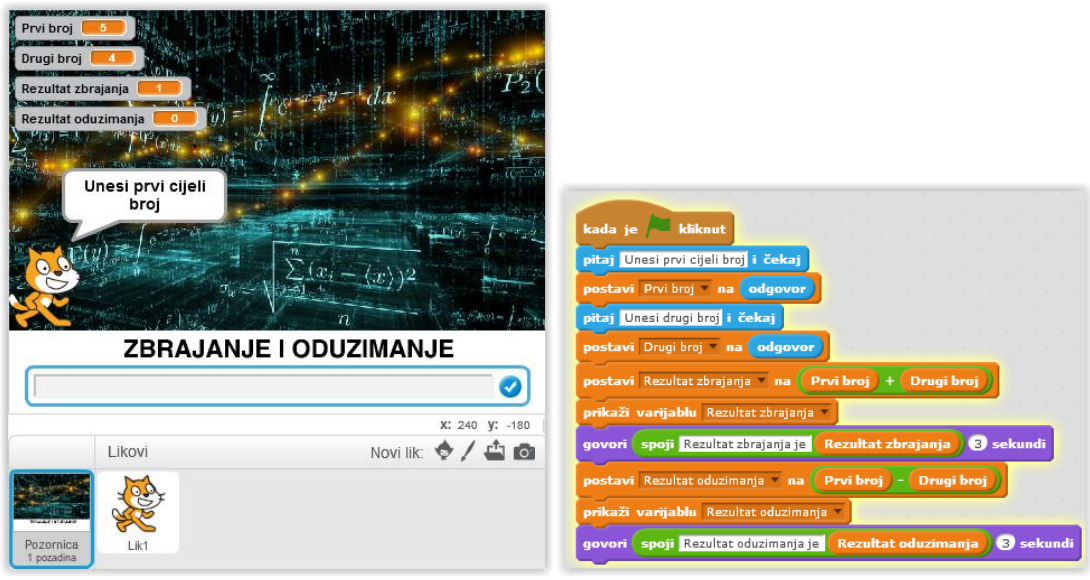


7.2. *MATEMATIKA*

Matematika nas uči da zadani problem gledamo iz različitih kutova, koristeći prethodno znanje, kao i povezivanje na prvi pogled potpuno različitih teorija i pravljenje analogija među objektima.

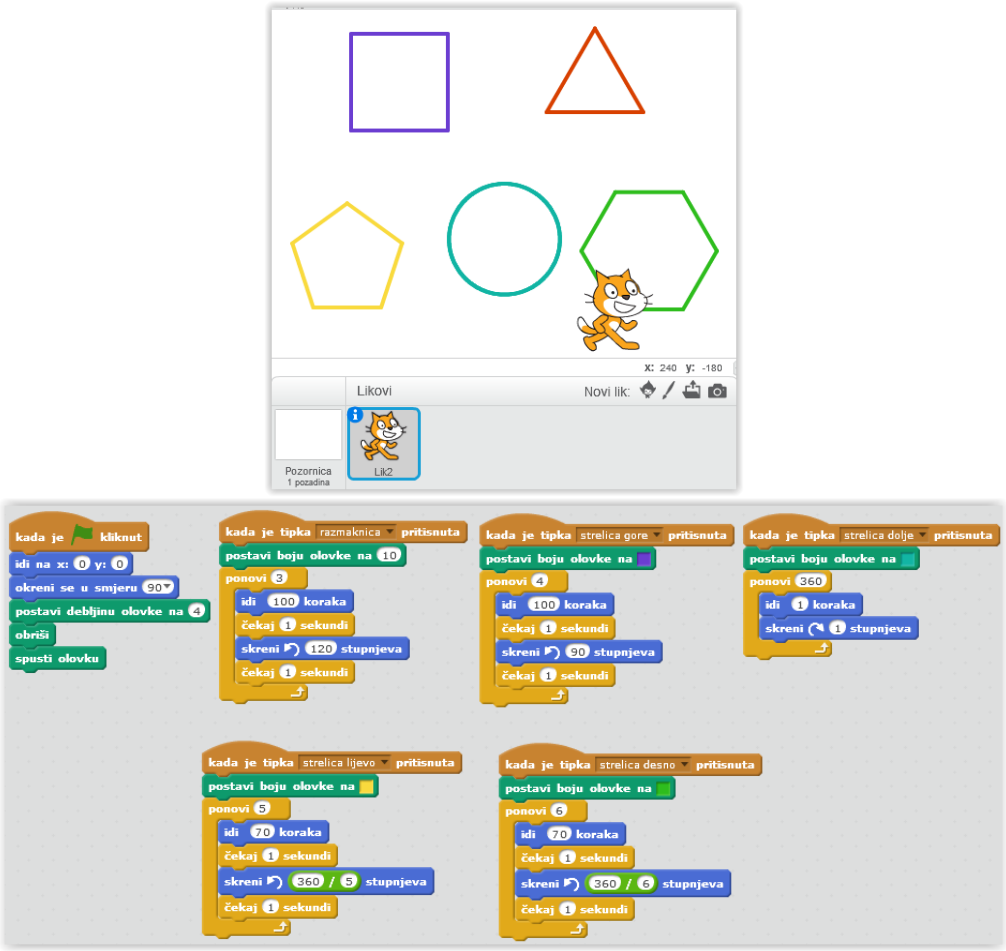
Scratch se može koristiti u podučavanju i učenju temeljnih matematičkih koncepata, aritmetike, geometrije, algebre i dr. Može se isprogramirati da program izvodi četiri računske operacije (zbrajanje, oduzimanje, množenje i dijeljenje), kvadrira, razlikuje parne i neparne brojeve, izračunava opseg, prikazuje različite geometrijske likove ili tijela te razne matematičke igre/kvizove (*slika 34).*

U navedenom primjeru, program izračunava zbroj i razliku dvaju unesenih cijelih brojeva. Nakon što se stvori/odabere željenu pozadinu i lik, potrebno je napisati skriptu kako bi program izvršavao ono što mu je zadano. Program se pokreće klikom na zelenu zastavicu stoga ona stoji na početku skripte. Pomoću bloka *Očitanja* odabere se „pitaj Unesi prvi cijeli broji čekaj“. Treba stvoriti mogućnost unošenja dva cijela broja – ti brojevi su varijable, kao i oba rezultata. Zatim pomoću varijable „rezultat zbrajanja“ i bloka *Operacije* postaviti rezultat zbrajanja na „prvi“ + „drugi“ broj (isti princip vrijedi za oduzimanje). Pod blokom *Izgled* odabrati naredbu govori _ sekundi: „rezultat zbrajanja/oduzimanja je (varijabla rez. zbrajanja i rez. oduzimanja)“. Za izvođenje programa bez ponovnog pritiska zelene zastavice, potrebno je upotrijebiti petlju/ponavljanje.



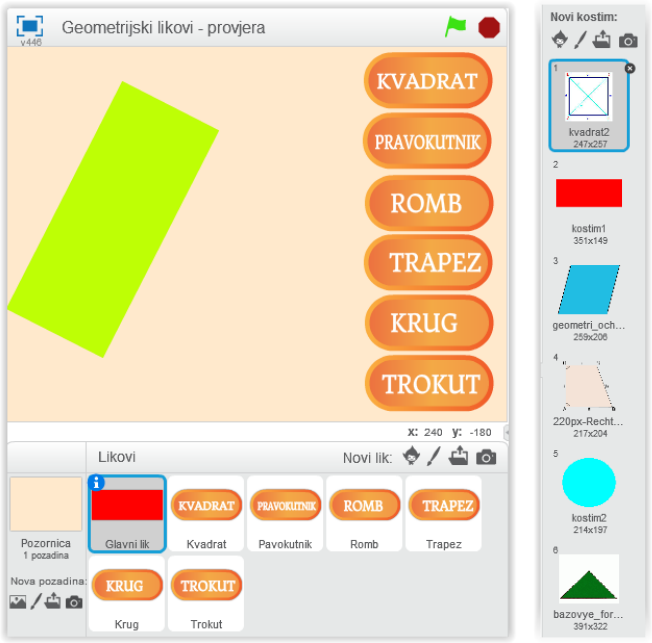
Slika 43. Izgled programa i skripte za izračunavanje zbroja i razlike dvaju unesenih brojeva

U sljedećem primjeru pritiskom određenih tipki (razmaknica, strelice gore, dolje, lijevo, desno) program izvršava crtanje različitih geometrijskih likova (trokut, kvadrat, kružnica, peterokut i šesterokut). Za početak treba postaviti lik na željene koordinate (u ovom slučaju x=0, y=0) te ga okrenuti za 90 stupnjeva. Želi se da lik prilikom kretanja „ostavlja trag“, stoga mu se mora spustiti olovka i odrediti njena debljina. Svakim pritiskom tipke razmaknica, program će iscrtavati trokut. Pritiskom strelice prema gore, program će plavom bojom iscrtavati kvadrat. To je postignuto naredbom *ponavljaj 4 puta* (jer kvadrat ima 4 stranice), *idi 100 koraka* (proizvoljna duljina), *čekaj i okreni se za 90 stupnjeva* (kutovi kvadrata su pravi kutovi te iznose 90 stupnjeva). Kad je tipka strelica dolje pritisnuta, program iscrtava kružnicu. Pritiskom tipke strelica lijevo, iscrtat će se peterokut (treba skrenuti za 360/5, jer peterokut ima 5 kutova). U slučaju da je tipka strelica desno pritisnuta, program će zelenom olovkom iscrtati šesterokut (treba skrenuti za 360/6, jer šesterokut ima 6 kutova). Ovim primjerom učenicima možemo prikazati crtanje geometrijskih likova na zanimljiv način.



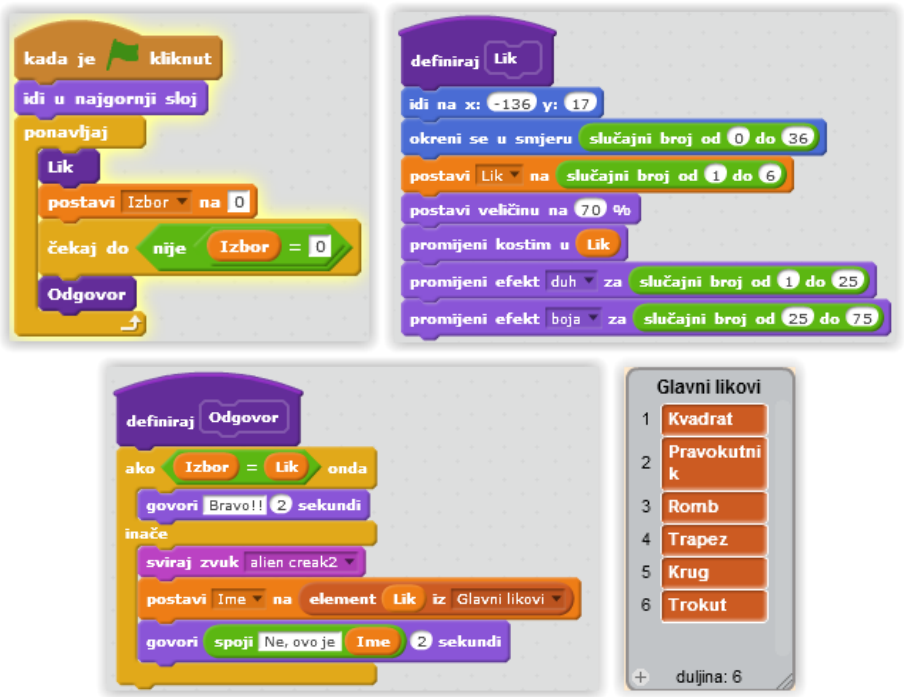
Slika 44. Izgled programa i skripte za iscrtavanje geometrijskih likova

Sljedeći primjer može poslužiti kao provjera znanja (prepoznavanja) različitih geometrijskih likova. Ovaj kviz nasumično prikazuje jedan od 6 geometrijskih likova (kvadrat, pravokutnik, romb, trapez, krug ili trokut) te učenik treba klasificirati geometrijski lik pritiskom na odgovarajući gumb koji prikazuje naziv lika.



