Nauči se programirati in postani računalniški maček

Sonja Lajovic

SCRATCH

O knjigi

Poznavanje osnov določenega programskega jezika je danes del informacijske pismenosti prav vsakega posameznika. Žal pa prvi koraki v svet programiranja običajno niso najlažji. Že narobe postavljena vejica lahko povzroči, da program ne deluje. Prav tako je težko napisati programe, ki bi bili za začetnike zanimivi: z veliko grafike, akcije, uporabo animacij, zvoka ... Avtorji jezika Scratch so sestavili programski jezik in okolje, kjer začetniki zlahka napišejo privlačne in zanimive programe in ob tem spoznajo osnovne koncepte programskih jezikov. Čeprav se je jezika zelo enostavno naučiti, pa je kljub temu zelo dobro, če si lahko pri tem pomagamo z dobrimi navodili, privlačnimi zgledi in kopico idej. In vse to nam prinaša knjiga Sonje Lajovic, ki pomeni dobrodošlo novost med računalniško literaturo. Upam, da bo ta knjiga omogočila številnim, da se bodo pogumno spustili v lep in zanimiv svet programiranja.

O avtorici

Avtorica Sonja Lajovic je profesorica matematike in računalništva in poučuje matematiko in računalništvo na osnovni šoli. Že nekaj let uspešno vodi računalniški krožek, ki ga otroci radi obiskujejo prav zaradi možnosti, da ob zabavni igri z računalniki osvajajo temelje programiranja.

O programu Scratch

Scratch je eden novejših programskih jezikov (2007), ki je bil narejen za učenje programiranja. Otroci lahko z njim programirajo svoje računalniške igre, izdelujejo animacije ali interaktivne zgodbe. Svoje izdelke lahko prikazujejo na svetovnem spletu in jih delijo z drugimi, izmenjujejo izkušnje in se pri tem učijo. Program je razvil inštitut Media Lab z ugledne ameriške univerze MIT pod vodstvom Mitchela Resnicka. Avtorji so se pri svojem delu zgledovali po znamenitih kockah LEGO. Otroci tako namesto pisanja zapletenih programskih kod premikajo in zlagajo pisane grafične bloke, ki vsebujejo različne programske ukaze. Odziv je bil izreden - do danes je bilo po vsem svetu z uporabo Scratcha napisanih 1,662,266 programov.

Scratch je na svetovnem spletu doma na naslovu
http://scratch.mit.edu/
Na tej spletni strani boste poleg brezplačnega programa našli številne dodatne informacije in primere uporabe.

Mnenja o knjigi

Knjiga .. predstavlja izjemno pomemben prispevek k popularizaciji računalništva, ki se s tem res lahko začenja v zgodnjih otroških letih. Pa tudi starejši - učitelji in starši, ki so kakorkoli povezani z računalništvom in z otroško zvedavostjo, bodo radi posegli po njej.
Prof.dr. Saša Divjak, Fakulteta za računalništvo in informatiko v Ljubljano

Čeprav se je jezika Scratch zelo enostavno naučiti, pa je kljub temu zelo dobro, če si lahko pri tem pomagamo z dobrimi navodili, privlačnimi zgledi in kopico idej. In vse to nam prinaša knjiga Sonje Lajovic, ki pomeni dobrodošlo novost med računalniško literaturo.
mag. Matija Lokar, vodja Računalniškega centra in predavatelj na Fakulteti za matematiko in fiziko v Ljubljani

Jaz imam zelo rada program Scratch. Zelo rada igram igre na računalnik, ko pa sem spoznala Scratch, sem ugotovila, da jih je še bolje delati. Nikoli se ne naveličam. Všeč mi je tudi, ker moraš veliko razmišljati, da narediš kaj dobrega. In potem zelo rada poskušam to kar sem naredila.
Lara, 7. razred

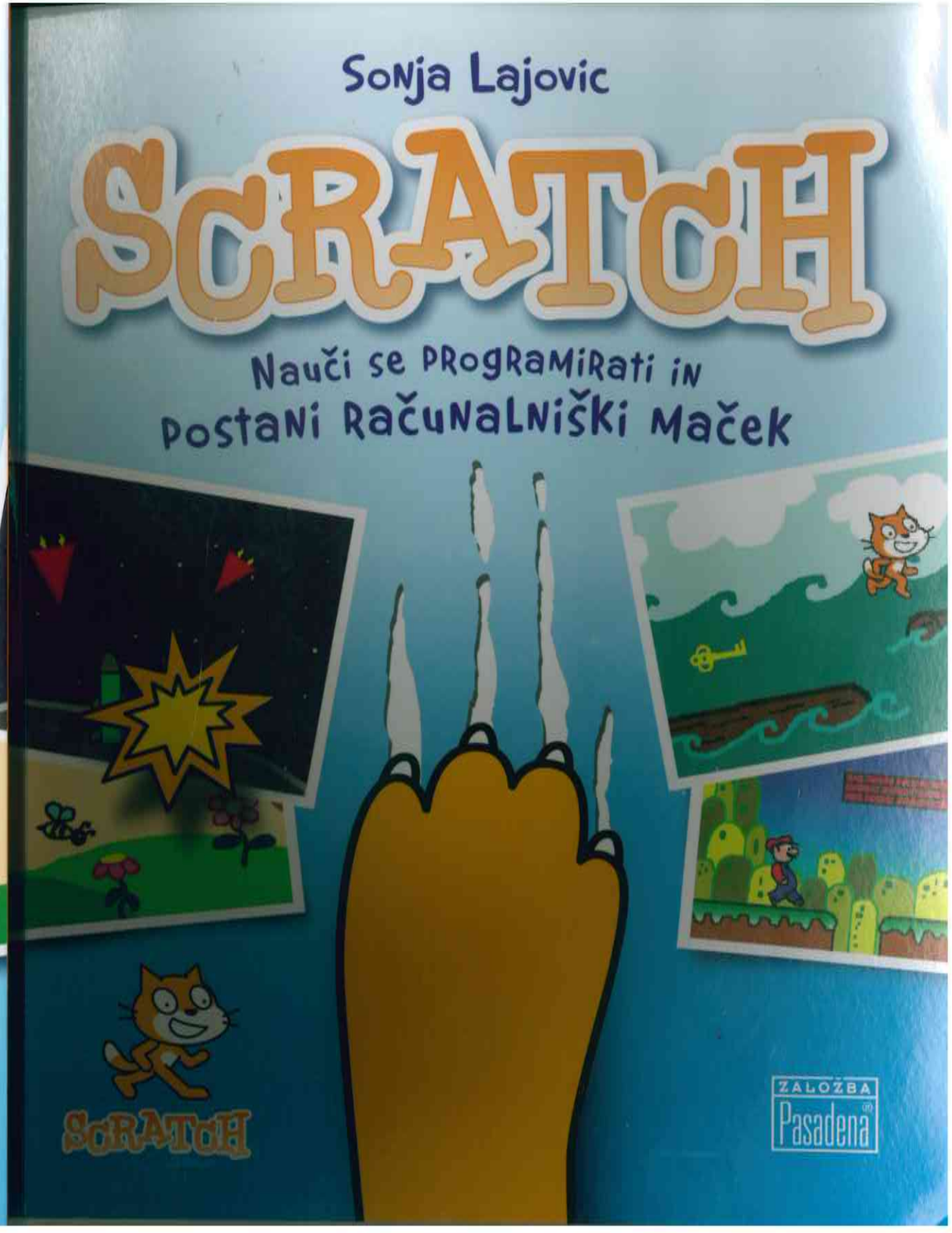
V knjigi mi je všeč, ker je na koncu rešitev in vse je dobr razložen. Pa narišeš lahko junake po svoje. Programiranje mi je bilo že prej všeč in ko sem dobil knjigo v roke mi je še bolj zraslo k srcu. Zdi se mi, da je zelo dobro razloženo ter spodbuja k zabavnemu načinu programiranja.
Igrice, ki smo jih naredili so bile zabavne in smo jih z veseljem sprogramirali nato pa še igrali. Najbolj so mi bile všeč igre s spremenljivkami ter programiranje igre Pac-Man.
Nika, Jera in Tamir, 9. razred

...Scratch pravzaprav najboljši korak za učenje rabe računalnika nasploh!
Založba Pasadena

Cena: 18,95 EUR
ISBN 978-961-6661-29-4
9 789616 661294

Založba Pasadena
Obiščite spletno knjigarno
www.pasadena.si
Založba Pasadena d.o.o.
1000 Ljubljana, Slovenija
telefon: 01 475 95 35

Sonja Lajovic
SCRATCH
Nauči se programirati in postani računalniški maček



CIP - Kataložni zapis o publikaciji
Narodna in univerzitetna knjižnica, Ljubljana

004.43(035)

LAJOVIC, Sonja, 1968-
Scratch / Sonja Lajovic. - 1. natis. - Ljubljana : Pasadena, 2011

ISBN 978-961-6661-29-4

255417600

Kazalo

O KNJIGI

Izkušnje pri delu z otroki v računalniških krožkih v osnovnih šolah so pokazale, da otroci (in njihovi starši) pogosto pričakujejo več kot le praktično delo in zabavo z računalniki. Mnogi bi se radi naučili osnov programiranja, spoznali in osvojili bi radi skrivnosti računalniškega izražanja. Nič čudnega, računalništvo je dandanes tako rekoč drugi materni jezik mladostnikov, temelj njihovega sporazumevanja z okolico in zlasti z vrstniki. Učitelji in mentorji se pri tem znajdejo v zadregi, saj literature, ki bi osnovnošolske otroke učila programiranja v slovenščini ni ravno veliko. Pravzaprav je skoraj ni.
Knjiga **Scratch** avtorice Sonje Lajovic to pomanjkanje vsaj deloma odpravlja. Knjiga otroke na kolikor je mogoče preprost in nazoren način seznanja s programom **Scratch**. To je eden novejših programskih jezikov, ki je bil narejen prav za učenje programiranja, za otroke od 8. leta starosti naprej. Otroci lahko z njim programirajo svoje računalniške igre, animacije ali interaktivne zgodbe. Svoje izdelke lahko prikazujejo na svetovnem spletu in jih delijo z drugimi, z njimi izmenjujejo izkušnje in se pri tem učijo. V knjigi Praskači se bodo otroci preko različnih primerov in nalog naučili osnovne logike, ki je značilna za klasično programiranje. Otroci se bodo naučili programirati klasične igre, kot sta PAC-MAN in Mario, različne strelske in druge igre, tisti bolj ustvarjalni pa bodo lahko programirali povsem svoje, originalne igrice. Avtorica je vsebino knjige zastavila v duhu aktivnega spoznavanja računalništva, ki otrok ne navaja na pasivno igranje različnih iger, pač pa jih spodbuja k ustvarjanju lastnih izdelkov. Seveda to ne velja samo za otroke, pač pa za ljudi vseh generacij, ki se želijo z računalništvom spoznavati aktivno in ustvarjalno. Scratch je namreč prav to - programiranje za vse.