Slika 45. Izgled programa za prepoznavanje geometrijskih likova i kostimi *glavnog lika*

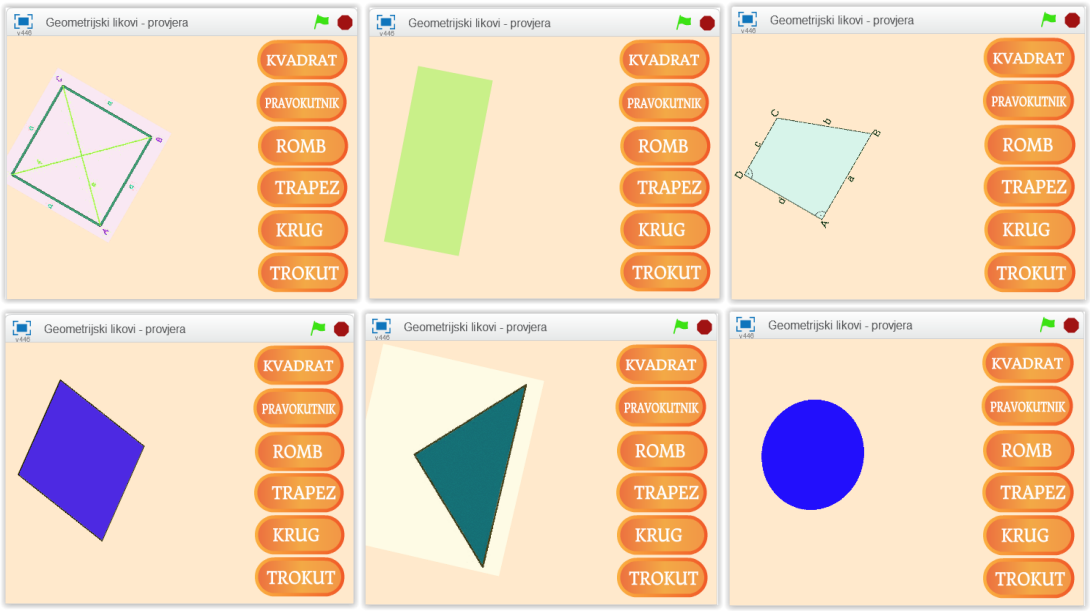
Kviz se sastoji od 7 likova *(sprites)* – prvi lik je *glavni lik* koji sadrži glavnu skriptu za izvođenje programa, a ostalih 6 likova su gumbići s nazivima. *Glavni lik* ima 6 kostima (kvadrat, pravokutnik, romb, trapez, krug i trokut) koji odgovaraju gumbićima (nazivima). U glavnoj skripti prvo treba postaviti *glavni lik* na gornji sloj („idi u gornji sloj“) kako bi stalno bio vidljiv (da ga gumbići ne prekrivaju). U svakom ponavljanju/petlji, skripta (program) prikazuje nasumično odabrani geometrijski lik, njegovu boju i smjer. Zatim se stvorena varijabla „Izbor“ postavlja na 0 te ona označava da igrač još nije odgovorio, odnosno pritisnuo gumbić. Svaki gumbić sadrži varijablu „Izbor“ koja ima drugačiju vrijednost (od 1 do 6). Tad program čeka sve dok se varijabla „Izbor“ ne promijeni u broj koji je različit od 0, a to se događa kad igrač pritisne jedan od šest ponuđenih gumbića. Kad igrač pritisne gumb, skripta poziva blok *Odgovor* i provjerava je li varijabla „Izbor“ jednaka „Liku“. Ako jest, onda se igraču prikazuje povratna informacija „*Bravo!*“, a ukoliko je igrač krivo odgovorio/pritisnuo, reproducira se zvuk „alien creak2“ i poruka „Ne, ovo je (točan naziv lika)“.



Slika 46. Skripte i lista za *Glavni lik*

...

Slika 47. Skripte za gumbiće (1-6, od kvadrata do trokuta)



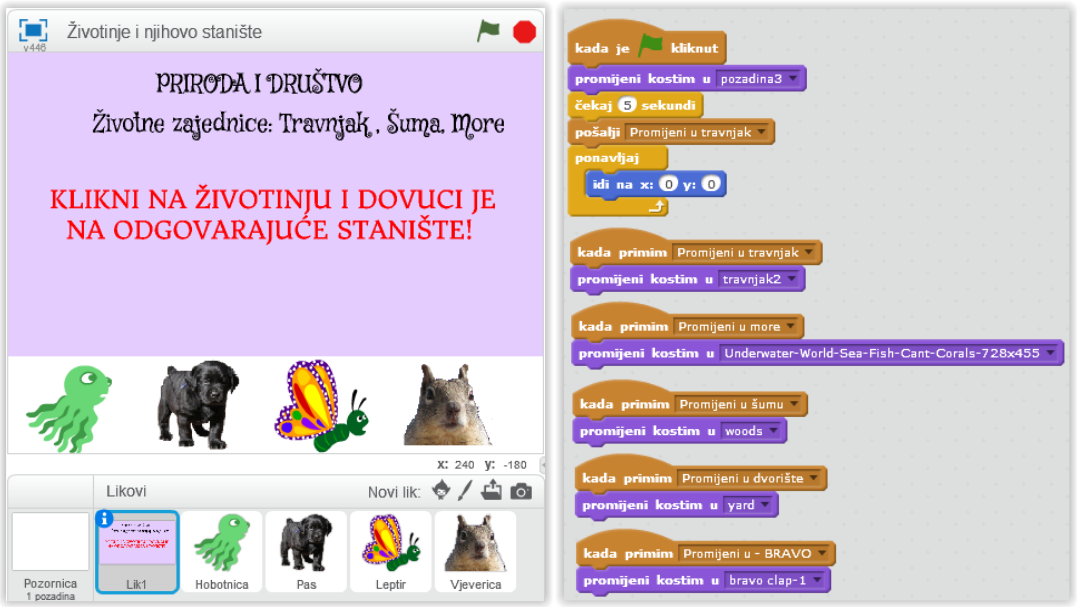
Slika 48. Prikaz svih 6 geometrijskih likova

Slika 49. Povratne poruke ovisno o rješenju

7.3. *PRIRODA I DRUŠTVO*

Mogućnosti primjene *Scratch*-a u nastavi prirode i društva su neograničene, od grafičkih prikaza, audio zapisa, interaktivnih kvizova, igara i animacija te korištenja simulacija stvarnih događaja ili procesa koje omogućavaju potpunije razumijevanje sadržaja.

Kao primjer izrađen je kviz u kojem učenici mogu ponoviti sadržaj vezan uz nastavne jedinice *Životne zajednice – travnjak, šume i more*. U kvizu treba postaviti prikazane životinje na odgovarajuće stanište.

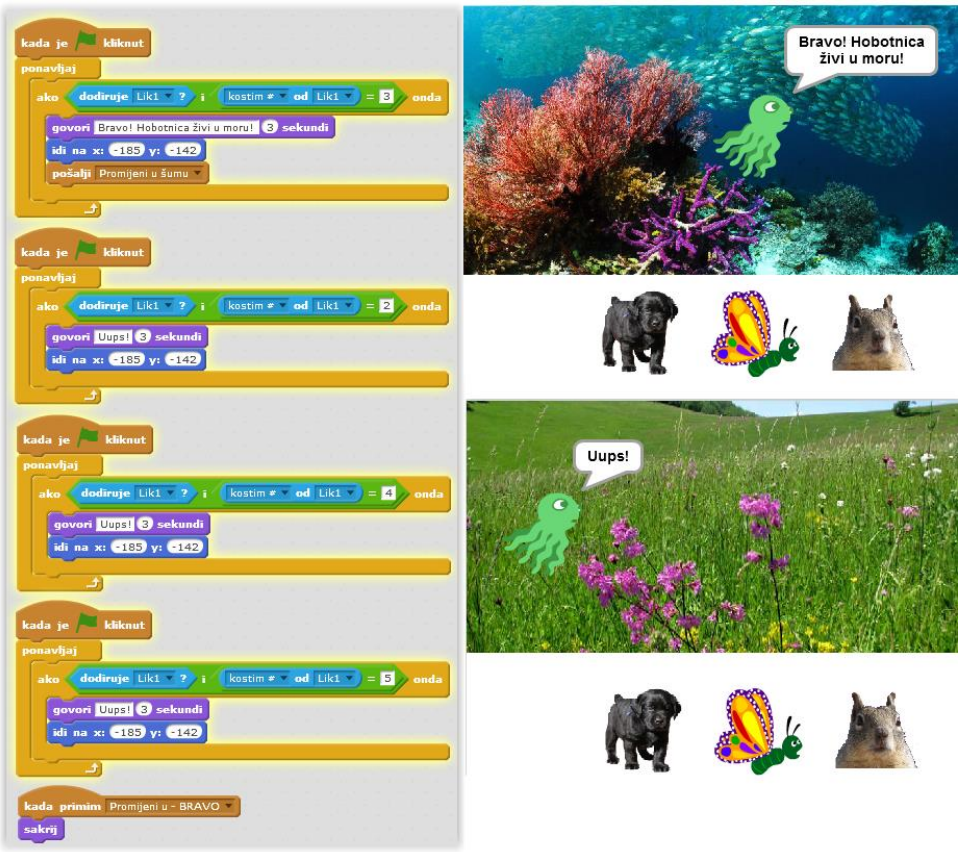


Slika 50. Program i skripta za mijenjanje kostima prvog lika

Cilj igre je postaviti životinje na odgovarajuće stanište tako što se klikom na pojedinu životinju ona odabere i postavi/odvuče na sliku prikladnog staništa. Ukoliko se životinja postavi na odgovarajuće stanište, javlja se poruka „*Bravo! Ova životinja živi u _____*“. U slučaju krivog postavljanja, životinja „izgovara“ poruku „*Uups*“ te je omogućen ponovni pokušaj/odabir (sve dok se na prikazano stanište ne postavi odgovarajuća životinja). Ovako učenici odmah dobivaju povratnu informaciju.

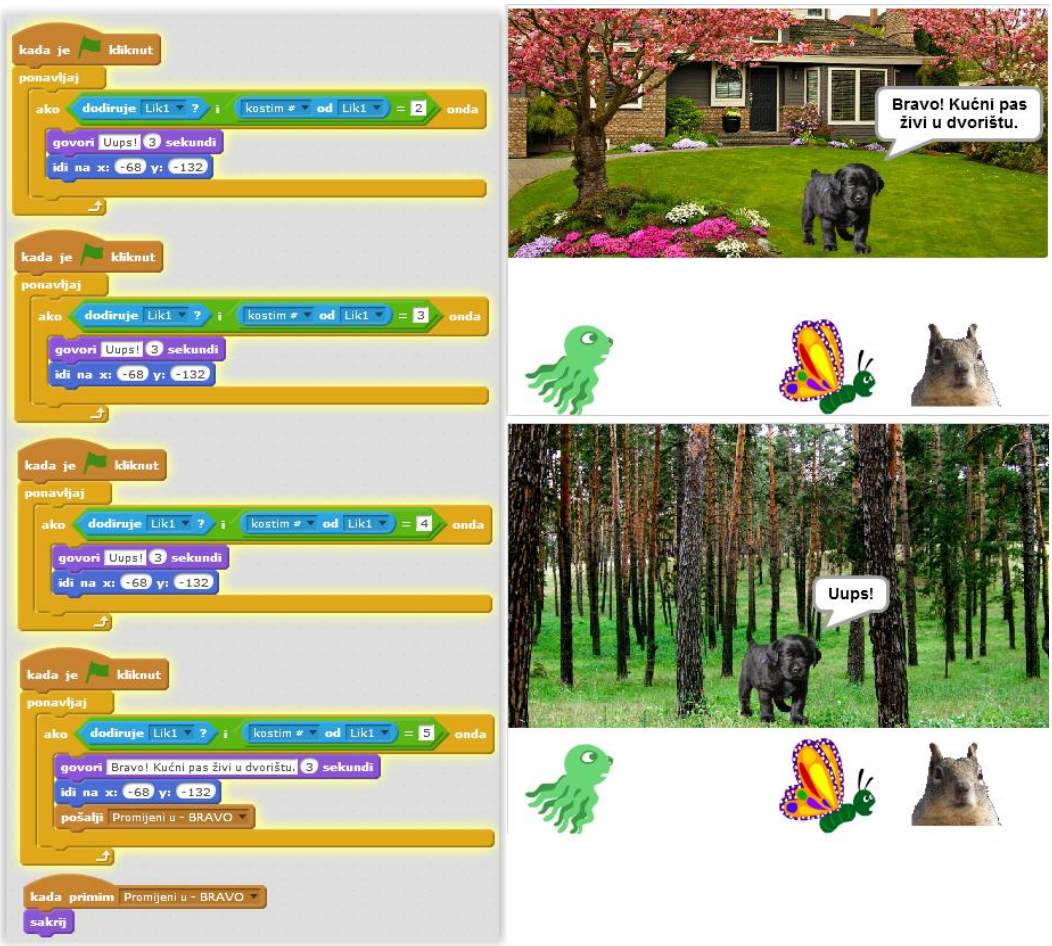
Kviz se sastoji od pet likova te svaki lik ima svoju skriptu. Prvi lik sadrži šest kostima koji se mijenjaju (naslovni slajd, travnjak, more, šuma, dvorište i završni slajd). Promjena kostima postignuta je naredbom „kad primim promijeni u“ iz bloka *Upravljanje* te naredbom „promijeni kostim u“ iz bloka *Izgled*.

Drugi lik je hobotnica čije je odgovarajuće stanište more. Ukoliko se hobotnica postavi u more, odnosno dodiruje kostim 3 (more) onda izgovara poruku „*Bravo! Hobotnica živi u moru*“ i šalje se poruka „*promijeni u šumu*“. Tad prvi lik prima poruku i mijenja kostim u *šuma*. U slučaju da hobotnica dodiruje bilo koji drugi kostim prvog lika (livadu, šumu ili dvorište), tad hobotnica „izgovara“ poruku „*Uups*“, vraća se na prvotni položaj/koordinate i omogućen je novi odabir.



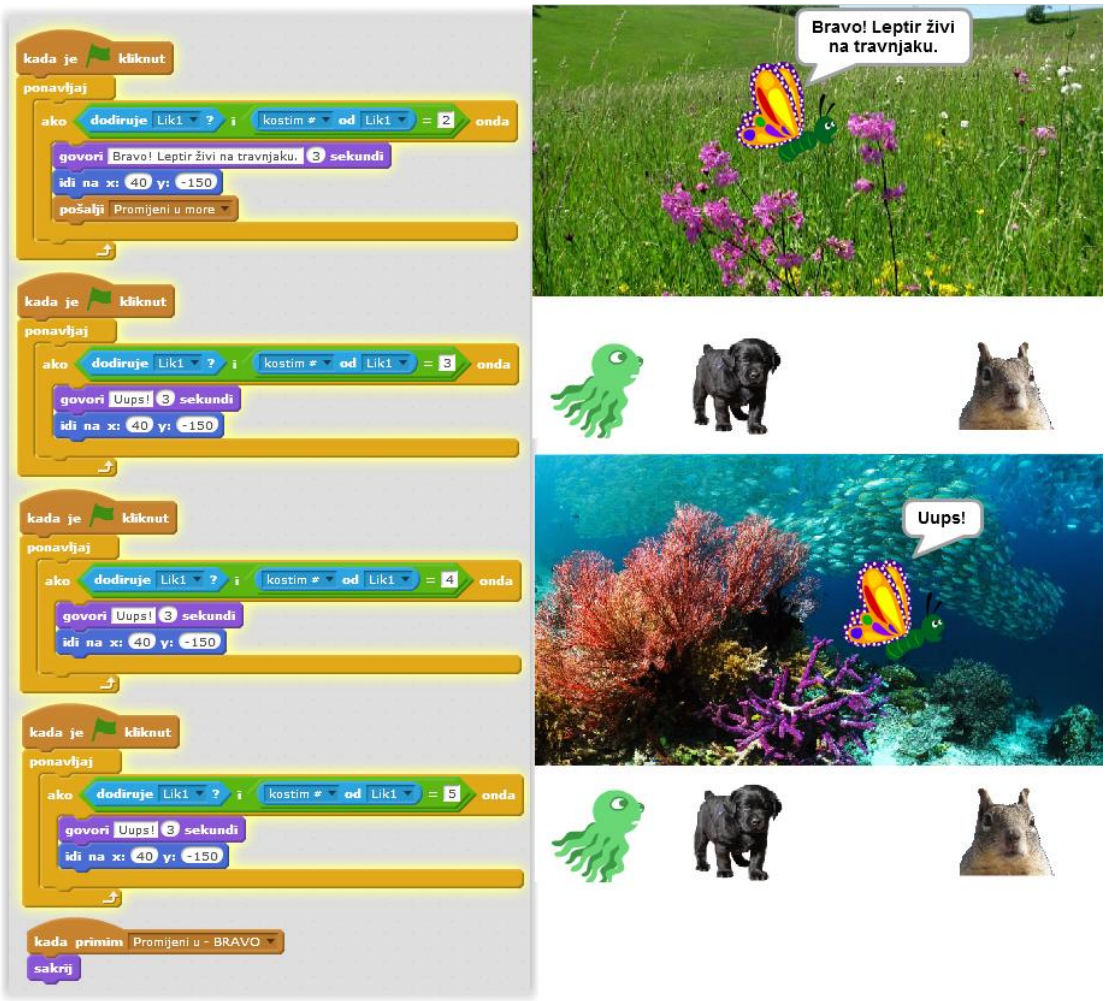
Slika 51. Program i skripta za lik "*Hobotnica*"

Drugi lik je kućni pas čije je odgovarajuće stanište dvorište. Ukoliko se psa postavi na dvorište, odnosno dodiruje kostim 5 prvog lika (dvorište) onda izgovara poruku „*Bravo! Pas živi na dvorištu*“. U slučaju da pas dodiruje bilo koji drugi kostim prvog lika (livadu, šumu ili more), tad „izgovara“ poruku „*Uups*“, vraća se na prvotni položaj/koordinate i omogućen je novi odabir.



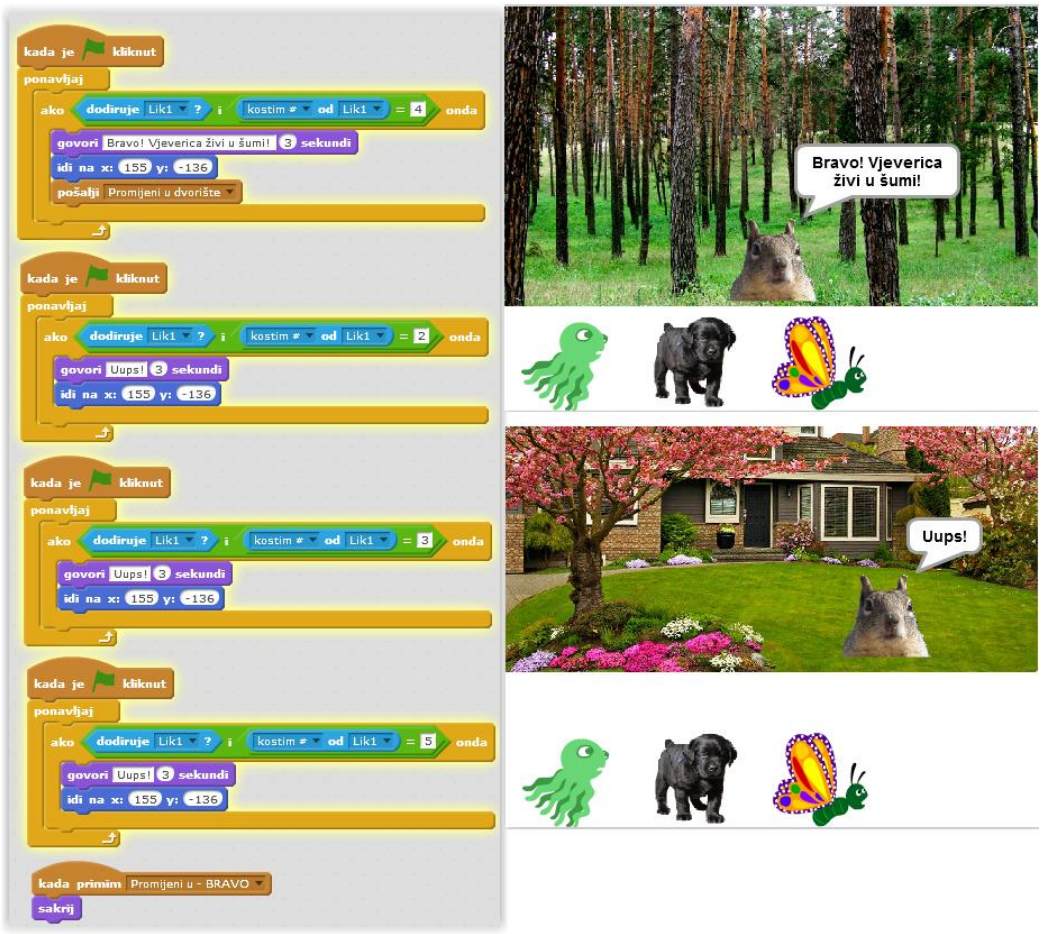
Slika 52. Program i skripta za lik "*Pas*"

Treći lik je leptir čije je odgovarajuće stanište livada. Ukoliko se leptir postavi na dvorište, odnosno dodiruje kostim 2 prvog lika (livadu) onda izgovara poruku „*Bravo! Leptir živi na livadi*“. U slučaju da leptir dodiruje bilo koji drugi kostim prvog lika (dvorište, šumu ili more), tad „izgovara“ poruku „*Uups*“ i ponuđen je ponovni odabir.



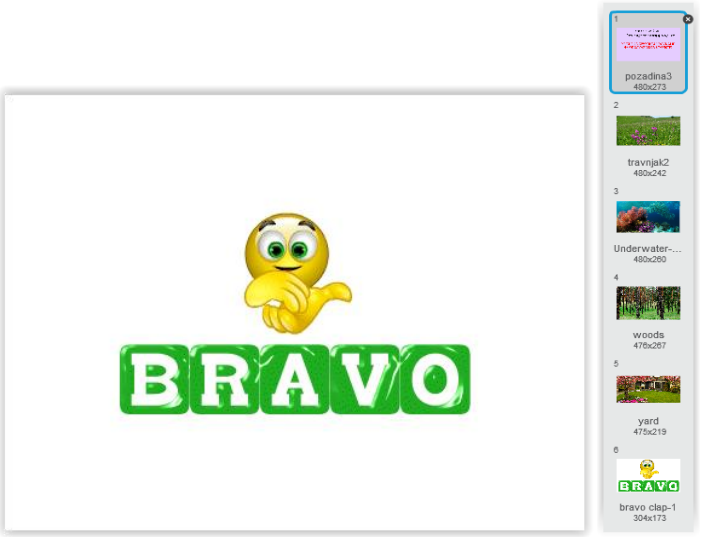
Slika 53. Program i skripta za lik "*Leptir*"

Zadnji lik je vjeverica čije je odgovarajuće stanište šuma. Kad je vjeverica postavljena u šumu, odnosno dodiruje kostim 4 prvog lika (šumu) onda izgovara poruku „*Bravo! Leptir živi na livadi*“ i šalje se poruka „*promijeni u more*“. Tad prvi lik prima poruku i mijenja kostim u *more*. U slučaju da vjeverica dodiruje bilo koji drugi kostim prvog lika (livadu, dvorište ili more), tad „izgovara“ poruku „*Uups*“ i ponuđen je ponovni odabir.



Slika 54. Program i skripta za lik "*Vjeverica*"

Kad su sve životinje postavljene na odgovarajuće stanište, prvi lik mijenja kostim u „*Bravo*“, koji označava da je kviz završen i točno riješen.



Slika 55. Poruka o uspješnom završetku kviza i svi kostimi prvog lika

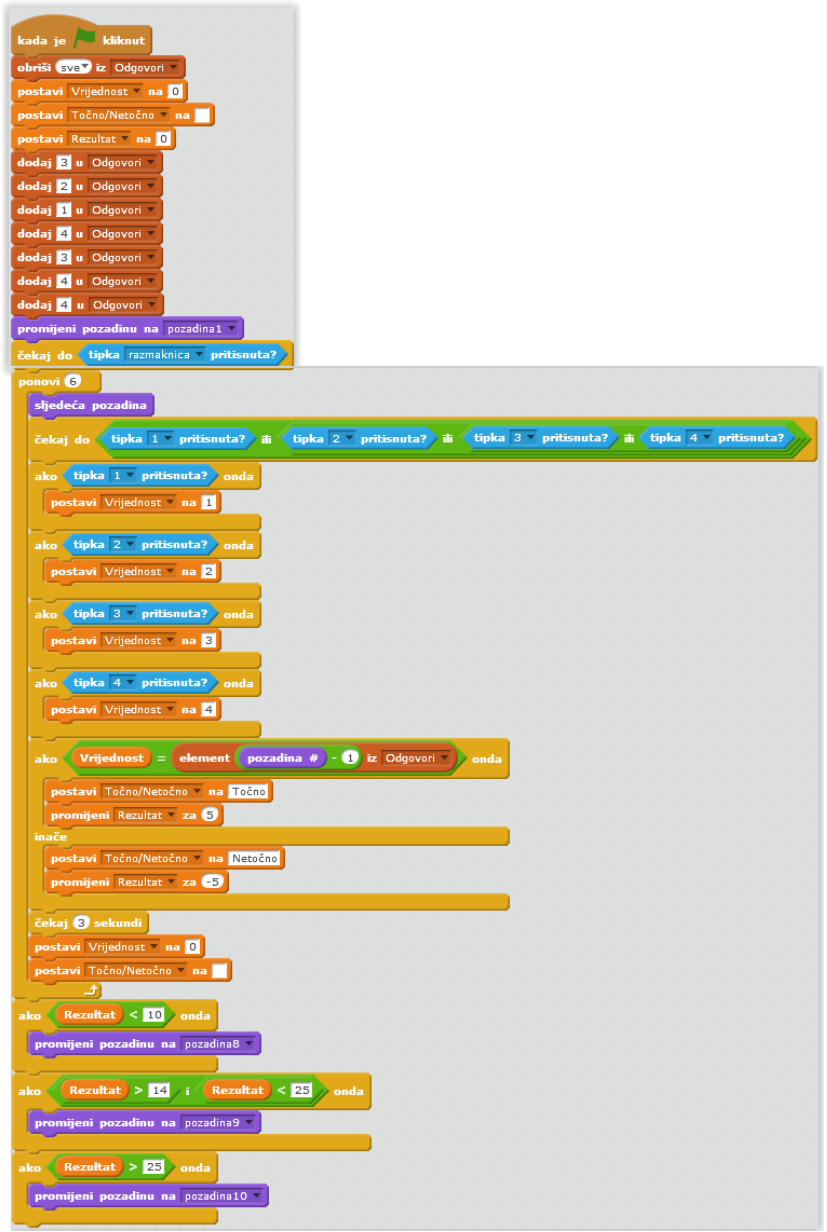
Sljedeći primjer iz prirode i društva je kviz o *Čistoći i zdravlju*. Kviz se sastoji od šest pitanja, a ona se prikazuju izmjenom pozadina. Svako pitanje ima četiri ponuđena odgovora *(1., 2., 3., 4.),* a kviz se rješava pritiskom na odgovarajući broj na tipkovnici *(1., 2., 3. ili 4.),*ovisno o tome pod kojim brojem korisnik smatra da se nalazi točan odgovor/tvrdnja.

Slika 56. Program i pozadine za kviz *Čistoća i zdravlje*

Prvo su postavljene tri varijable: *Vrijednost* (1-4), *Točno/Netočno* i *Rezultat*. Varijabla *Vrijednost* prikazuje uneseni broj (od 1 do 4), varijabla *Točno/Netočno* govori je li rezultat točan ili netočan, a varijabla *Rezultat* prikazuje ostvaren brojčani rezultat. Zatim je napravljena lista *Odgovori* koja sadrži brojeve pod kojima se nalaze točni odgovori. Za početak kviza potrebno je pritisnuti tipku *razmaknica* (*space*). Pojavljuje se jedno po jedno pitanje te se čeka odgovor (pritisak tipke *1., 2., 3., 4.*). Ako je varijabla *Vrijednost* jednaka elementu koji je unesen u listu *Odgovori*, onda se varijabla *Točno/Netočno* postavlja na „Točno“ i rezultat se povećava/mijenja za 5. Ako je varijabla *Vrijednost* različita od elementa koji je unesen u listu *Odgovori*, tad se varijabla *Točno/Netočno* postavlja na „Netočno“ i rezultat se umanjuje za 5. Najveći rezultat koji se može postići je 30 (šest pitanja * 5 bodova za svako pitanje), a najmanji je – 30. Ako je rezultat manji od 10, pojavljuje se osma pozadina koja korisniku prikazuje povratnu informaciju da ostvaren rezultat nije

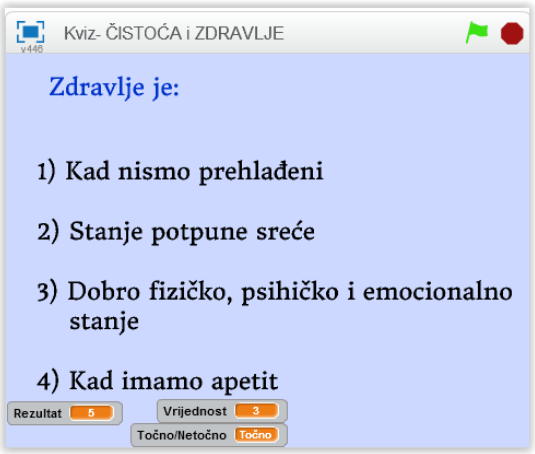
zadovoljavajući. U slučaju da je rezultat veći od 14 i manji od 25, pojavljuje se deveta pozadina koja prikazuje povratnu informaciju da je rezultat zadovoljavajući. Ukoliko je korisnik/učenik ostvario rezultat veći od 25, pojavljuje se deseta pozadina koja prikazuje opis *BRAVO*!