»Programiranje za vse« pa ne pomeni, da bi vsi bralci in učenci postali programerji - »programiranje za vse« pomeni, da je (vsaj po mnenju založbe) Scratch pravzaprav najboljši korak za učenje rabe računalnika nasploh.

Knjiga je razdeljena na:
- uvodni del, ki podaja osnove programa Scratch
- naloge za učenje programiranja in razvijanje programerske logike
- programiranje igric

Avtorica Sonja Lajovic. profesorica matematike in računalništva, poučuje matematiko in računalništvo na osnovni šoli in že nekaj let uspešno vodi računalniški krožek, ki ga otroci radi obiskujejo prav zaradi možnosti, da ob zabavni igri z računalniki osvajajo temelje programiranja.

VSEBINA KNJIGE

Knjiga je razdeljena na tri dele, ki jih kot osrednji pripovedovalec povezuje maček Praskač. Njegova podoba krasi tudi naslovnico knjige.

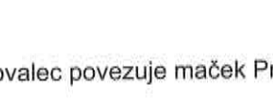
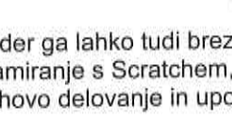
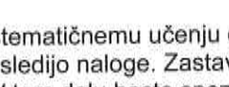
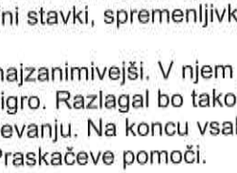
V **Uvodu** spoznate domačo stran programa Scratch, od koder ga lahko tudi brezplačno naložite na svoj računalnik. Praskač vas bo vpeljal v programiranje s Scratchem, nato pa boste z različnimi konkretnimi primeri spoznavali ukaze, njihovo delovanje in uporabo.

Drugi del, **Programirajmo,** je namenjen postopnemu in sistematičnemu učenju osnov programiranja. Nove vsebine so razložene s primeri, ki jim sledijo naloge. Zastavljene so tako, da jih lahko rešite sami, nato pa preverite svoje rezultate. V tem delu boste spoznali pomembne koncepte iz programiranja, kot so zanke, pogojni stavki, spremenljivke in podobno.

Tretji del knjige **Računalniške igre** bo za večino verjetno najzanimivejši. V njem je zbranih 10 iger. Praskač vas bo prijazno vodil in učil, kako narediti igro. Razlagal bo tako, da vas bo spodbudil k razmišljanju in včasih tudi k samostojnemu reševanju. Na koncu vsake igre so naloge za utrjevanje znanja, ki jih boste morali rešiti brez Praskačeve pomoči.

PREGLED PROGRAMSKIH KONCEPTOV JEZIKA SCRATCH

V tej knjigi boste spoznali pomembne programske koncepte, značilne za program Scratch. Predstavljeni so v spodnji tabeli, ki je povzeta po domači strani programa Scratch na naslovu http://info.scratch.mit.edu/sites/infoscratch.media.mit.edu/files/file/ScratchProgrammingConcepts-v14.pdf

Koncept	Razlaga	Primer
Zaporedje	Ukazi se izvajajo drug za drugim. Zato je pomemben njihov vrstni red.	
Zanke	Ukaza *ponovi* in *za vedno* omogočata, da se določeno zaporedje ukazov večkrat izvede.	
Pogojni stavki	Stavka *če* in *če-drugače pa* izvedeta zaporedje ukazov, če je izpolnjen pogoj.	
Spremenljivke	Spremenljivke lahko hranijo številčne podatke ali črkovne nize. Spremenljivke so lahko lokalne ali globalne.	

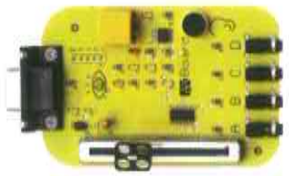
Koncept	Razlaga	Primer
Tabele	Podatki v tabelah so lahko števila ali črkovni nizi.	
Odziv na dogodke	Program se lahko odzove na različne dogodke, ki jih sproži uporabnik ali kak drug del programa.	
Niti	Več ukaznih blokov se lahko izvaja hkrati.	
Časovna usklajenost in sinhronizacija	Ukazni bloki s pošiljanjem in prejemanjem sporočil usklajujejo dejanja več figur in skrbijo za sinhronizacijo dogodkov.	
Vnos podatkov preko tipkovnice	Ukaz *ask* omogoča uporabniku, da natipka besedilo, ki ga nato prebere ukaz *answer*.	
Naključna števila	Ukaz *izberi naključno* generira naključna števila med določenima mejama.	
Logične operacije	Primeri takšnih operacij so logični in, logični ali in logični ne.	
Dinamična interakcija	Npr. spremenljivka *glasnost* omogoča interakcijo v realnem času.	

Koncepti, ki jih program Scratch ne podpira:

- procedure in funkcije,
- rekurzija,
- definiranje lastnih razredov objektov,
- dedovanje,
- branje/pisanje iz datoteke.

SCRATCH IN ZUNANJI SVET

Scratch ni le programski jezik za računalnik. Z njim se lahko povežete tudi z zunanjimi napravami. Ena takih je PicoBoard.



Picoboard je plošča s tipali za svetlobo, zvok in podobno ter drugimi orodji za interaktivno povezovanje s programom Scratch. Picoboard je prek USB vtiča povezan z računalnikom in tako omogoča, da se programi, napisani v Scratchu, odzivajo na zunanje dražljaje.

S Scratchem lahko upravljamo tudi LEGO robotke. Komplet Lego WeDo je ena novejših pridobitev LEGO kock. Poleg običajnih kock vsebuje krmilnik, ki se ga preprosto priključi na USB vtič na računalniku. V kompletu so tudi motor in dve tipali. Uporaba LEGO WeDo in programa Scratch tako nudi nove možnosti uporabnega, praktičnega programiranja, ko LEGO robotki počnejo to, kar ste jim ukazali.

SPOZNAJVA SCRATCH

UVOD

Živijo! Moje ime je Praskač in sem glavni junak programa Scratch. To je program, ki je namenjen prav tebi. Z njim lahko delaš zgodbe, animacije in računalniške igre. Svoje izdelke lahko deliš z drugimi na svetovnem spletu. Scratch spodbuja aktivnost, ustvarjalnost, razvija domišljijo, uči pa tudi osnovna pravila računalniškega programiranja ter logičnega razmišljanja.

Ne skrbi, pri učenju ti bom pomagal. V prvem delu knjige bova spoznala program Scratch in ugotovila, kaj vse zmore in kako ga uporabljati. Nato lahko izbiraš. Če želiš čim prej začeti s programiranjem iger, izberi tretji del knjige z naslovom Računalniške igre. Če pa se želiš najprej naučiti osnovnih pravil in logike programiranja, izberi drugi del knjige z naslovom Programiranje.

V prvem delu knjige si bova ogledala:

- kako namestiti program Scratch,
- spoznala bova njegovo programsko okolje,
- ugotovila bova, kaj je figura,
- izdelala bova najin prvi projekt in
- spoznala ukaze, s katerimi bova kasneje lahko naredila prave računalniške igre.

NAMESTITEV PROGRAMA

Ker na svojem računalniku verjetno še nimaš programa Scratch, ga morava namestiti. Program lahko naložiš na svoj računalnik s svetovnega spleta brezplačno. V času nastajanja te knjige je bila zadnja verzija programa 1.4.

Poveži se s svetovnim spletom in v brskalnik napiši naslov http://scratch.mit.edu/. Odprla se bo domača stran programa Scratch. Tam najdeš podrobnosti o programu, pomoč, razne videolekcije, že narejene igrice in druge projekte. Domačo stran lahko nekoliko pregledaš, nato pa bova Scratch namestila.

Namestitev bo takšna kot pri katerem koli drugem programu. Preden se lotiva dela, morava izbrati jezik. Seveda bova namestila slovensko verzijo Scratcha.

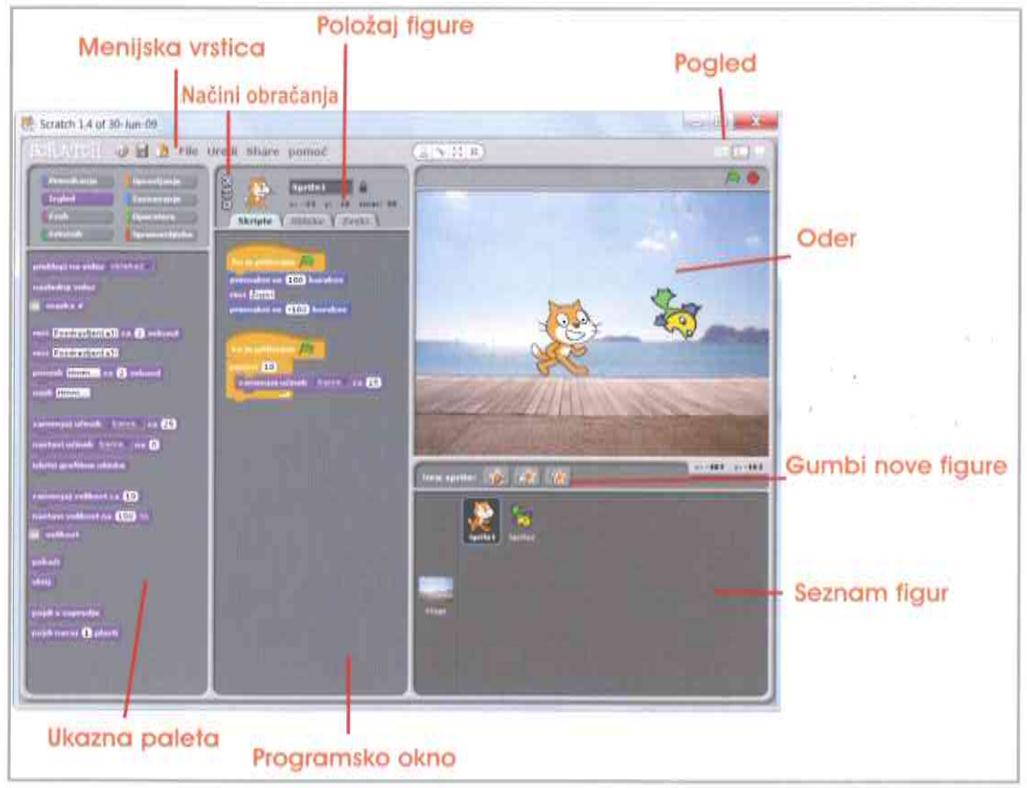
Klikni na gumb *Download Scratch*. Odpre se obrazec, ki ga lahko izpolniš, če želiš prejemati posodobitve programa. Nato klikni na gumb *Continue to Scratch Download*. Program Scratch podpirajo različni operacijski sistemi, Windows, MAC OS X in Linux. Izberi operacijski sistem, ki ga uporabljaš, program se bo nato samodejno prenesel v določeno mapo.

Ko se bo program prenesel, dvoklikni na namestitveno datoteko. Odpre se okno s čarovnikom, ki te bo vodil do konca namestitve programa.

Na koncu namestitve klikneš le še na gumb Dokončaj. Program je tako nameščen in lahko ga začneva uporabljati. Ko je namestitev končana, se program zažene sam od sebe. Na namizju se pojavi tudi bližnjica za program Scratch.

GRAFIČNI VMESNIK

Delo v Scratchu poteka s pomočjo grafičnega vmesnika, ki omogoča pisanje in preizkušanje programov, grafično oblikovanje in igranje igric. Grafični vmesnik vsebuje več polj, ki jih vidiš na spodnji sliki.



Kot lahko opaziš, je v grafičnem vmesniku večina besed napisana v slovenščini. Prevedeni so tudi vsi ukazi. Žal slovenski prevod programa Scratch za zdaj še ni narejen v celoti in natančno. Midva bova kljub temu uporabljala slovenske besede, ki pa jim bova, če bo treba, v oklepaju pustila angleški prevod. Na primer: v menijski vrstici ni prevedena beseda File. File po slovensko pomeni datoteka, zato bova rekla: »V menijski vrstici izberi *Datoteka*.« Originalno besedo v angleščini bova pustila v oklepaju. Torej: »V menijski vrstici izberi *Datoteka* (File).« Se strinjaš? Slovenščina je kul (cool)! ☺

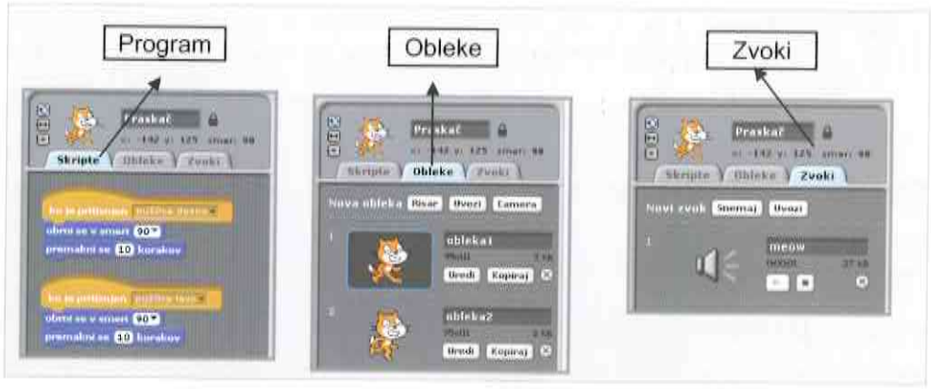
Kot vidiš, se na grafičnem vmesniku največkrat ponovi beseda sprite. Zakaj je tako pomembna in kako jo prevesti, bova spoznala v naslednjem poglavju.

SPRITE ALI FIGURA

Glavni elementi Scratcha so bitja in predmeti, ki jim v angleščini pravimo sprite. Po slovensko sprite pomeni: škrat, vila, duh, prikazen. V slovarju računalniških izrazov pa sprite pomeni figura, sličica. Midva bova za besedo sprite uporabljala besedo figura, saj je ta izraz najprimernejši glede na njegov namen in vsebino. Ne pozabi, **sprite** je odslej za naju **figura**.

Figuro lahko narišeš ali pa uporabiš katero koli sliko s svojega računalnika.

- Figuri lahko daješ navodila, kako naj se premika, kakšno glasbo naj predvaja, kakšen naj bo njen videz. Zbirki navodil pravimo **program ali skripta**.
- Figuri lahko zamenjaš videz tako, da ji zamenjaš **obleko**.
- Vsaka figura ima lahko tudi svoj seznam **zvokov**, ki jih posnameš ali poiščeš na svojem računalniku.



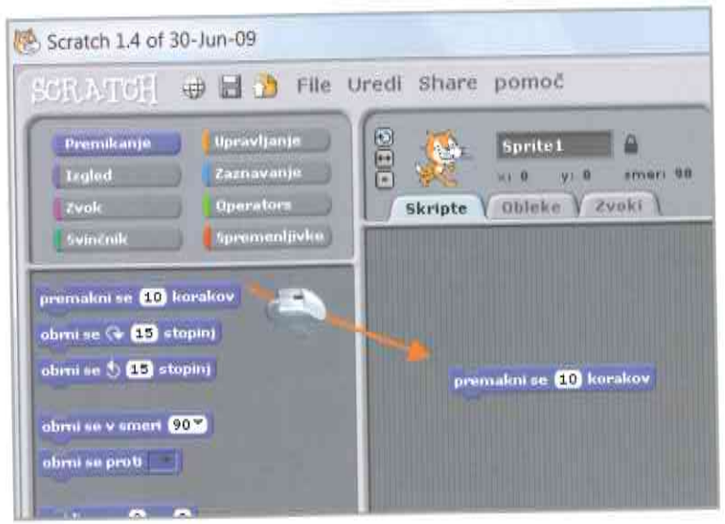
Na seznamu figur so prikazane ikone vseh figur, ki jih potrebuješ za posamezni projekt.

Če želiš neki figuri napisati program, ji narediti obleko ali dodati zvok, jo moraš najprej izbrati. To storiš tako, da klikneš na njeno ikono.

NAJIN PRVI PROJEKT

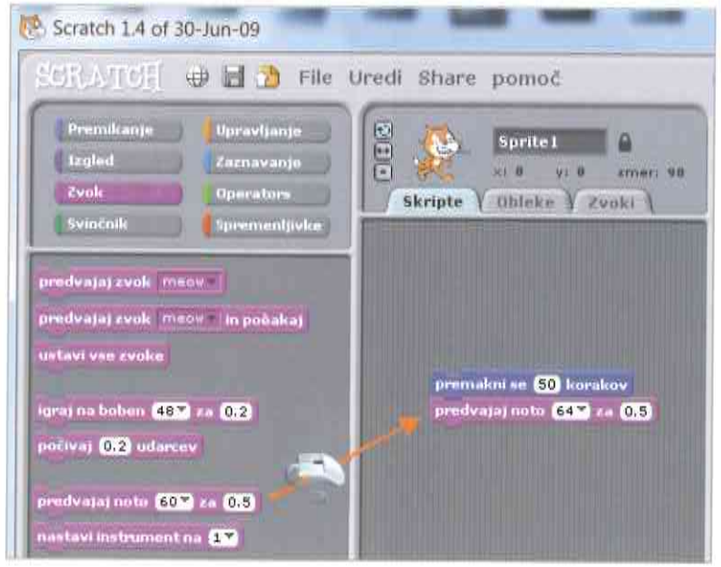
Ne pozabi, da sem glavna figura programa Scratch. V prvem projektu bova morala napisati navodila, kaj naj počnem na odru. Drugače povedano, napisati morava program za Praskača. A ne skrbi pomagal ti bom.

Potegni ukaz za premikanje v programsko okno.



Klikni v besedilno polje in napiši številko 50.  Nato klikni na ukaz in premaknil se bom za 50 korakov.

Da ukaze lažje najdeš, so razdeljeni v več skupin. Izberi skupino *Zvok* Zvok in poišči ukaz *predvajaj noto*. Povleci ga poleg prejšnjega ukaza in ukaza združi.



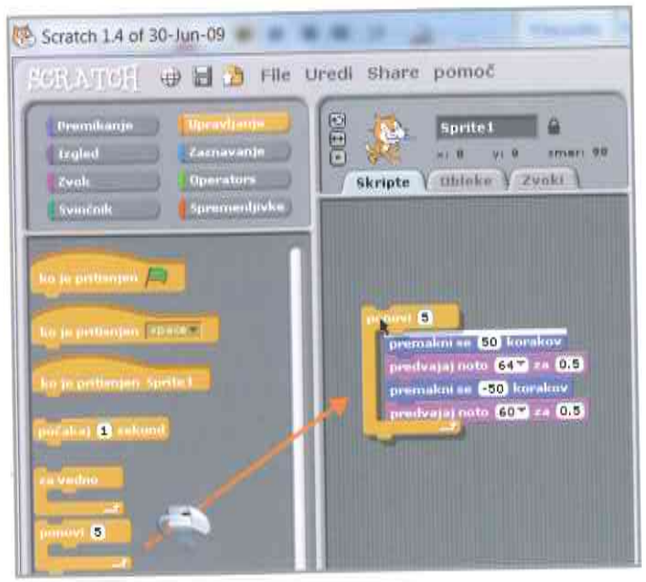
Noto izberi na izvlečnem meniju.

Dodaj še ukaz za premikanje in ukaz za predvajanje not. V ukazu za premikanje v besedilno polje napiši – 50, v ukazu za predvajanje not pa izberi noto C.

Klikni na enega izmed ukazov in poglej, kaj se zgodi.

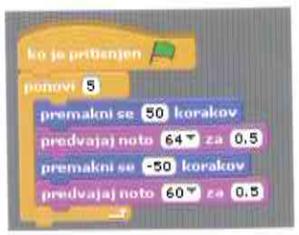
Združila sva več ukazov in nastal je **ukazni blok**. Če klikneš na ukazni blok, se ukazi, ki so v njem zapisani, izvršijo. Temu pravimo, da smo program zagnali. Ukazi v ukaznem bloku se izvajajo po določenem vrstnem redu, od zgoraj navzdol.

V programsko okno potegni ukaz *ponovi* iz skupine Upravljanje Upravljanje in ga združi z ostalimi ukazi. Klikni na ukazni blok.



Če sem zašel preblizu roba odra, me lahko s pomočjo kazalca miške potegneš nazaj. Če pa sem povsem izginil, z desno tipko miške klikni na mojo ikono v seznamu figur in izberi *pokaži*.

Ukazni blok lahko zaženeš tudi tako, da na vrh bloka dodaš ukaz ko je pritisnjen



in klikneš na zeleno zastavico na vrhu odra. Kako se ti zdi?

Kadar koli klikneš na zeleno zastavico, se zaženejo vsi ukazni bloki, ki imajo na vrhu ukaz ko je pritisnjen. Program ustaviš s klikom na gumb *stop*.