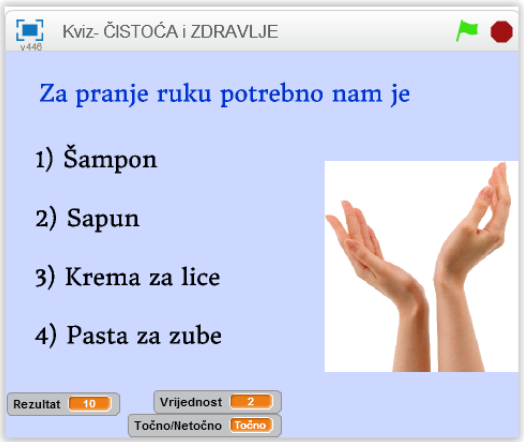
Slika 57. Lista odgovora i varijable "*Vrijednost, Rezultat, Točno/Netočno*"

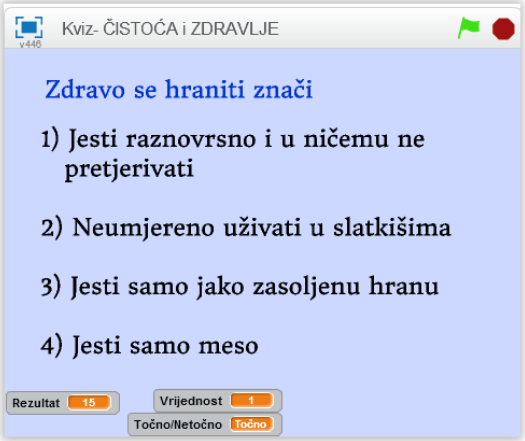


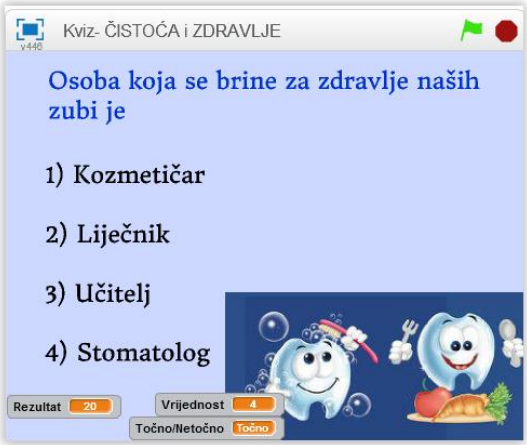
Slika 58. Skripta za izvođenje programa (kviz *Čistoća i zdravlje*)

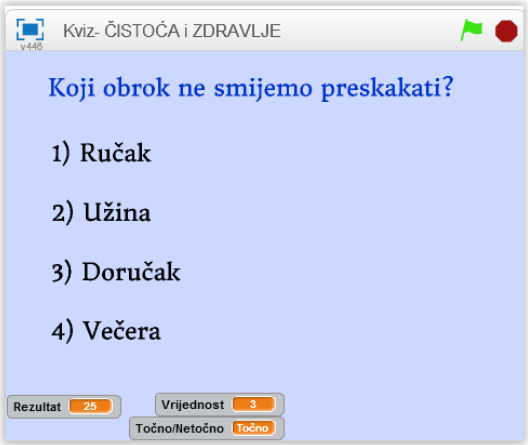
Izgled kviza *Čistoća i zdravlje* sa svim pitanjima

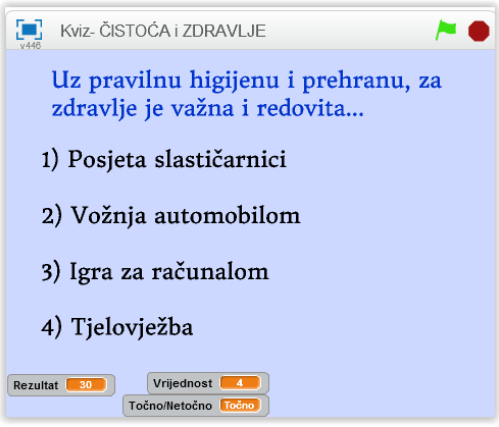


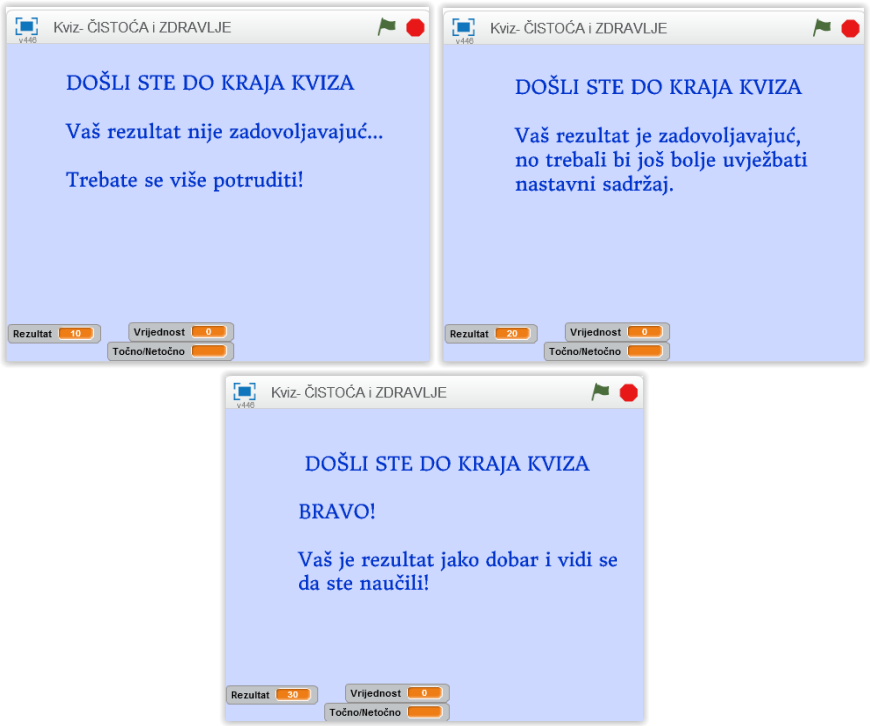










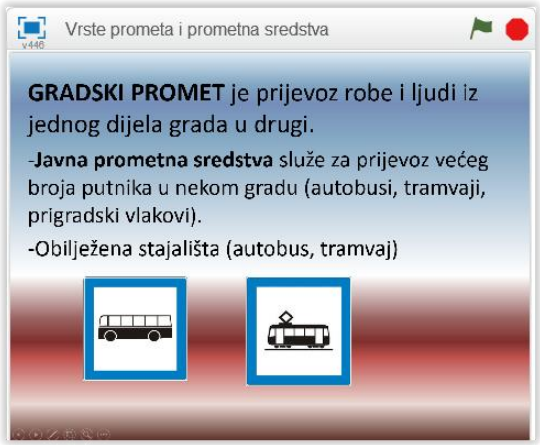


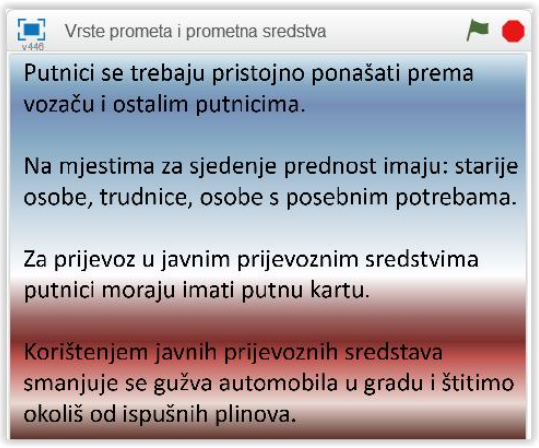
Slika 59. Povratne informacije po završetku kviza i ostvaren rezultat

Sljedeći primjer može učenicima pomoći prilikom obrade ili ponavljanja nastavnog sadržaja vezanog uz *Vrste prometa i prometna sredstva*. Izmjenom različitih pozadina o vrstama prometa (kopneni, vodeni i zračni), gradskom prometu, biciklistima, pravilima ponašanja u prometu te uporabom zvuka za pojedina prometna sredstva, dobiva se *slideshow* pomoću kojeg učenici mogu na zanimljiv način naučiti/uvježbati naveden nastavni sadržaj.

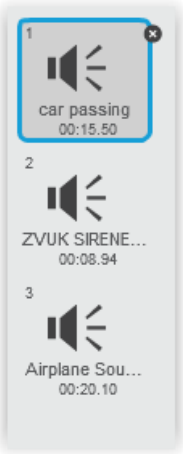
Slika 60. Program i skripta za *Vrste prometa i prometna sredstva*

Izgled programa - *Vrste prometa i prometna sredstva*





U skriptu su umetnuta tri zvuka: zvuk auta i motora, zvuk sirene broda te zvuk polijetanja zrakoplova i helikoptera. Ti zvukovi se reproduciraju na pozadinama koje prikazuju kopneni, vodeni i zračni promet.

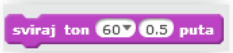


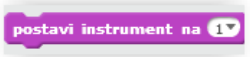
Slika 61. Upotrijebljeni zvukovi (auto i motor, brod, zrakoplov i helikopter)

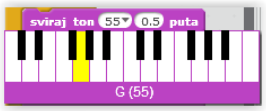
7.4. *GLAZBENA KULTURA*

Glazba je duboko ukorijenjena u čovjeku i sastavni je dio svih kultura svijeta te je važna za kvalitetan, skladan i cjelovit razvoj svakog pojedinca. Učenje i poučavanje predmeta *Glazbena kultura* u skladu je sa suvremenim znanstvenim spoznajama, koje upućuju na otvorenost i prilagodljivost, istraživačko i projektno učenje, ali i na potrebu uporabe informacijsko-komunikacijske tehnologije.

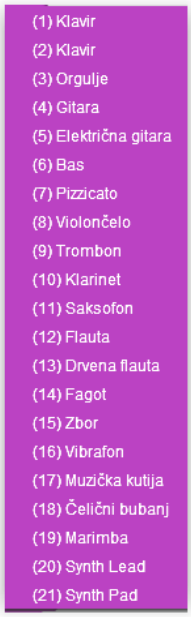
Uporabom programskog jezika *Scratch,* učenici se mogu na zanimljiv način upoznati s izgledom i zvukom različitih instrumenata, osnovnim pojmovima (metar, tempo, dinamika, melodija...) mogu gledati i analizirati stvorene spotove/animacije te rješavati interaktivne glazbene kvizove.

Pod blokom *Zvuk* postoje opcije 

i 

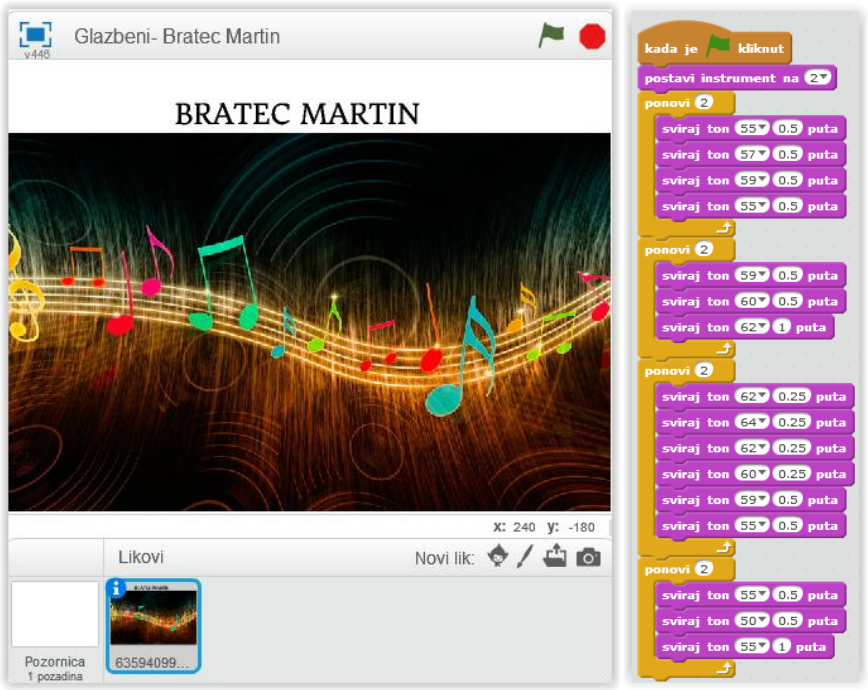
. Pritiskom na padajući izbornik kod „sviraj ton“ otvara se prozorčić koji prikazuje klavijature i tonove 

. Ovdje se može odabrati željeni ton klikom na tipku klavijature. Također, klikom na „postavi instrument“ otvara se padajući izbornik s popisom dostupnih instrumenata koji se mogu koristiti.



Slika 62. Popis dostupnih instrumenata (*Scratch 2*)

Kao primjer odabrane su pjesme *„Bratec Martin“* i *„Blistaj, blistaj zvijezdo mala“*. Odabirom željenih tonova i nota, njihovog trajanja, glasnoće i brzine izvođenja, moguće je stvoriti skripte koje reproduciraju razne pjesme koje se na nastavi obrađuju prema *Nastavnom planu i programu*.