Tako, uspelo ti je. Zaplesal sem ob ritmu glasbe, naredila sva najin prvi projekt. Zdaj ga morava le še shraniti. V menijski vrstici izberi Datoteka (File).

Izberi možnost Shrani kot.

Določi mapo, kamor želiš shraniti svoj projekt, in napiši ime datoteke.

UKAZI

Ukazi so razdeljeni v osem skupin, ki združujejo vsebinsko podobne ukaze. Ukazi znotraj skupine so iste barve, zato ne bo težko najti posameznega ukaza.

Izbrana je skupina ukazov za premikanje.

Klikni na posamezno skupino in razišči, katere ukaze vsebuje. Kot vidiš, so ukazi različnih oblik:

- **Skladi**, kot je na primer premakni se 10 korakov, so najpogostejši ukazi. Skladi imajo vdolbino in izboklino. Take ukaze sestaviš v ukazne bloke, podobno kot drva v skladovnico.
- **Zanke**, kot je na primer ponovi 10, imajo obliko črke C. Zanke so narejene tako, da vanje vstaviš druge ukaze.
- **Pokrivala,** kot je na primer ko je pritisnjen zastavica, so na vrhu ukaznega bloka in pod določenim pogojem zaženejo ukaze pod njim.
- **Poročevalci,** kot sta na primer položaj x in velikost, so spremenljivke: hranijo podatke, ki se lahko spreminjajo. Vstavimo jih v druge ukaze.

. Spremenljivke z ovalnima robovoma, kot je na primer smer, imajo številčno vrednost. Spremenljivke s špičastima robovoma, kot je na primer miška pritisnjena?, povedo, ali je neka trditev pravilna ali napačna. Take spremenljivke imajo tako le dve vrednosti: prav ali narobe. Pravimo jim Boolove spremenljivke.

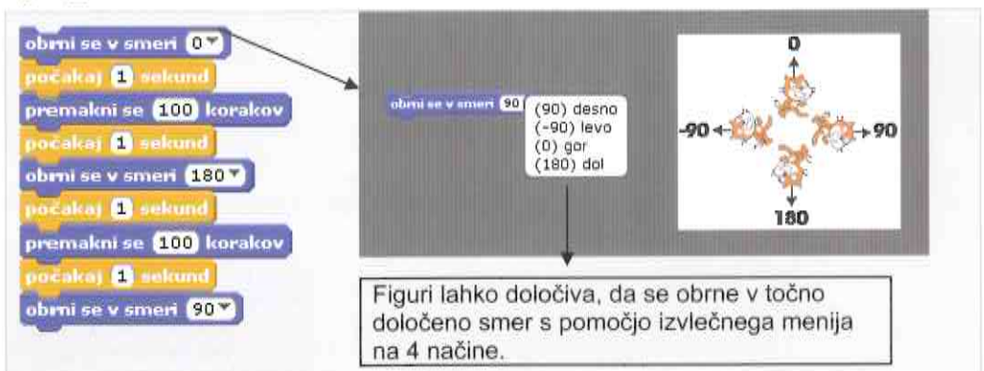
V nadaljevanju bova spoznala nekatere ukaze, s katerimi bova kasneje lahko programirala računalniške igre. Za vsako skupino so v tabelah prikazani vsi ukazi. Seveda ni treba, da že takoj na začetku natančno pregledaš vse ukaze. To lahko narediš tudi kasneje. Najbolj značilne ukaze bova spoznala tako, da bova primere, ki so že napisani, sestavila in preizkusila, kako delujejo. Začniva z ukazi za premikanje.

PREMIKANJE

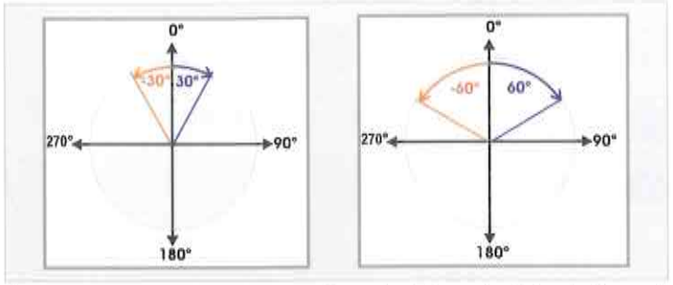
Figuro lahko premikaš na dva osnovna načina: lahko ji določiš, za koliko korakov naj se premakne v določeno smer, ali ji določiš mesto na odru, kamor naj se premakne.

Ukaz	Razlaga
premakni se 10 korakov	Figura se premakne za določeno število korakov.
obrni se 15 stopinj / obrni se 15 stopinj	Figura se obrne za določeno število stopinj.
obrni se v smeri 90 / obrni se proti Žoga	Figura se obrne v določeno smer (npr. smer 90 je gor) ali proti drugi figuri.
pojdi na x: 0 y: 0	Figura se na odru premakne na točno določeni koordinati x in y.
pojdi na kazalec miške	Figura se premakne k drugi figuri ali proti kazalcu miške.
drsi 1 sekund do x: 0 y: 0	Figura se v določnem času premakne na točno določeni koordinati na odru.
zamenjaj x za 10 / zamenjaj y za 10	Figuri spremeni lego na osi x oziroma y za določeno vrednost.
nastavi x na 0 / nastavi y na 0	Figura se na odru premakne na točno določeno koordinato x oziroma y.
odbij se če si na robu	Ko figura pride do roba odra, se odbije.
položaj x / položaj y / smer	Spremenljivke povedo, kakšen je položaj ali smer figure.

V naslednjem primeru bova določila smer mojega gibanja in število korakov, ki jih bom naredil. Sestavi ukazni blok in opazuj, kako se premikam. Če še vedno stojim pri miru, klikni na ukazni blok.



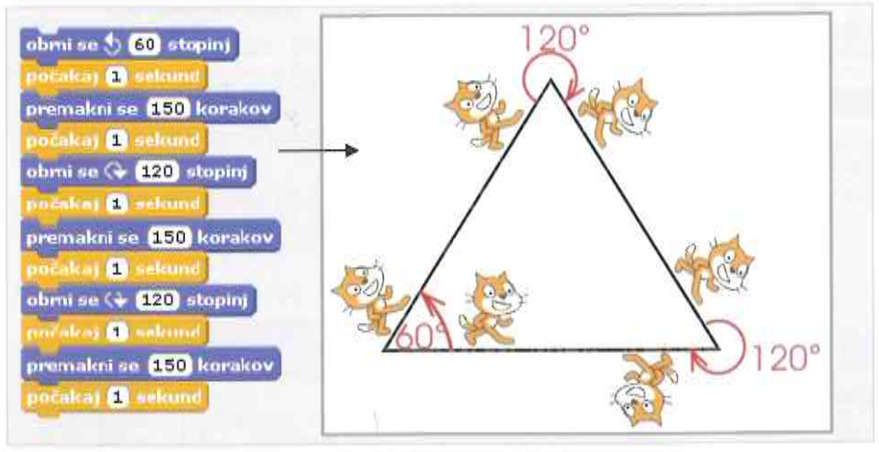
Če želiš, da se obrnem v ustrezno smer, lahko uporabiš ukaz *obrni se v smeri* in določiš smer glede na strani neba. Lahko pa mi smer določiš tudi s pomočjo kota zasuka. Če želiš, da se obrnem za četrtino kroga, se moram zasukati za 90°. Polovica kroga predstavlja 180°, šestina kroga 60° in dvanajstina kroga 30°.



Če kotov še ne poznaš, naredi naslednjo vajo. Izberi ukaz *obrni se 90 stopinj*

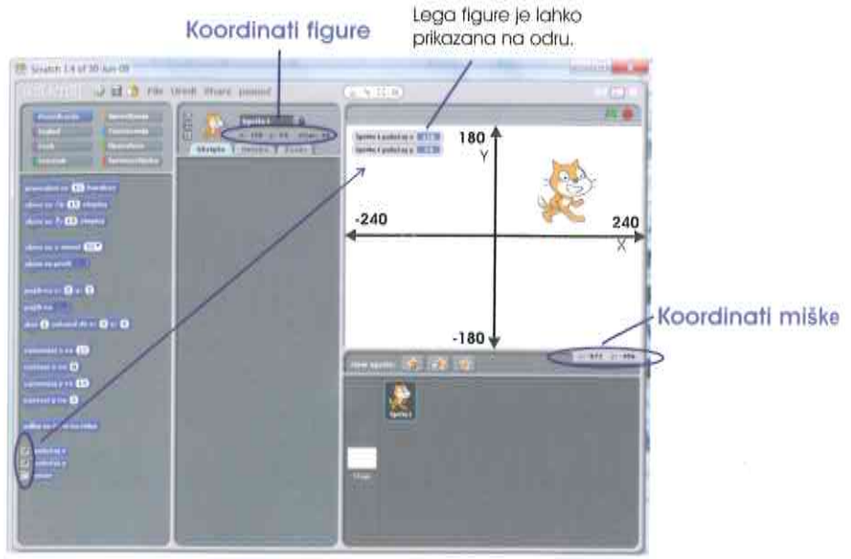
in štirikrat zapored klikni nanj. Enako naredi še s 60 in 30 stopinjami. Pomisli, kolikokrat moraš klikniti na posamezni ukaz, da bom naredil cel obrat.

Zdaj, ko že bolje poznaš kote, lahko sestaviš ukazni blok tako, da se bom premikal po stranicah enakostraničnega trikotnika.

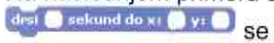


Kot vidiš, sem obrnjen na glavo. Klikni na ukaz *obrni se v smeri 90* in me postavi na noge. Pomisli, kako naj se obračam po stranicah trikotnika, da na koncu ne bom obrnjen na glavo. Ni težko, le po zadnji stranici bom moral hoditi ritensko.

Drugi način premikanja je, da figuri določiva položaj na odru s pomočjo koordinatnega sistema. Na odru se vodoravna os X razteza od – 240 enot do 240 enot, navpična os Y pa od -180 enot do 180 enot. Vsaka enota predstavlja en korak. Na sredini odra sta koordinati x in y enaki 0.



Na naslednjem primeru si poglejva še ta način premikanja. S pomočjo ukaza

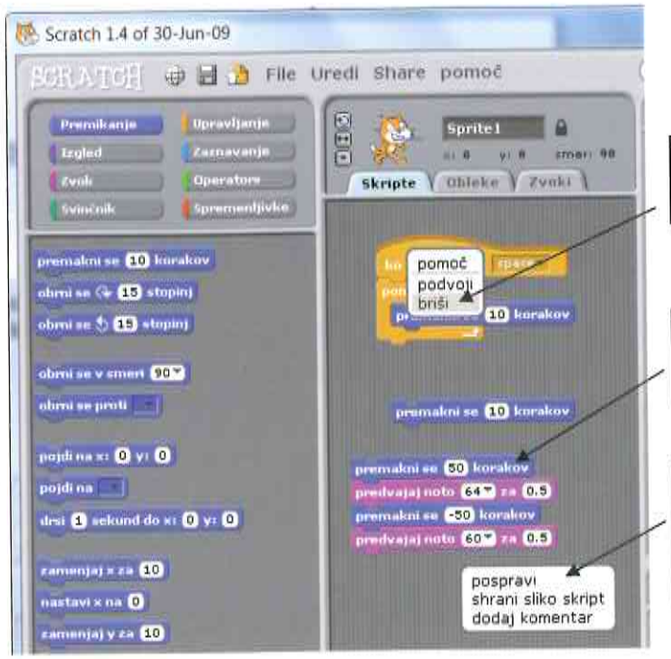


se bom premaknil v vsako četrtino odra, nato pa se bom vrnil v sredino.

Preizkusi še naslednji primer, kjer bova spoznala zelo uporaben ukaz

.

V programskem oknu se je nakopičilo kar nekaj ukaznih blokov, ki jih v nadaljevanju ne bova potrebovala. Ukazne bloke zato lahko zbrišeš, lahko pa jih tudi pospraviš.



Izberi ukazni blok, klikni na desno tipko miške in v pogovornem oknu izberi *briši*.

Ukazni blok lahko zbrišeš tudi tako, da ga potegneš v levo okno.

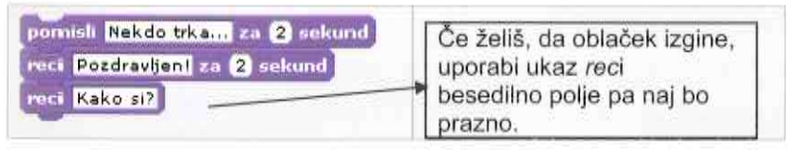
Če v programskem oknu klikneš z desno tipko v prazno, se odpre pogovorno okno. Ugotovi, kako uporabno je.

IZGLED

Ukazi v skupini *Izgled* lahko spremenijo videz figure ali ozadja.

Ukaz	Razlaga
preklopi na videz obleka2	Figura spremeni videz tako, da zamenja obleko.
naslednji videz	Figura zamenja obleko tako, da izbere naslednjo obleko s seznama.
maska #	Sporoči številko pred obleko.
preklopi na ozadje ozadje1	Zamenja sliko ozadja.
reci Pozdravljen(a)! za 2 sekund	Figura za določen čas v oblačku prikaže napisano besedilo.
pomisli Hmm... za 2 sekund	Figura za določen čas v oblačku za razmišljanje prikaže napisano besedilo.
zamenjaj učinek barva za 25	Figura s pomočjo raznih učinkov (barva, ribje oko, ...) spremeni videz za stopnjo, ki je določena s številko.
izbrisi grafične učinke	Grafično podobo figure vrne v prvotno stanje.
nastavi velikost na 100 %	Nastavi velikost figure na določeno vrednost.
velikost	Sporoči velikost figure v %.
pokaži	Figura se pokaže na odru.
skrij	Figura se skrije.
pojdi nazaj 1 plasti	Figura se premakne za eno plast nazaj.
pojdi v ospredje	Figura se premakne pred vse ostale figure.

Figuro lahko povečaš, pomanjšaš, ji spremeniš barvo ali pa ji menjaš obleke. Figura lahko tudi pove, kaj misli. Kako se to naredi, bo pokazal naslednji ukazni blok.



Poglejva, kako me lahko oblikuješ. Spremeni mi obliko, barvo in velikost.

Kako pa se preoblačim? Klikni na jeziček *Obleke* in pokazalo se bo, da imam dve različni obleki.

Jeziček *Obleke*

Običajno ima vsaka figura na začetku le eno obleko, ostale pa moramo narediti sami. Kako se to naredi, bova pogledala na drugi, novi figuri. Uvozi figuro dog2-a iz mape Živali (Animals).

Klikni na gumb *Izberi novi sprite iz datoteke.*

Sprite2 preimenuj v Kuža. Imena figur bova pisala z veliko začetnico.

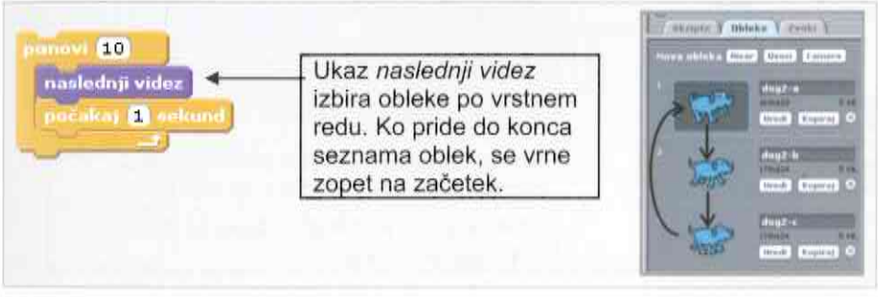
Izbriši ime sprite in vpiši novo ime.

Zdaj mene, Praskača, lahko izbrišeš.

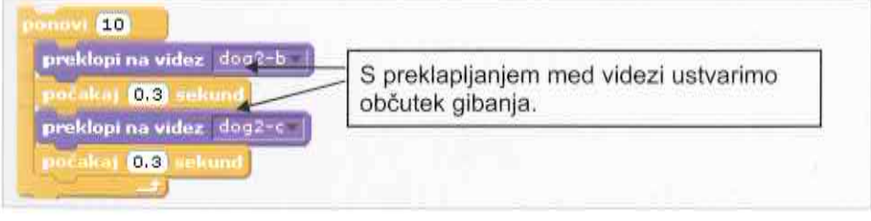
Izberi Praskača, klikni na desno tipko miške in v pogovornem oknu izberi *briši.*

Izberi jeziček *Obleke*. Kot vidiš, ima Kuža le eno obleko. Naredila mu bova še dve. Klikni na gumb *Uvozi* in v mapi Živali (Animals) najprej izberi obleko dog2-b in nato še obleko dog2-c. Kuža ima sedaj tri različne obleke.

Izberi jeziček *Skripte*. Sestavi naslednje zaporedje ukazov in opazuj, po kakšnem vrstnem redu se spreminja videz psa.



Kaj pa, če izbiraš le med dvema določenima oblekama?



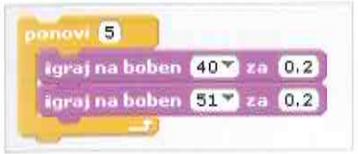
ZVOK

Vsaka prava računalniška igra ima tudi zvočne učinke. Nič drugače ne bo v igrah, ki jih bova naredila v Scratchu. Veliko zvokov ima Scratch že pripravljenih. Uporabiva jih s pomočjo skupine ukazov *Zvok*.

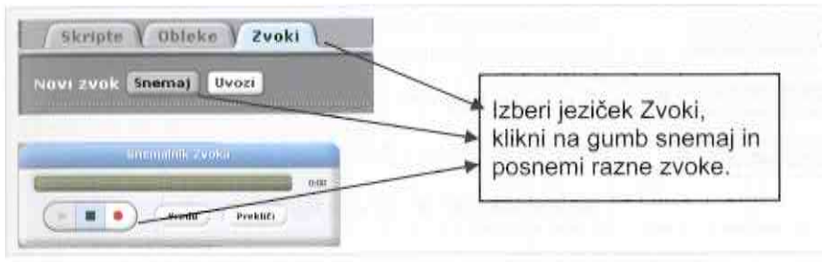
Ukaz	Razlaga
predvajaj zvok meow	Začne predvajati zvok, ki je izbran v meniju, nato gre takoj k naslednjemu ukazu.
predvajaj zvok meow in počakaj	Predvaja zvok toliko časa, dokler ni končan.
ustavi vse zvoke	Vsi zvoki se prenehajo izvajati.
igraj na boben 48 za 0.2	Določen čas igra na izbrano tolkalo.
počivaj 0.2 udarcev	Naredi pavzo.
predvajaj noto 60 za 0.5	Določen čas predvaja izbrani ton.
nastavi instrument na 1	V meniju izbereš inštrument.
spremeni nivo zvoka za -10	Zvok predvaja glasneje ali tišje.
nastavi glasnost na 100 %	Določi jakost zvoka.
glasnost	Pove, kako glasen je zvok.
spremeni tempo za 20	Spremeni hitrost izvajanja.
nastavi tempo na 60 bpm	Nastavi predvajanje zvoka na določeno hitrost. Določeni so udarci na minuto.
tempo	Pove, kakšna je hitrost (udarci na min.) predvajanja zvoka.

Če se ti zdi, da je v tvojem grafičnem vmesniku že prava zmešnjava, bo najbolje, da odpreš nov projekt. V menijski vrstici izberi *Datoteka* (*File*) in nato *Nova*. Lahko pa tudi ponovno zaženeš Scratch.

Sestavi naslednje ukazne bloke in prisluhni.



Če ti ne zadostujejo vgrajeni zvoki, lahko zvoke tudi posnameš ali poiščeš na internetu. Če želiš zvok posneti, mora imeti tvoj računalnik mikrofon.

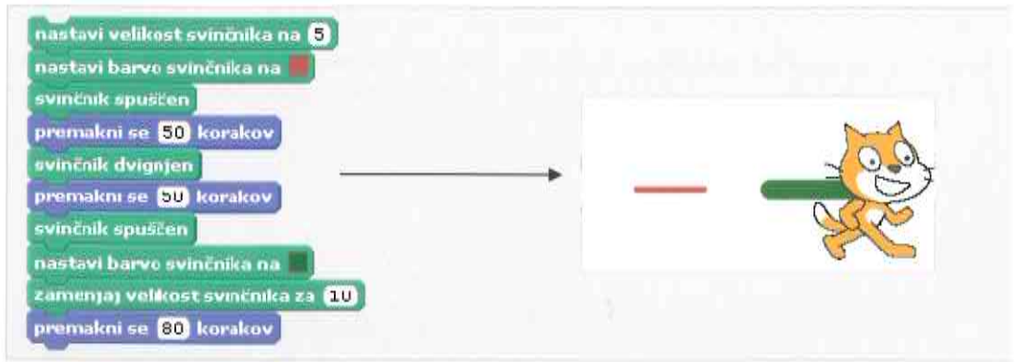


SVINČNIK

Ukazi iz skupine *Svinčnik* omogočajo risanje in na odru nastajajo različne kombinacije raznobarvnih črt in podob. Seveda pa ne bova rekla, da riševa na oder, ampak na platno.