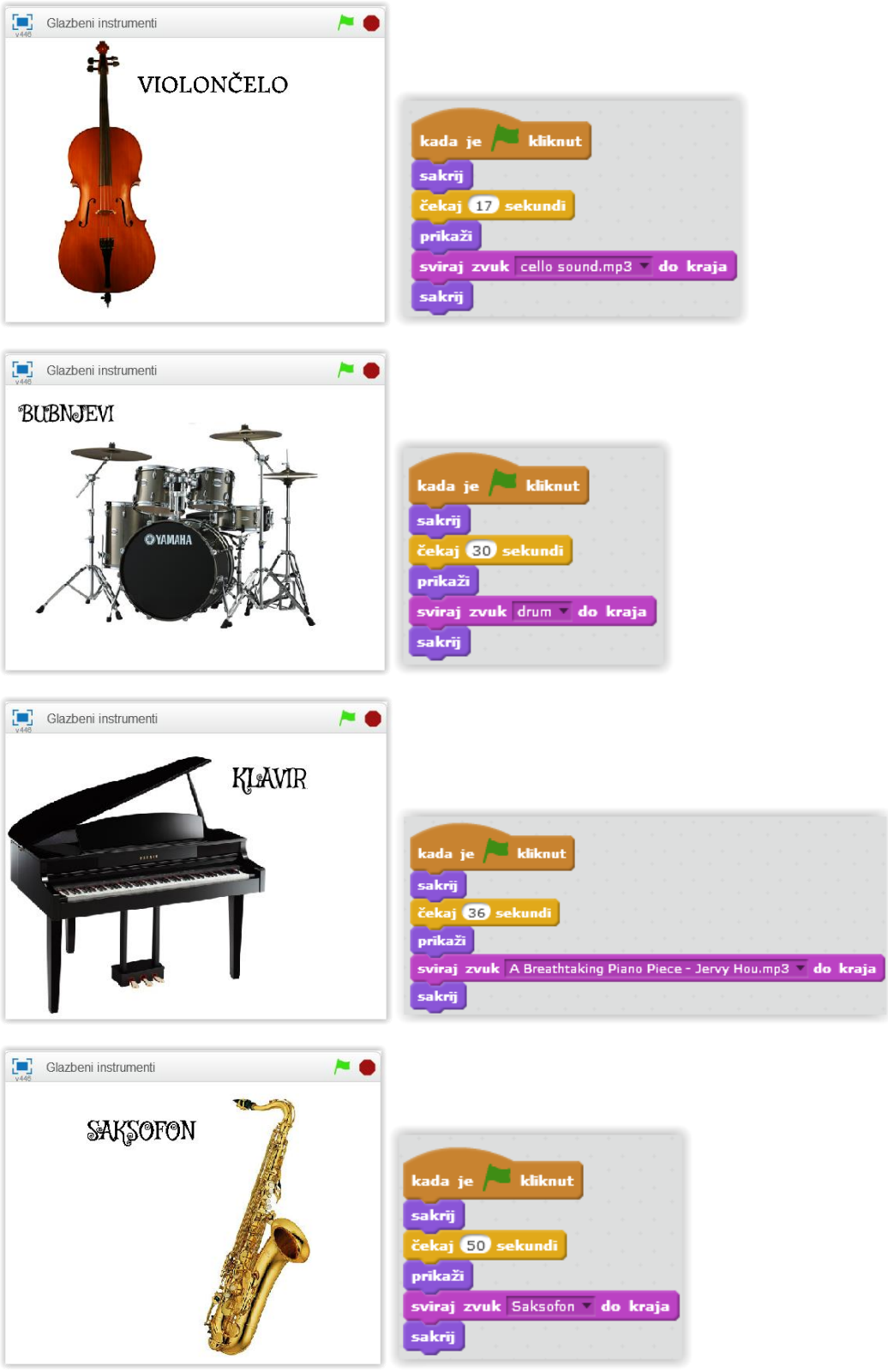


Slika 63. Izgled programa i skripte za pjesmu *Bratec Martin*

Slika 64. Izgled programa i skripte za pjesmu *Blistaj, blistaj zvijezdo mala*

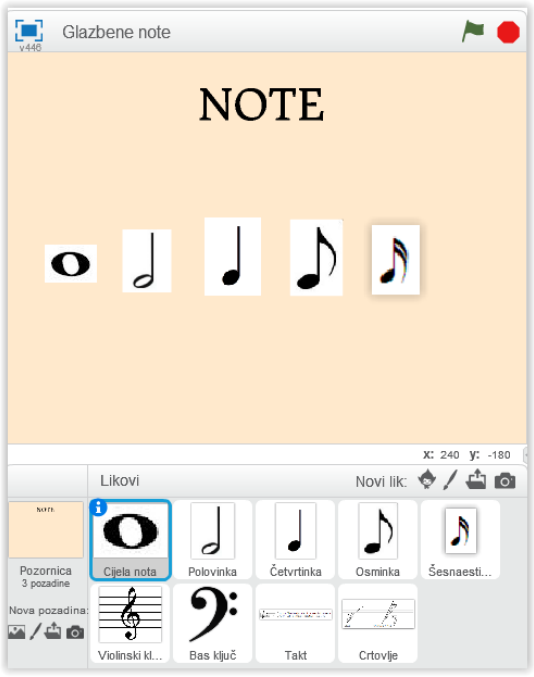
Sljedeći primjer skripte prikazuje *Glazbene instrumente* (gitara, violončelo, bubnjevi, klavir, saksofon) te reproducira njihov zvuk. Na ovaj način učenici mogu vidjeti sliku instrumenta i čuti zvuk, kako bi lakše upoznali i zapamtili pojedine instrumente. Program sadrži pet likova (instrumenata) te svaki lik ima svoju skriptu. Na početku skripte nalazi se pozadina *Glazbeni instrumenti* koja se prikazuje tri sekunde i zatim počinje blijediti primjenom efekta „duh" te se prebacuje na bijelu pozadinu. Tada se redom počinju prikazivati slike instrumenata, jedne po jedne te se reproducira njihov zvuk (zvukovi instrumenata uvezeni su sa računala).

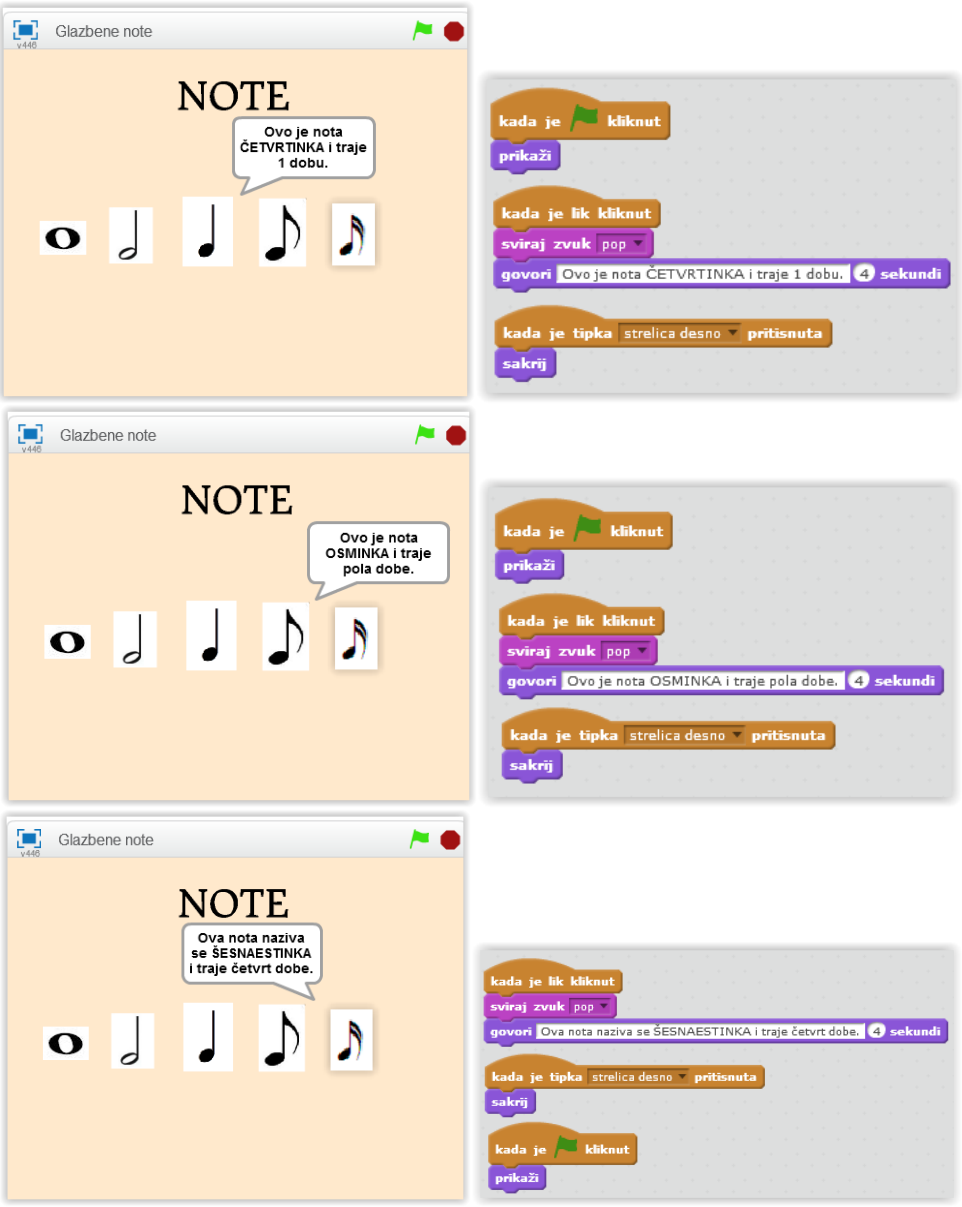
Slika 65. Program i skripta za pozadinu *Glazbeni instrumenti*



Slika 66. Slike i skripte za pojedine instrumente

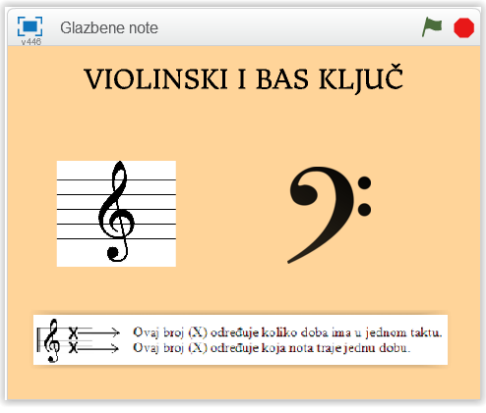
Pomoću sljedećeg primjera, učenici se mogu upoznati s različitim notnim vrijednostima, crtovljem te violinskim i bas ključem. Pritiskom/klikom na svaku notu začuje se zvuk „pop“, pojavljuje se naziv note te njezin opis. Svaka nota i ključ su zasebni likovi koji imaju svoje skripte.

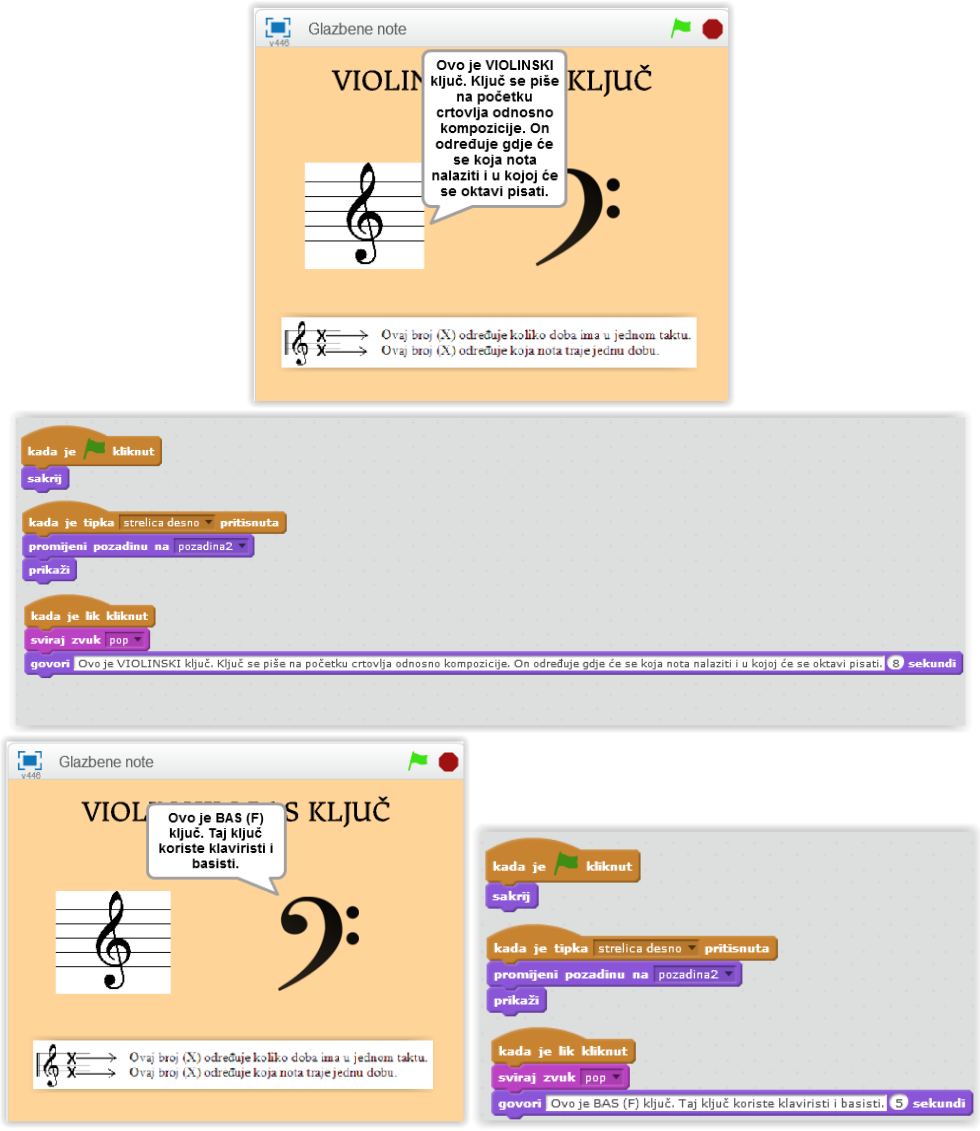




Slika 67. Program i skripte za različite note

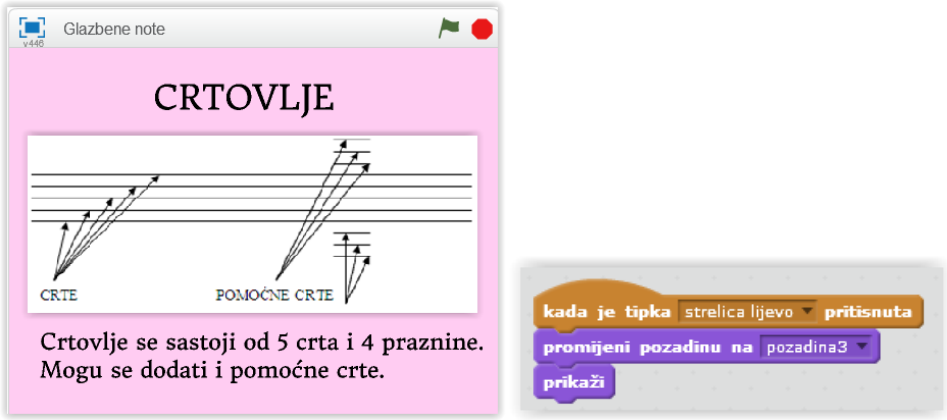
Pritiskom tipke „strelica desno“, pojavljuje se nova pozadina koja prikazuje violinski i bas ključ. Klikom na pojedini ključ prikazuje se njegovo objašnjenje.