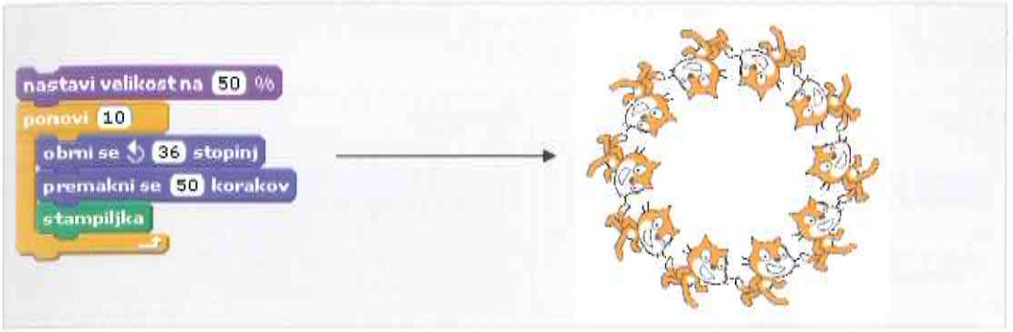
Ukaz	Razlaga
izbriši	Z odra izbriše vse sledi.
svinčnik spuščen	Figura spusti svinčnik in za sabo pušča sled.
svinčnik dvignjen	Figura dvigne svinčnik.
nastavi barvo svinčnika na	Določi barvo svinčnika s pomočjo kapalke.
zamenjaj barvo svinčnika za 10	Spremeni barvo svinčnika za določeno število. Barve so določene s številkami.
nastavi barvo svinčnika na 0	Nastavi barvo svinčnika. Barve so določene s številkami.
zamenjaj senco svinčnika za 10	Spremeni intenzivnost barve.
nastavi senco svinčnika na 50	Nastavi intenzivnost barve. Intenzivnost barve je določena s številko.
zamenjaj velikost svinčnika za 1	Spremeni debelino svinčnika za določeno debelino.
nastavi velikost svinčnika na 1	Nastavi debelino svinčnika na določeno debelino.
stampiljka	Na mestu, kjer se nahaja figura, naredi svoj odtis.

Ker se premikam zelo hitro, mi je precej težko slediti. Zato naju bo zanimalo, ali so navodila za gibanje, ki sva jih dala meni ali kateri drugi figuri, res takšna, kot sva želela. To lahko preveriva tako, da uporabiva ukaze iz skupine *Svinčnik*. Če uporabiva ukaz *svinčnik spuščen*, bom za sabo puščal sled.



V naslednjem ukaznem bloku bova uporabila ukaz *štampiljka*, tako da bo nastala simetrična slika. To, kar bova videla na odru, pa ne bodo nove figure, ampak le njihovi odtisi na platnu.

Če je platno popisano, ga najprej počistiva z ukazom *izbriši* izbriši .



ZAZNAVANJE

Ukazi iz skupine *Zaznavanje* so nujno potrebni za izdelavo dobre računalniške igre. S pomočjo teh ukazov ter s tipkami na tipkovnici ali z miško se lahko igralec pogovarja z junaki igre in jim daje navodila. Ukazi zaznavanja so pomembni tudi zato, da zaznajo dogajanje na odru, na primer ali se dve figuri dotikata.

Ukaz	Razlaga
se dotika kazalec miške ?	Spremenljivka pove, ali se figura dotika kazalca miške (roba zaslona ali druge figure).
se dotika barve ?	Spremenljivka pove, ali se figura dotika določene barve.
barva je več kot ?	Spremenljivka pove, ali se barva prve figure dotika barve druge figure.
miska y miska x	Spremenljivka pove lego figure na osi x oziroma na osi y.
miska pritisnjena?	Spremenljivka pove, ali je leva tipka miške pritisnjena.
ali je gumb puščica desno pritisnjen?	Spremenljivka pove, ali je pritisnjena desna puščica (ali katerakoli druga tipka) na tipkovnici.
razdalja do Žoga	Spremenljivka pove, kakšna je razdalja od prve do druge figure (npr. Žoge) ali do kazalca miške.
štoparica	Spremenljivka sporoči čas štoparice v sekundah.
ponastavi štoparico	Štoparico nastavi na nič.
položaj x od Sprite1	Spremenljivka pove, kakšna je vrednost koordinate x ali y izbrane figure.

Najprej preizkusiva, kaj vse lahko doseževa s premikanjem miške, če uporabiva različne ukaze zaznavanja. Napiši naslednji ukazni blok in opazuj, kako gibanje miške gor ali dol vpliva na moj videz. Ne pozabi klikniti na ukazni blok.

Kaj pa premikanje miške levo in desno? To si oglej v naslednjem bloku.

Dogodke lahko sprožiš s pritiskom na tipke na tipkovnici. Poglejva, kako se bom premikal, če pritisneš na tipko a.

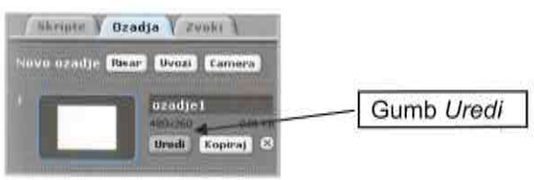
Če ima tvoj računalnik mikrofon, lahko poskusiš še, kaj povzroči naslednji ukazni blok.

Če se ni zgodilo nič, moraš zavpiti glasneje.

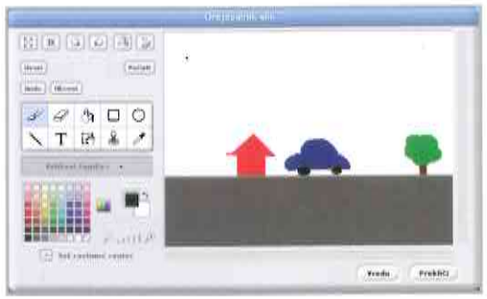
Figure lahko s pomočjo ukazov zaznavanja prepoznavajo tudi različne barve na odru. Za spoznavanje teh ukazov moraš najprej narisati barvito ozadje. Ozadje je nov pojem, ki ga doslej še nisva spoznala. V Scratchu ima podobno vlogo, kot jo imajo v filmih kulise. Lahko spreminja videz, ne more pa se premikati.

Klikni na ikono *Ozadje*, ki jo najdeš poleg seznama figur.

Izberi jeziček *Ozadja* in klikni na gumb *Uredi*.



Odpre se pogovorno okno, ki te verjetno spominja na program Slikar. Če poznaš Slikarja, ti ne bo težko narisati ozadja, na katerem bodo rdeča hiša, moder avto in drevo.



Za prepoznavanje barve bova uporabila ukaz 

. Želiva, da bom spreminjal videz, ko se bom dotaknil rdeče barve hiške, modre avta in zelene drevesa. Barvo določiva s pomočjo kapalke.

V oknu Seznam figur izberi ikono Praskača. Postavi me na začetek odra, napiši ukazna bloka, klikni na zeleno zastavico in opazuj, kaj se z mano dogaja.

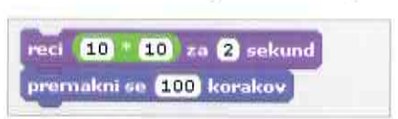
OPERATORJI

Pri računalniškem programiranju so, kot bova kmalu spoznala, poleg računskih operacij zelo pomembne tudi logične operacije, kot so na primer logični IN, logični ALI in logični NE.

Ukaz	Razlaga
	Sešteje, odšteje, zmnoži ali deli dve števili.
	Izbere naključno celo število med določenima mejama.
	Spremenljivka pove, ali je prva vrednost manjša, enaka oziroma večja od druge vrednosti.
	Spremenljivka pove, ali sta obe vstavljeni trditvi pravilni.
	Spremenljivka pove, ali je vsej ena od vstavljenih trditev pravilna.
	Spremenljivka pove, ali je vstavljena trditev napačna.
	Združi besede.
	Črka na določenem mestu.
	Dolžina besede.
	Ostanek pri deljenju.
	Zaokroži na najbližje celo število.
	Kvadratni koren določenega števila.

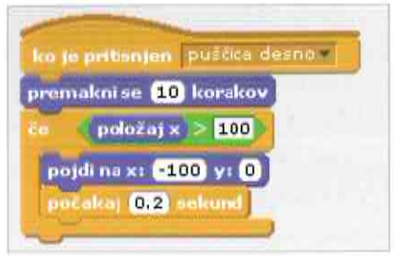
Kako uporabiti posamezne operatorje, bova spoznala pri naslednjih primerih.

To ve vsak, 10*10 je 100. Sem pameten, kaj?



Katera števila lahko padejo, ko mečeš kocko?

Pritiskaj na puščico desno. Premikal se bom v desno, a do kod? Kaj me ustavi?



Sestavi spodnji ukazni blok in ga zaženi.

Prišel sem do roba, zdaj bom skočil v sredino odra, vendar mora biti izpolnjen še en pogoj. Kateri?

In še malo poklepetajva …

V besedilno polje vtipkaj svoje ime in pritisni tipko enter.

UPRAVLJANJE

Ukazi upravljanja omogočajo, da se ukazni bloki zaženejo ob pravem času oziroma, da jih lahko sprožijo različni dogodki, na primer pritisk na določeno tipko na tipkovnici. Omogočajo tudi, da se določeno zaporedje ukazov večkrat ponovi. Takim ukazom rečemo **zanke**.

Ukaz	Razlaga
ko je pritisnjen	Ko pritisnemo na zeleno zastavico, se zaženejo spodaj pripeti ukazi.
ko je pritisnjen space	Kadar je pritisnjena izbrana tipka na tipkovnici, se zaženejo vsi spodaj pripeti ukazi.
ko je pritisnjen Sprite1	Kadar je pritisnjena izbrana figura, se zaženejo vsi spodaj pripeti ukazi.
počakaj 1 sekund	Počaka določen čas, nato nadaljuje z izvajanjem naslednjega ukaza.
za vedno	Znova in znova ponavlja izvajanje ukazov, ki so znotraj zanke.
ponovi 10	Ukazi znotraj zanke se ponovijo za določen večkratnik.
predvajaj	Pošlje sporočilo vsem figuram.
predvajaj in počakaj	Pošlje sporočilo vsem figuram in počaka, da se izvedejo tisti ukazni bloki, ki so sporočilo prejeli.
za vedno če	Zanka ves čas preverja, ali je določeni pogoj izpolnjen. V tem primeru se izvedejo ukazi znotraj zanke.
če	Če je pogoj izpolnjen, se izvedejo ukazi znotraj zanke.
če drugače pa	Če je pogoj izpolnjen, se izvedejo ukazi znotraj zanke *če*, sicer pa znotraj zanke *drugače pa*.
počakaj dokler	Čaka, dokler pogoj ni izpolnjen, nato se začnejo izvajati spodaj pripeti ukazi.