Slika 68. Program i skripte za *violinski* i *bas ključ*

Pritiskom tipke „strelica lijevo“, prikazuje se nova pozadina s crtovljem i objašnjenjem.

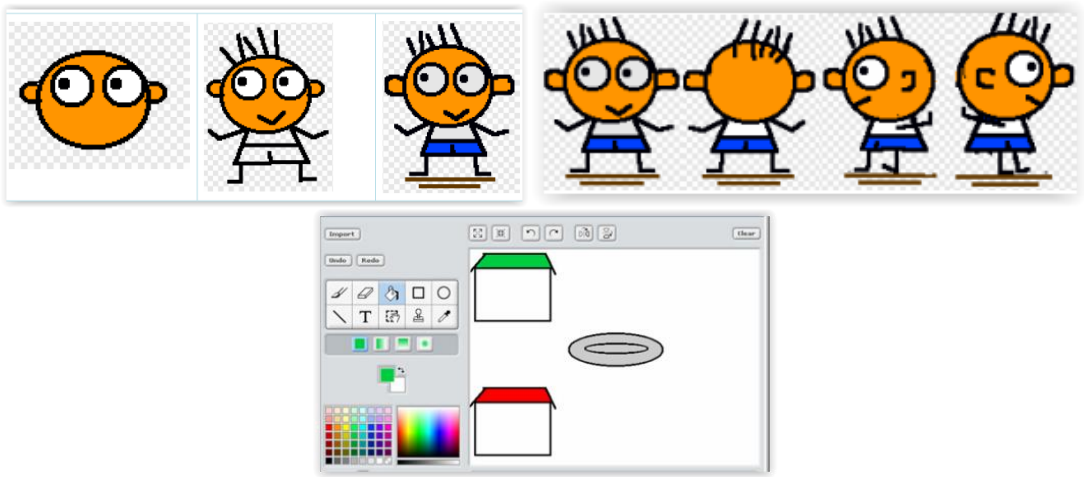


Slika 69. Program i skripta za crtovlje

7.5. *LIKOVNA KULTURA*

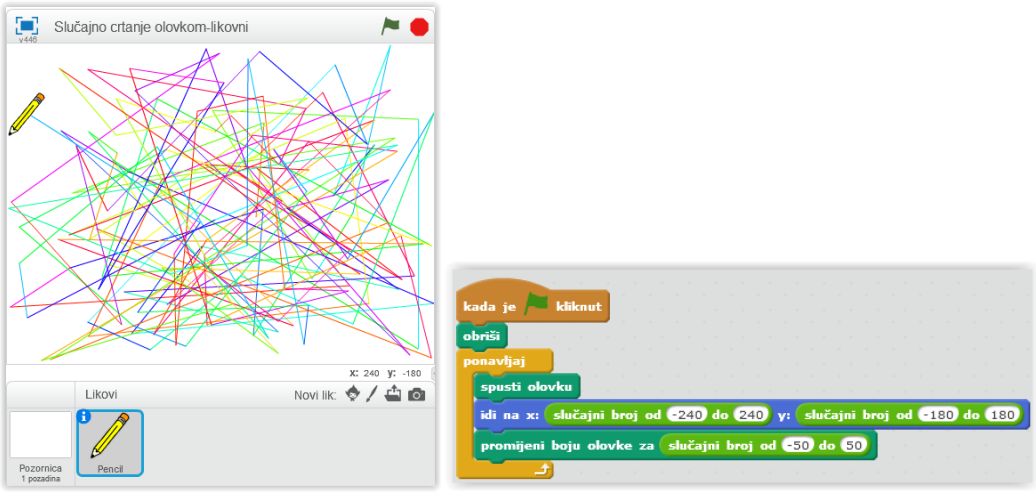
Program likovne kulture u osnovnoj školi temelji se na procesu istraživačkoga učenja i stvaranja. Struktura tog programa uvažava i prati razvojne faze učenikova likovnoga izražavanja i stvaranja, a od učitelja zahtjeva kreativan i fleksibilan pristup. Uporabom *Scratch*- a učenici mogu crtati svoje likove i pozadine u Paint Editoru te osmišljavati kompozicije poštujući pravilnosti poput proporcija, ritma, boja, kontrasta... Na satu likovne kulture učitelji mogu koristiti i prikazati različite programe (simulacije crtanja, reprodukcije, teorijske zapise, kvizove i sl.).

Jednostavni primjeri crtanja u Paint Editoru.



Slika 70. Primjeri crtanja u Paint Editoru

Primjer slučajnog crtanja olovkom u različitim bojama i smjerovima koji može poslužiti za raspravu i komentiranje (vrste crta, boje, apstraktnost).

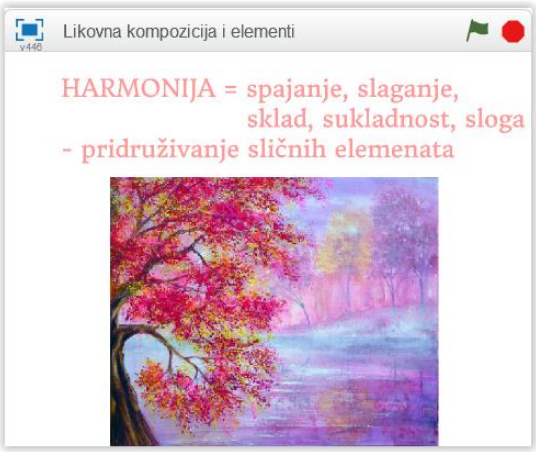


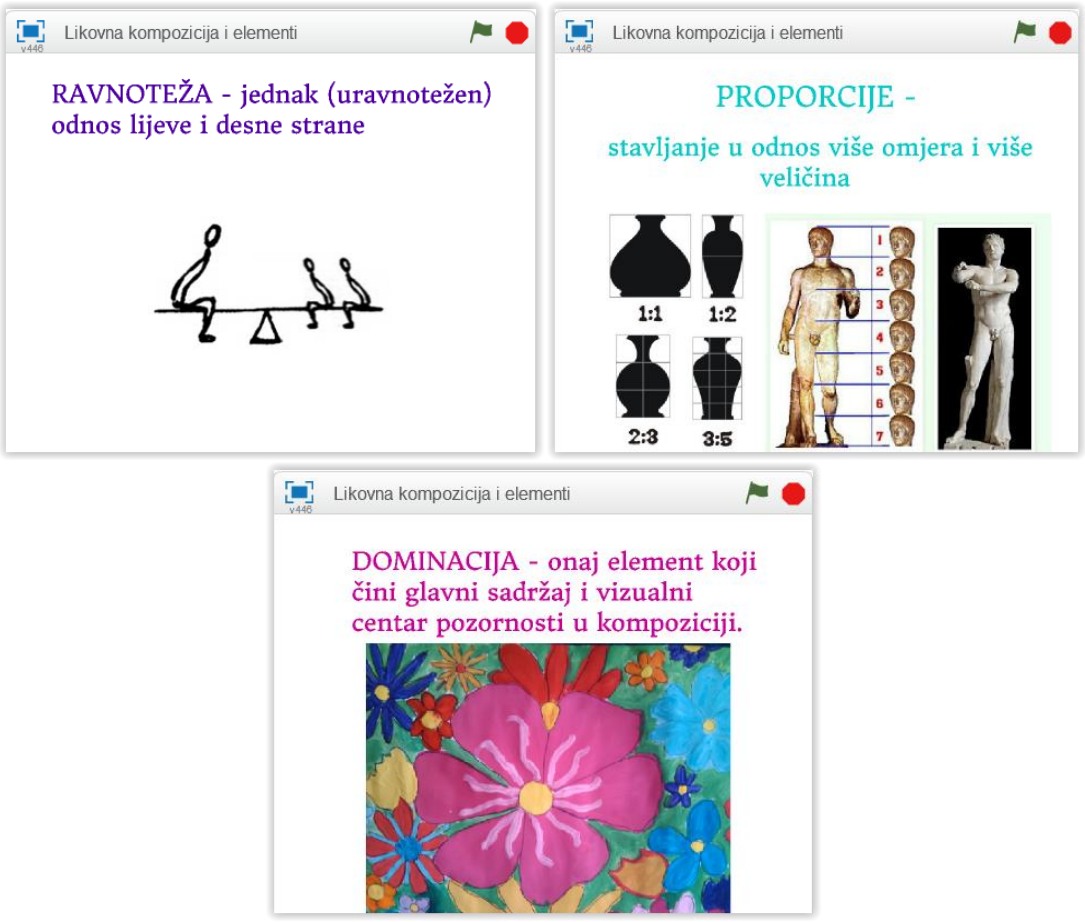
Slika 71. Izgled programa i skripte za slučajno crtanje olovkom

Sljedeći primjer može poslužiti u razumijevanju *Likovne kompozicije* i *likovnih elemenata (ritam, kontrast, harmonija, ravnoteža, proporcije, dominacija).* Program sadrži 9 stvorenih pozadina koje prikazuju objašnjenja i primjere kompozicije i likovnih elemenata. Pritiskom na zelenu zastavicu pojavljuje se naslovna pozadina, a svakim klikom na tipku „razmaknica“ mijenjaju se pozadine (tako korisnik upravlja programom).

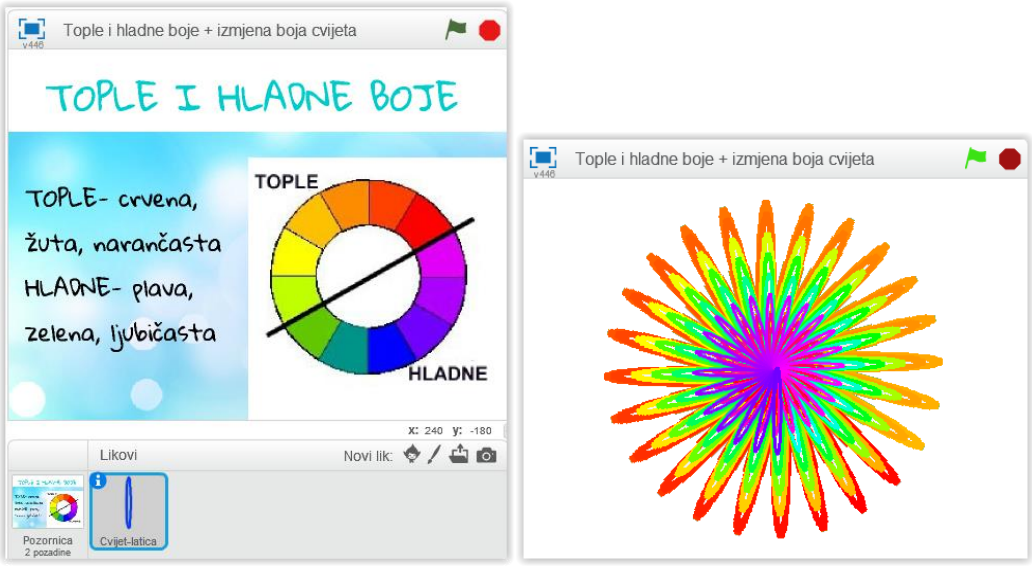
Slika 72. Izgled programa, pozadina i skripte za *Likovnu kompoziciju i lik. elemente*

Izgled programa – *Likovna kompozicija i likovni elementi*





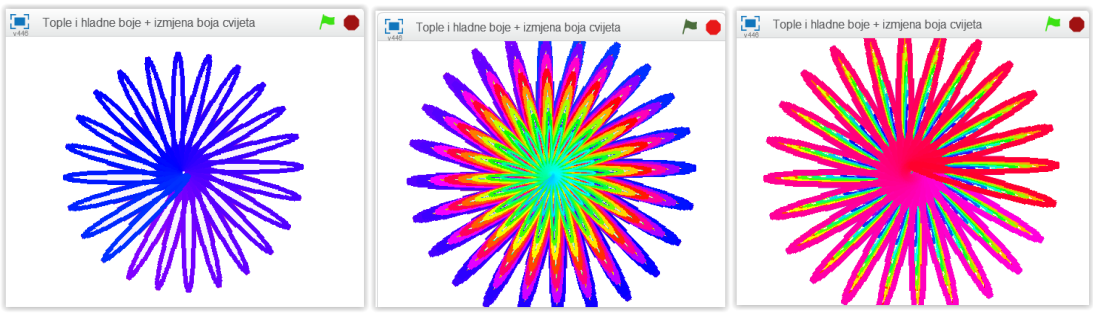
Sljedeći primjer može se koristiti za prikaz i učenje toplih i hladnih boja (2. razred) te se nakon toga pomoću promjene smjera i izmjena boja latice iscrtava efektan cvijet koji mijenja boje.