Ukaz	Razlaga
	Ukazi znotraj zanke se izvajajo, dokler pogoj ni izpolnjen.
	Konča izvajanje ukaznega bloka.
	Ustavi vse programe pri vseh figurah.

Kot veš, se izvajanje ukazov začne tudi tako, da klikneš na zeleno zastavico  .

V tem primeru morajo imeti ukazni bloki na vrhu ukaz  . Če je takih ukaznih blokov v programu več, se bodo začeli izvajati vsi naenkrat. Sestavi spodnja ukazna bloka, klikni na zeleno zastavico in razmisli, zakaj stojim na mestu.

Čeprav sta bloka postavljena drug pod drugim, se oba začneta izvajati hkrati. Prvi me premika v desno, drugi pa me hkrati vleče v levo. Zato mirujem. Če želiš, da se bom najprej za 200 (10*20) korakov premaknil v desno, nato pa za enako število korakov v levo, bova morala narediti drugače.

Ker veš, da se ukazi v enem ukaznem bloku izvajajo drug za drugim, ne bo težko najti prave rešitve.

Ker sva si z ukazi upravljanja pomagala pri spoznavanju ukazov iz drugih skupin, si oglejva le še en primer. Sestavi naslednji ukazni blok, zaženi ga, preštej do tri in poglej, kaj se mi zgodi.

SPREMENLJIVKE

Zadnja skupina ukazov je skupina *Spremenljivke*. Preden lahko vidiš, kateri ukazi so v tej skupini, je treba ustvariti novo spremenljivko in nov seznam.

Naredila bova novo spremenljivko in jo poimenovala Rezultat ter nov seznam z imenom Avtomobili.

Poglejva, kateri ukazi so se prikazali.

Ukaz	Razlaga
Rezultat	Vrednost spremenljivke rezultat.
nastavi Rezultat na 0	Nastavi vrednost spremenljivke na določeno število.
zamenjaj Rezultat za 1	Spremeni vrednost spremenljivke za določeno število.
pokaži spremenljivko Rezultat	Vrednost spremenljivke se vidi na odru.
skrij spremenljivko Rezultat	Vrednost spremenljivke na odru se skrije.
Avtomobili	Pove vsa polja v seznamu.
prištej Ferrari k Avtomobili	Doda polje k seznamu.
briši 1 od Avtomobili	Zbriše določeno polje s seznama.
vstavi Golf pri 2 od Avtomobili	Vstavi določeno polje na določeno mesto v seznamu.
zamenjaj predmet 1 od Avtomobili z thing	Zamenja dve polji.
predmet 1 od Avtomobili	Spremenljivka pove vrednost določenega polja.
dolžina od Avtomobili	Spremenljivka pove, koliko polj ima seznam.
Avtomobili contains Audi	Spremenljivka pove, ali seznam vsebuje določeno vrednost.

Več o spremenljivkah boš izvedel v naslednjih poglavjih, sezname pa sva le omenila.

Tako, Scratch zdaj poznaš. Saj ni bilo pretežko, kajne? Zdaj naju čakajo nove dogodivščine, učenje programiranja in ustvarjanje lastnih iger. Prepričan sem, da bo zanimivo.

PROGRAMIRAJMO

Ta del knjige je namenjen sistematičnemu učenju osnov programiranja. Tu lahko podrobneje spoznaš pomembne koncepte programiranja, kot so zanke, pogojni stavki, spremenljivke, naključna števila, Boolova logika. Vse to ti bo omogočilo samostojno reševanje novih, drugačnih problemov.

Del Programirajmo morda še najbolj spominja na klasične računalniške zbirke nalog. Na začetku vsakega poglavja je snov razložena s pomočjo primerov, sledi pa samostojno reševanje nalog in seveda rešitve. Naloge so zastavljene tako, da jih lahko rešiš z znanjem, pridobljenim pri prejšnjih nalogah ali primerih.

Če je naloga zahtevnejša, so zraven tudi napotki za reševanje. Treba je poudariti, da je rešitev naloge, ki je v knjigi, le ena od možnih. Povsem mogoče je, da bo tebi uspela drugačna, lahko tudi boljša rešitev. Dobro je, da svojo rešitev primerjaš s tisto v knjigi, še posebej takrat, ko so poleg rešitve napisane opombe. Tako lahko spoznaš nove prijeme in jih uporabiš pri naslednjih nalogah.

Del Programirajmo je razdeljen na dve poglavji, Svinčnik in Štampiljka, glede na to, kateri ukaz uporabljaš za risanje vzorcev. Ukaza poznaš že iz uvoda, njuna uporabnost pa se bo pokazala pri reševanju nalog. Vsako od poglavij je razdeljeno na podpoglavja, ki obravnavajo določen programski koncept. Včasih se nek koncept pojavi vzporedno pri poglavju Štampiljka in Svinčnik. Morda bi bilo bolj smiselno, da bi naloge, ki obravnavajo isti koncept, združila v isto poglavje, vendar bi se tako zaradi neprestanega preskakovanja iz ukaza *svinčnik* na ukaz *štampiljka* izgubila rdeča nit.

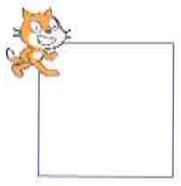
Zdaj pa vklopi možgane »na ful« in veselo na delo! ☺

SVINČNIK

Kot veš, lahko figura za sabo pušča sled, črto, če uporabiš ukaz *svinčnik spuščen*. Zaženi program Scratch in iz skupine *svinčnik* izberi ukaz svinčnik spuščen. Začniva s preprostim primerom.

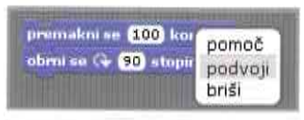
PREMAKNI SE, POJDI NA

1. primer: Narisala bova kvadrat z velikostjo stranice 100 enot, kar pomeni sto korakov.



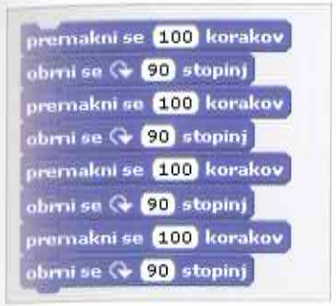
Najprej nariši 100 korakov in se obrni za 90°.

Nastala je prva stranica kvadrata. Ker za naslednjo stranico potrebuješ enaka ukaza, lahko izvedeš preprost trik. Postavi se na ukazni blok, klikni z desno tipko miške in v pogovornem oknu, ki se odpre, izberi *podvoji*.



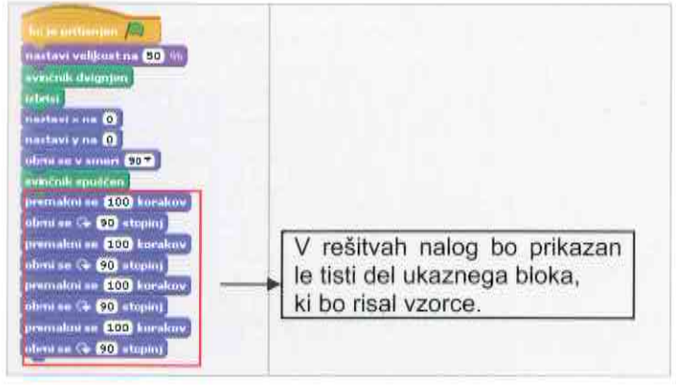
Ukazni blok se podvoji in lahko ga priključiš prejšnjemu ukaznemu bloku.

Še enkrat podvoji in narejen bo ukazni blok, ki nariše kvadrat.

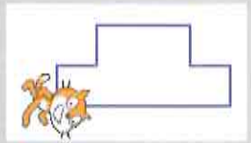


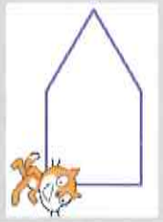
Ko se bova lotila nove naloge, bova verjetno hotela, da bo platno spet čisto, jaz, Praskač, pa na svojem prvotnem mestu. Zato napišiva program, ki s klikom na zeleno zastavico počisti platno, mene pa postavi na sredino odra in me usmeri proti desni. Da ne bom prekrival prevelikega dela slike, me lahko pomanjšaš na polovico.

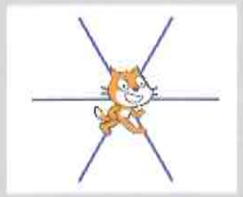
Program shrani in ga poimenuj Svinčnik. Ukazni blok, ki počisti platno, bova lahko uporabila pri nalogah, ki bodo sledile, saj bo služil kot nekakšna radirka. Če bo naloga zahtevala, da narišeš kvadrat, bova ukaznemu bloku, ki riše kvadrat, pripela še ukazni blok, ki počisti ekran.



1. naloga: Odpri datoteko Svinčnik in jo shrani z imenom Avto. Nariši avto, hiško in snežinko. Izberi poljubno velikost stranic.







Rešitev:

premakni se 130 korakov
obrni se 90 stopinj
premakni se 30 korakov
obrni se 90 stopinj
premakni se 30 korakov
obrni se 90 stopinj
premakni se 30 korakov
obrni se 90 stopinj
premakni se 70 korakov
obrni se 90 stopinj
premakni se 30 korakov
obrni se 90 stopinj
premakni se 30 korakov

Rešitev:

premakni se 70 korakov
obrni se 90 stopinj
premakni se 70 korakov
obrni se 30 stopinj
premakni se 70 korakov
obrni se 120 stopinj
premakni se 70 korakov
obrni se 30 stopinj
premakni se 70 korakov

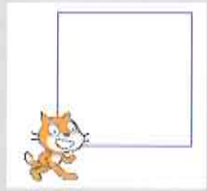
Rešitev:

premakni se 70 korakov
obrni se 180 stopinj
premakni se 70 korakov
obrni se 180 stopinj
obrni se 60 stopinj

.
.
.

Ukazni blok se šestkrat ponovi.

2. naloga: Odpri program Svinčnik in ga shrani z novim imenom Kvadrat.
Spet napiši program, ki nariše kvadrat s stranico 100 korakov. Tokrat le z uporabo ukaza pojdi na x: y: . Razmisli, kako nastaviti koordinati za posamezno oglišče.