Slika 73. Izgled programa *Tople i hladne boje + cvijet*

Promjenom pozadine u *pozadina2*, pojavljuje se lik *latica*, koji je postavljen na veličinu 200% te se nalazi na koordinatama *x=0*, *y=0*. Naredbom *ponavljaj*, promjenom smjera/skretanjem za 15 stupnjeva udesno pa zatim ulijevo, ostavljanjem žiga te promjenom boje i veličine postupno se iscrtava efektan cvijet različitih boja.

Slika 74. Skripta za cvijet



Slika 75. Primjeri cvjetova

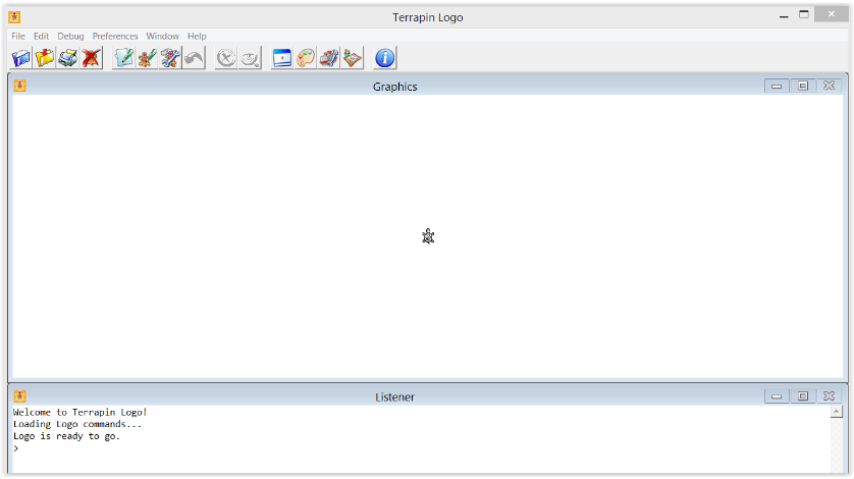
8. DRUGI PROGRAMSKI JEZICI U OSNOVNOJ ŠKOLI (*Logo, Basic, Pascal*) te PREDNOSTI I NEDOSTACI *SCRATCH*-a U ODNOSU NA NJIH

Kao jezici za programiranje u osnovnoj školi uglavnom se koriste *Logo* i *BASIC* te nešto rjeđe *Pascal*. U dijelu škola nastavnici uopće ne rade programiranje. Najčešći razlog za to jest činjenica da je informatika u osnovnoj školi izborni predmet i na učenike treba djelovati stimulativno, a većina učitelja (i učenika) programiranje smatra kompliciranim i demotivirajućim.

8.1. *LOGO*

U Hrvatskoj početnici u programiranju najčešće započinju s radom u nekoj verziji programa *Logo* koji se obrađuje u sklopu predmeta *Informatika* od petog razreda osnovne škole. Program je stvoren za obrazovnu uporabu, ponajprije za konstruktivističko učenje. Prva verzija programa *Logo* nastala je 1968. godine kao dio istraživanja provedenog u Bolt, Bernak & Newman, Inc. u Cambridgeu, a razvoj programa nastavio se pod vodstvom W. Feurziga i S. Paperta. Program je bio dizajniran isključivo za djecu, a glavna ideja *Loga* bila je omogućiti djeci crtanje crteža na ekranu dajući naredbe imaginarnom robotu (kornjači), pri čemu za pisanje programa, djeca trebaju kornjači kroz naredbe govoriti kuda ona treba ići, kada treba početi crtati, a kada se kretati bez crtanja. Među najpopularnijim verzijama programa *Logo* nalaze se *Terrapin Logo, Berkeley Logo, MicroWorlds, MsWlogo* i *NetLogo*. (Đurđević, 2014, str. 95). Komercijalna verzija programa *Terrapin Logo* najčešće se upotrebljava u osnovnim školama u Hrvatskoj.

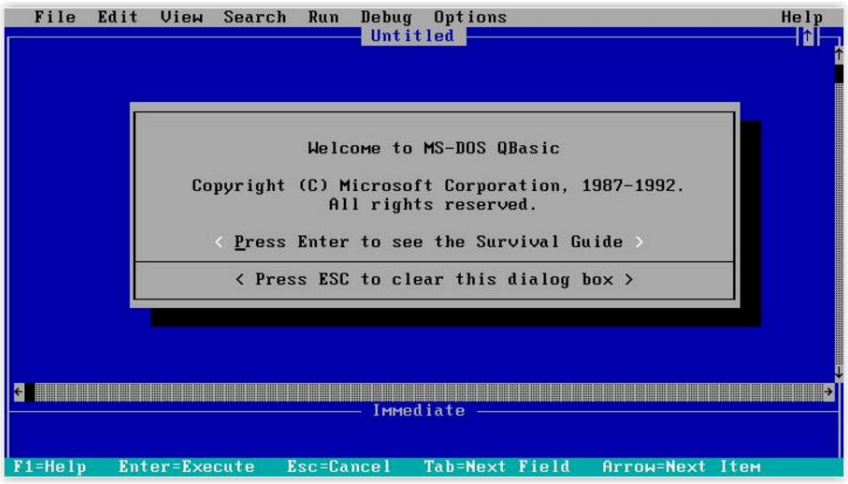
Papert, jedan od tvoraca *Logo*-a, tvrdio je da programski jezici trebaju imati „nizak pod“ (eng. *Low floor* - treba biti lako započeti), „visoki strop“ (eng. *High ceiling* - mogućnost da se s vremenom naprave sve kompleksniji projekti), i „široke zidove“ (eng. *Wide walls* – mogućnost podržavanja različitih tipova projekata za osobe sa različitim interesima i stilovima učenja). Zadovoljavanje „trojke“: niski pod/visoki strop/ široki zidovi nije lako. *Logo* je otvorio put novim idejama i po uzoru na njega napravljeni su novi alati za vizualno programiranje (Bubica, N., Mladenović, M., Boljat, I., 2014).



Slika 76. Sučelje programa *Terrapin Logo*

8.2. *BASIC*

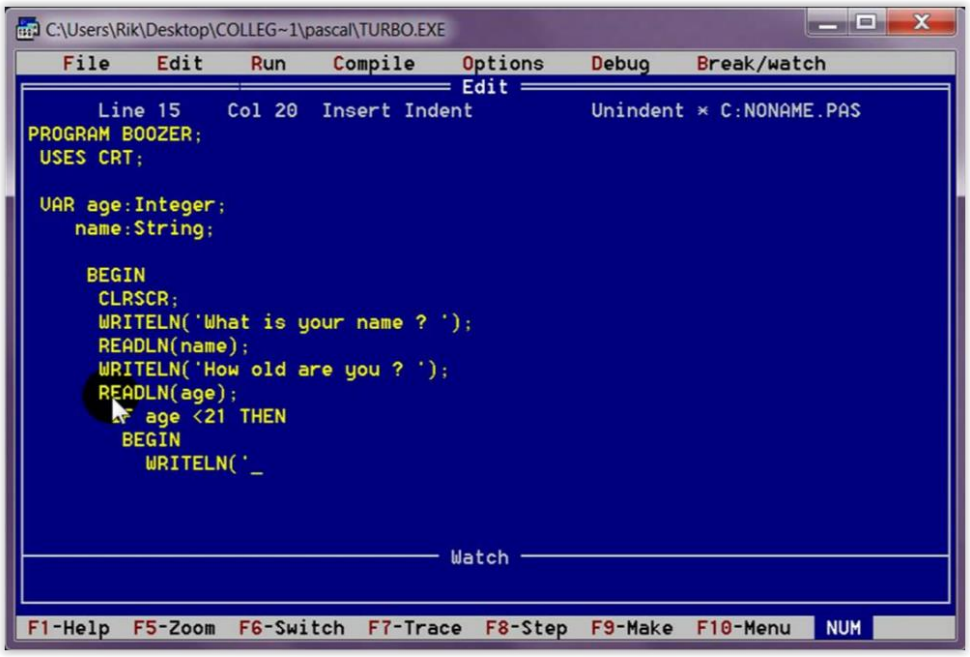
BASIC je kratica za *"Beginner's All-purpose Symbolic Instruction Code"*, jedan od najjednostavnijih i najpopularnijih programskih jezika koji se često koristi za učenje osnovnih koncepata programiranja. Osmišljen od strane Johna Kemeneya i Thomasa Kurtza, 1963. godine, postao je naširoko upotrebljavan na osobnim računalima diljem svijeta. Program služi za stvaranje jednostavnih programa i simulacija, a kao i većina drugih programskih jezika, za programiranje koristi naredbe i varijable koje zajedno čine programski kod kojim se piše program. Do danas je stvoren velik broj verzija ovog programskog alata, no najčešće upotrebljavana je Microsoftov *Quick Basic* poznatiji pod nazivom *Qbasic* (Rouse, 2011). On se koristi za učenje programiranja u osnovnim školama u Hrvatskoj.



Slika 77. Sučelje programa *QBasic*

8.3. *PASCAL*

Pascal je programski jezik stvoren od švicarskog znanstvenika Niklausa Wirtha 1970. godine kao jezik posebno pogodan za strukturalno programiranje (za razliku od danas opće prihvaćene OOP metode – objektno orijentirano programiranje). Baziran na temeljima programskog jezika *Algol* dobio je ime u čast matematičara i filozofa Blaisea Pascala. Prvenstvena namjena *Pascala* je u rješavanju problema "algoritamske prirode" gdje dolazi do izražaja strukturno programiranje. Strukturno programiranje karakterizira to da se razvoj programa tj. rješenje problema odvija po dijelovima. Točnije, rješenje čini skup modula od kojih svaki predstavlja i izvršava pojedinačnu, nezavisnu funkciju programa. *Pascal* je prvenstveno zamišljen kao jezik za učenje programiranja i razvijanja lijepog stila u programiranju te je postao više od samo akademskog jezika i počeo se koristiti komercijalno. Ubrzo nakon pojave i masivnijom upotrebom objektno orijentiranih jezika kao C++ ili Java tijekom sredine 90ih godina 20. stoljeća, *Pascal* počinje gubiti vodstvo u školama kao i u industriji. Neke inačice Pascala koriste se još i danas ne samo za učenje nego i za kreiranje i razvoj softvera (*Borland Delphy*) (CARNet, online tečaj „Programiranje u Pascal-u", https://tesla.carnet.hr/).



Slika 78. Sučelje programa *Turbo Pascal*

8.4. PREDNOSTI I NEDOSTACI *SCRATCH*- a U ODNOSU NA NAVEDENE JEZIKE

Uz jednostavan programski jezik kao što je *Scratch* mogu se naučiti najosnovniji elementi programiranja koji mogu biti korisni kasnije u životu. Jednom kad je interes za programiranje otkriven i pokrenut, ne može se reći gdje će završiti i u kojem će se smjeru kretati.

Jedna od najvažnijih prednosti programskog jezika *Scratch* u odnosu na *Logo, BASIC, Pascal* i druge programske jezike je ta što se programi u *Scratchu* ne pišu, već se slažu (princip slaganja koda kao LEGO kockica). Korisnik nije prisiljen učiti složene izraze kako bi opisao temeljne programske elemente već su mu ti elementi neposredno predstavljeni. Nadalje, blokovi programa su označeni različitim bojama i organizirani u sekcije, ovisno o tome što izvode/rade. Nema stroge sintakse već je vizualno jasno mogu li se neke naredbe spojiti ili ne. (Krpan, D., Mladenović, S., Zaharija, G., 2014). Blokovi programa se jednostavno mogu povući iz palete sa strane (unošenje jednostavnih ili naprednih naredbi), umjesto pisanja koda. Čak i ako kod nije valjan u nekom trenutku, projekt će svejedno raditi, dok u drugim programima program ne radi čim dođe do sintaktičke pogreške (npr. zaboravljen znak: točka-zarez i sl.). Još jedna prednost je ta da je Scratch potpuno besplatan program te ga mogu koristiti svi, za osobnu ili akademsku uporabu. Također, postoji prijevod na hrvatski jezik što dodatno olakšava rad, a *Scratch* ima i sjajnu zajednicu koja može pomoći korisniku u stvaranju vlastitih projekata.

Scratch je dobar izbor prvog programskog jezika za djecu (prva 4 razreda osnovne škole), radi učenja osnova rada na računalu (korištenje tastature, miša, mikrofona, kamere i ostalih uređaja), računalnoj grafici (crtanje jednostavnih likova i pozadina) i programiranju (izradi priča ili jednostavnih igara) (Buklijaš, 2010). Osnovna prednost vizualnog programskog jezika je ta da zahtijeva relativno malo inicijalnog znanja kako bi ga se moglo početi efikasno koristiti. To je zbog toga što su svi elementi programskog jezika, njegove semantike i sintakse vizualno reprezentirani te se korištenjem kontekstne ovisnosti dinamički sužava izbor elemenata koji se mogu upotrijebiti i na taj način korisnika se vodi kroz postupak programiranja. (Vukotić, D., Tanković, N., 2011).

Napredna mogućnost *Scratch*-a je interakcija više različitih projekata (na istom ili više udaljenih računala). Taj postupak se naziva „mesh“ i omogućuje projektima zajedničko korištenje varijabli i poruka (Krpan, D., Mladenović, S., Zaharija, G., 2014).

Nedostaci ovog programskog jezika jesu ti da online uređivač zahtjeva prijavu (izradu korisničkog računa za koji treba e-mail adresa) te stalnu internetsku vezu. Međutim, ovaj problem može se riješiti preuzimanjem programa na računalo. Ako učenici stvore svoje korisničke račune i tako sudjeluju u *Scratch* zajednici, bilo bi dobro da svoje korisničko ime i lozinku podijele s učiteljem/icom u slučaju da ih zaborave. Sljedeći nedostatak je da drugi korisnici mogu pregledavati i uređivati tuđe projekte te ih spremiti, preurediti ili preuzeti kao svoje i time pridobiti zasluge. Još jedan nedostatak je taj što potrebno mnogo „ručnog“ rada, npr. kopiranje skripti za svaki lik te mijenjanje naredbi u skriptama i njihovo ponovno dupliciranje te mnogo klikanja, povlačenja i ispuštanja blokova (*drag and drop*). Nadalje, kako korisnici rade sve složenije skripte tako područje za izradu postaje premalo i nepregledno. Ozbiljniji korisnici (programeri) navode kako *Scratch* zaostaje u prikazivanju koliko moćno programiranje može biti.

Unatoč navedenim nedostacima i ograničenjima, važno je napomenuti da će osobe koje savladaju ovaj programski jezik puno lakše shvatiti i razumjeti naprednije programske jezike (npr. *Python*), jer će već razumjeti koncepte petlji, odlučivanja, varijabli, toka programa, itd.

9. KRITERIJI ODABIRA POČETNOG PROGRAMSKOG JEZIKA I MOGUĆI PROBLEMI

Danas se razmatranja programskih jezika više fokusiraju na pedagoški aspekt učenja programiranja pa je raspon promatranih jezika širok. Predlaže se nekoliko različitih kriterija za odabir prvog jezika programiranja, no najsnažniji utjecaj na rad u ovom području došao je od kreatora četiri programska jezika: Seymoura Paperta (*LOGO*), NiklausaWirtha (*Pascal*), Guida van Rossuma (*Python*) i Bertranda Meyera (*Eiffel*). Unutar kriterija provjeravaju je li jezik pogodan za nastavu, nudi li opći okvir, promovira li novi pristup poučavanju programiranja, promovira li pisanje ispravnih programa, dopušta li probleme koji se rješavaju u "malim dijelovima", nudi li fleksibilna razvojna okruženja, nudi li podršku u obliku zajednice korištenja, je li otvorenog koda tako da svatko može pridonijeti njegovom razvoju, je li dosljedno podržan kroz razvojno okruženje, prate li ga dobri nastavni materijali te koristi li se samo u obrazovanju. (Bubica, 2015, str.7).

Bez obzira na prednosti i prikladnost pojedinih jezika, sintaksa programskog jezika ostaje značajna prepreka programerima početnicima. U analizi početničkih pogrešaka sintaktičke pogreške dolaze odmah nakon nerazumijevanja apstrakcije, polimorfizma i koncepata objektno orijentiranog programiranja. Programiranje u grafičkim okruženjima kao što je *Scratch* u kojima ne postoje sintaktičke pogreške može ublažiti stresan početak učenja programiranja. No, sintaktičke pogreške pojavljivat će se i dalje pri prijelazu na neki tekstualni programski jezik. Ipak, mnoge programere početnike, pri pisanju prvog računalnog programa, prati osjećaj neuspjeha jer programiranje izaziva neočekivana ponašanja programa kao što su sintaktička pogreška, pogreška izvođenja programa ili neka izlazna vrijednost programa koju početnik nije očekivao. Sve navedeno može se ubrojiti u oblike povratnih informacija. Povratne informacije ključne su za pomaganje učenicima u razumijevanju samog programa te načina na koji računala interpretiraju program. Vještine učenika moraju se razvijati baveći se i neispravnim programima kojima su učenici potaknuti na razvoj slučajeva za testiranje. (Bubica, 2015, str. 8). Mogući problem su i stavovi te stručnost učitelja koji iskazuju određeni otpor prema neophodnim promjenama u procesu poučavanja općenito, pa tako i u poučavanju programiranja.

10. ZAKLJUČAK

S obzirom da su današnje generacije od malih nogu okružene tehnologijom, potrebno je iskoristiti njihov interes i podignuti razinu logičkog i apstraktnog razmišljanja, a učenje programskih jezika upravo je idealna platforma za ostvarenje toga cilja.

Postoje razni stereotipi koji negativno utječu na percepciju i poimanje programiranja, koje ne mora biti teško za učenje. Izbor jezika koji se koristi u početnom programiranju uzrok je mnogih rasprava jer upravo taj „prvi“ jezik ima značajan utjecaj na programera. Osnovni cilj svakog uvodnog predmeta programiranja, a time i učenja prvog programskog jezika treba biti usvajanje osnovnih koncepata programiranja koji se mogu primijeniti jednako dobro u bilo kojem programskom jeziku. Također, programski jezik mora biti dovoljno intuitivan za početnika kako on ne bi odustao na samom početku (Bubica, 2015).

Početni programi trebali bi biti vizualni tako da se jasno vidi način spajanja naredbi, zanimljivi i jednostavni za korištenje. Upravo takav je programski jezik *Scratch* koji omogućava jednostavno stvaranje interaktivnih priča, igara i animacija te dijeljenje radova preko interneta. Ovaj program idealan je za upoznavanje sa svijetom programiranja na poticajan način, bez kompliciranih izraza jer omogućuje shvaćanje logike programiranja bez upoznavanja sintakse i pravila programskih jezika. Mogućnosti primjene *Scratch*- a u osnovnoj školi su neiscrpne jer je pogodan za učenje i razvoj sposobnosti programiranja na kreativan i učenicima pristupačan način, a ujedno zadovoljava sve potrebe današnjih koncepata i načina programiranja.

Učenje programiranja pomoću *Scratch*- a potiče djecu na razmišljanje, kreiranje te oživljavanje i realiziranje svojih ideja. Ono djeci daje samopouzdanje i vjeru kako mogu postati dizajneri i stvaratelji te ih uči međusobnoj suradnji. Učiteljima treba biti obveza dokazati i pokazati kako programiranje nije „teško i dosadno“, te otkloniti bezrazložni strah kroz primjenu programiranja u različitim predmetima i područjima, na stimulativan način. Također, učitelji bi svojim znanjem i entuzijazmom trebali kvalitetno obrazovati i motivirati učenike, jer programiranje kao digitalna vještina predstavlja novu pismenost koju je potrebno učiti od malih nogu baš kao učenje čitanja, pisanja i računanja.

LITERATURA

- Brennan, K., Balch, C., Chung, M. *Creative computing*. Harvard Graduate School of Education.; URL: https://theeducationfellowship-public.sharepoint.com/SiteAssets/Pages/Digital-Technologies/Creative%20Computing.pdf (pristupljeno 15.2.2016.)
- Bubica, N., Mladenović, M., Boljat, I. (2013). *Programiranje kao alat za razvoj apstraktnog mišljenja,* (str. 1-11)., URL: https://bib.irb.hr/datoteka/702093.Programiranje_kao_alat_za_razvoj_apstraktnog_miljenja-CUC-zbornik.pdf (pristupljeno 10.2.2016.)
- Bubica, N. (2014). *Strategije poučavanja i faktori koji utječu na unapređenje znanja programera početnika,* URL: http://www.pmfst.eu/wp-content/uploads/2014/06/Istraziva--ki-seminar1-Bubica.pdf (pristupljeno 10.2.2016.)
- Buklijaš, S. (2010). *Scratch- Vizualni programski jezik za djecu*, URL: https://cuc.carnet.hr/2010/images/b1_52a14.pdf?dm_document_id=182&dm_dnl=1 (pristupljeno 15.6.2016.)
- CARNet, online tečaj „*Programiranje u Pascal-u*", URL: https://tesla.carnet.hr/ (pristupljeno 13.6.2016.)
- Computing at School (CAS). *The Raspberry Pi Education Manual,* URL: http://pi.cs.man.ac.uk/download/Raspberry_Pi_Education_Manual.pdf (pristupljeno 15.2.2016.)
- Chiang, J. *Shall we learn Scratch Programming.*; URL: http://www.cs.sun.ac.za/rw146/doc/ScratchAnimation.pdf (pristupljeno 15.2.2016.)
- Đurđević, I. (2014). *Procjene studenata Učiteljskog studija o tri računalna programa namijenjena malim početnicima u programiranju*. Radovi Zavoda za znanstveni i umjetnički rad u Požegi, 3, (str. 93-108).
- Fessakis, G., Gouli, E., Mavroudi, E. (2013). *Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study*. Computers & Education, 63, 87–97.; URL: file:///E:/Documents/FAKS/DIPLOMSKI/2013_CE_FESSAKIS_GOULI_MAVROUDI_vF.pdf (pristupljeno 15.2.2016.)

- Krpan, D., Mladenović, S., Zaharija, G. (2014). *Vizualni programski jezici u visokom obrazovanju*.; URL: https://radovi2014.cuc.carnet.hr/modules/request.php?module=oc_proceedings&action=view.php&a=Accept&id=51&type=2 (pristupljeno 10.2.2016.)
- Lee Ford Jr., J. (2009). *Scratch Programming for Teens.* Boston: Course Technology.; URL: https://cs4hsatcss.wikispaces.com/file/view/Scratch+Programming+for+Teens.pdf (pristupljeno 15.2.2016.)
- Lipljin, N. (2004). *Programiranje 1*. Varaždin: TIVA tiskara
- Marji, M. (2014). *Learn to program with Scratch*. San Francisco: No Starch Press.; URL: http://kssrtmkt6.e-tech.my/bahan%20scratch_lee_1.pdf (pristupljeno 15.2.2016.)
- McManus, S. (2013). *Scratch Programming*. Southam: In Easy Steps.; URL: http://scratched.gse.harvard.edu/sites/default/files/179715213-scratch-programming-in-easy-steps-pdf-sampler.pdf (pristupljeno 15.2.2016.)
- Mujačić, E. *Scratch priručnik za polaznike*. Centar za održivi razvoj, URL: http://www.cor.ba/publikacije/brosure/Scratch_prirucnik.pdf (pristupljeno 15.2.2016.)
- Ministarstvo znanosti, obrazovanja i sporta. *Nacionalni okvirni kurikulum za predškolski odgoj i obrazovanje te opće obvezno i srednjoškolsko obrazovanje*, 2011.; URL: http://public.mzos.hr/lgs.axd?t=16&id=18247 (pristupljeno 10.2.2016.)
- Ministarstvo znanosti, obrazovanja i sporta. HNOS- *Nastavni plan i program za osnovnu školu*, 2006.; URL: http://public.mzos.hr/fgs.axd?id=14181 (pristupljeno 10.2.2016.)
- Olabe J.C., Olabe M.A., Basogain, X., Maiz, I., Castaño, C. (2011). *Programming and Robotics with Scratch in Primary Education*, (str. 356-363).; URL: http://www.formatex.info/ict/book/356-363.pdf (pristupljeno 11.2.2016.)
- Otvoreno društvo za razmjenu ideja (ODRAZI), *SCRATCH - Vodič za korisnike i korisnice*, Zagreb, URL: http://www.odrazi-se.org/scratch/Scratch_vodic.pdf (pristupljeno 10.2.2016.)

- Rouse, M. (2011). *BASIC (Beginner's All-purpose Symbolic Instruction Code)*, URL: http://whatis.techtarget.com/definition/BASIC-Beginners-All-purpose-Symbolic-Instruction-Code (pristupljeno 7.6.2016.)
- Scott, J. *Starting from Scratch.*; URL: https://www.royalsoced.org.uk/cms/files/education/computing_materials/Starting_from_Scratch_LEARNER.pdf (pristupljeno 15.2.2016.)
- *Scratch - Imagine, Program, Share (Scratch community)*, URL: https://scratch.mit.edu/
- Valić, B., Radovan, A., Pavlović, D. (2013)., *Korištenje programskog jezika Scratch za podučavanje osnova programiranja od malih nogu.;* URL: https://radovi2013.cuc.carnet.hr/modules/request.php?module=oc_proceedings&action=view.php&a=Accept&id=50&type=2 (pristupljeno 11.2.2016.)
- Vukotić, D., Tanković, N. (2011). *Alati za razvoj aplikacija bez kodiranja*.; URL: https://bib.irb.hr/datoteka/523767.Vukoti_Tankovi-Alati_za_razvoj_aplikacija_bez_kodiranja.pdf (pristupljeno 15.6.2016.)

Kratka biografska bilješka

Moje ime je Diana Budiša. Rođena sam 12.10.1992. godine u Bjelovaru. Osnovnu školu završila sam u Ivanskoj (O.Š. Ivanska), nakon čega upisujem Gimnaziju u Bjelovaru (opći smjer), koju završavam 2011. godine. Nakon završene Gimnazije odlučila sam upisati Učiteljski fakultet. Apsolventica sam pete godine na Učiteljskom fakultetu - Odsjek u Čakovcu, smjer razredna nastava, modul informatika. Razumijem, govorim i pišem na engleskom jeziku. Svoj diplomski rad odlučila sam pisati iz područja informacijskih znanosti (programiranje) jer me to oduvijek interesiralo i smatram da informatička pismenost i programiranje predstavljaju budućnost te pružaju neograničenu mogućnost stvaranja svega što ljudski um može zamisliti.

Izjava o samostalnoj izradi rada

Ja, Diana Budiša, izjavljujem da sam diplomski rad pod naslovom *Programski jezik Scratch i njegova primjena u osnovnoj školi* izradila samostalno uz vlastito znanje, pomoću stručne literature, uz mentorstvo doc. dr. sc. Predraga Oreškog.

Potpis: _____