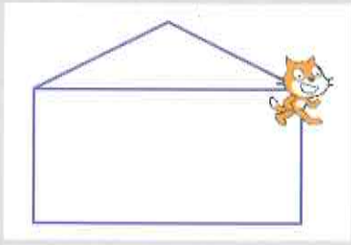


Rešitev:

pojdi na x: 100 y: 0
pojdi na x: 100 y: 100
pojdi na x: 0 y: 100
pojdi na x: 0 y: 0

3. naloga: Odpri datoteko Svinčnik in jo shrani z imenom Pismo. Nariši pismo z eno potezo. Zopet uporabi le ukaz pojdi na x: y: .

Pomoč: Če ne maraš računanja z negativnimi števili, začni risati na x: 0 y: 100.



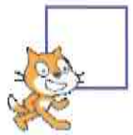
Rešitev:

pojdi na x: 0 y: 100
svinčnik spuščen
pojdi na x: 200 y: 100
pojdi na x: 200 y: 0
pojdi na x: 0 y: 0
pojdi na x: 0 y: 100
pojdi na x: 100 y: 150
pojdi na x: 200 y: 100

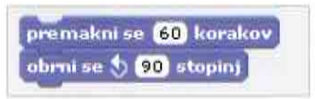
PONOVI

Kadar se neko zaporedje ukazov ponavlja, lahko uporabimo ukaz *ponovi*. Kako to deluje, lahko vidiš v 2. primeru.

2. primer: Narišiva kvadrat s pomočjo ukaza *ponovi*. Stranica kvadrata naj bo velika 60 korakov.



Kvadrat sva v 1. primeru narisala tako, da sva štirikrat ponovila zaporedje ukazov:



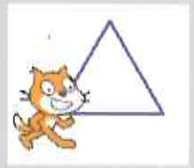
Z ukazom *ponovi* bo ukazni blok, ki nariše kvadrat, veliko krajši.

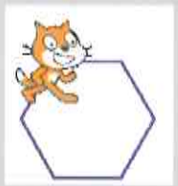
Zaporedje ukazov se ponavlja, kot da so ujeti v zanko. Zato bova ukaze, kjer se neko zaporedje večkrat ponavlja, poimenovala **zanke**.

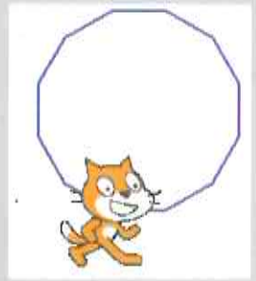
4. naloga: Odpri program Svinčnik in ga shrani kot Večkotniki.
Nariši trikotnik, šestkotnik in dvanajstkotnik. Smiselno izberi velikost stranice in izračunaj kot obrata.

Pomoč: Vsakič, ko narišem večkotnik, se obrnem za en krog, to je za 360°.

- Pri kvadratu se obračam 4-krat, vsakič za 360°/4, kar je za 90°.
- Pri šestkotniku se obračam 6-krat, vsakič za 360°/6, kar je za 60°.
- Pri dvanajstkotniku se obračam 12-krat, vsakič za 360°/12, kar je za 30°.



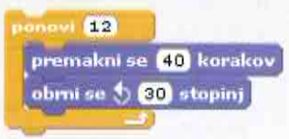




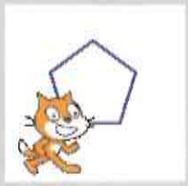
Rešitev:

Rešitev:

Rešitev:



5. naloga: Odpri program Večkotniki in ga shrani kot Večkotniki 1. Nariši petkotnik ter 120-kotnik. Velikosti kota obračanja ni potrebno izračunati, saj lahko uporabiš operator deljenja  .

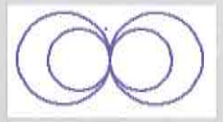


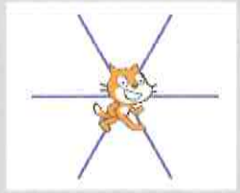
Rešitev:

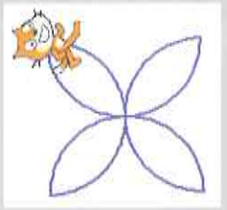
Rešitev:

Opomba: Opaziva, da je 120-kotnik na videz že krog. Kot vidiš, lahko kroge predstaviva z večkotniki z dovolj velikim številom stranic. Za manjše kroge bova potrebovala manjše število stranic kot za večje kroge. Poskusi narisati nekaj različno velikih krogov.

6. naloga: Odpri program Večkotniki in ga shrani kot Mušje oko. S pomočjo ukaza *ponovi* nariši še mušje oči, zvezdo in cvet.







Rešitev:

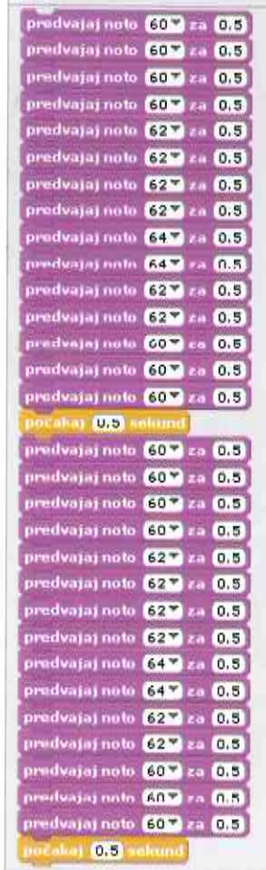
Rešitev:

Rešitev:

GNEZDENJE

3. primer: Kaj pomeni gnezdenje ukazov, je zelo razvidno na primeru pesmice Kuža pazi. Oglej si zaporedje ukazov na levi strani tabele in opazuj ponavljajoče se vzorce in ukaze.

Pesmica Kuža pazi	V programski zaslon povleci spodnje ukaze in jih sestavi v pesmico Kuža pazi.

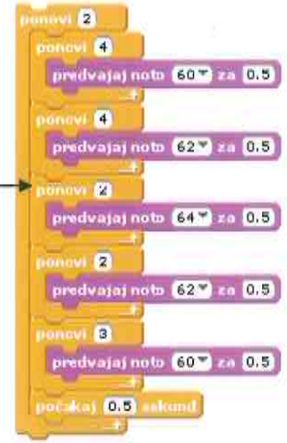


Če ti je uspelo z ukazi *ponovi* ustvariti melodijo pesmice Kuža pazi, lahko spodnjo razlago kar preskočiš.

Kitica se dvakrat ponovi.

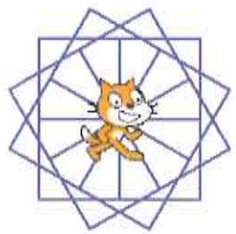
Ton C se ponovi štirikrat.

Končna rešitev.

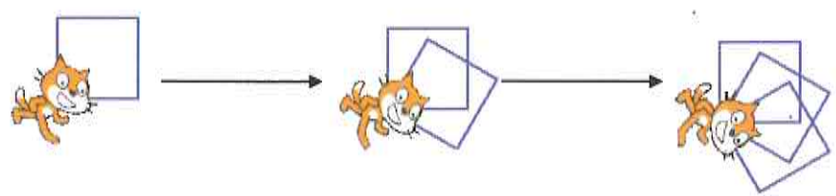


Spoznala sva, da lahko zanko *ponovi* vstaviva v drugo zanko *ponovi*. Vstavljanju zanke v novo zanko rečemo **gnezdenje**.

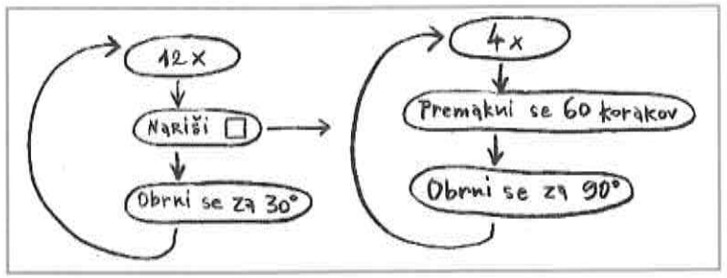
4. primer: Razmisliva, kako bi gnezdenje uporabila, da bi dobila vzorec na spodnji sliki.



Če natančno pogledaš, opaziš, da je osnovni motiv vzorca kvadrat. Kvadrat je narisan 12-krat in je vsakič zasukan za 30°.

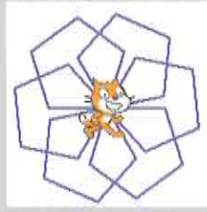


Ko rišem, 12-krat izvedem zunanjo zanko. Ko zanko izvedem prvič, narišem kvadrat in se obrnem za 30°. Drugi kvadrat je tako že narisan z zamikom 30°. Tretji kvadrat je narisan z zamikom 60° itd. Pomagajva si z miselnim vzorcem.



Povzemiva: zunanja zanka dvanajstkrat nariše kvadrat ter naredi zasuk. Notranja zanka pa riše kvadrat. Če ti ni uspelo napisati programa, si poglej spodnjo rešitev.

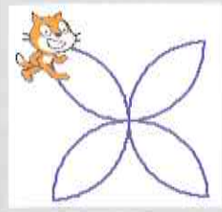
7. naloga: Odpri program Svinčnik in ga shrani kot Vzmet. Uporabi gnezdenje in nariši cvet ter vzmet.



Rešitev:

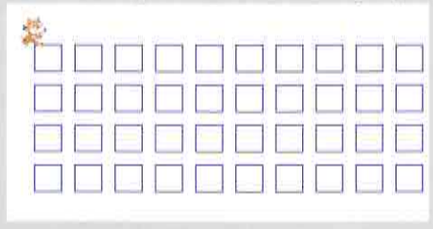
Rešitev:

8. naloga: Odpri program Mušje oko in ga shrani kot Cvet. Nariši cvet. Tokrat uporabi gnezdenje.



Rešitev:

9. naloga: Nariši polje kvadratov. Ker so kvadrati majhne velikosti, me pomanjšaj na 20 % moje velikosti. Tako lažje spremljaš, kako rišem.



Pomoč:

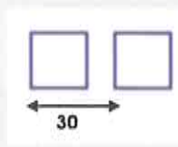
Pri težji nalogi je problem smiselno razgraditi na manjše enote in nalogo reševati po korakih. V tem primeru lahko nalogo rešiš v treh korakih:

a) narišeš motiv, ki je kvadrat;
b) narediš eno vrstico kvadratov;
c) narediš 4 vrstice kvadratov.

a) Motiv

Rešitev:

Opomba: Pozorno opazuj, kje se nahajam, ko narišem prvi kvadrat. Naslednji kvadrat je 10 korakov oddaljen od prvega. Ker končam risanje kvadrata tam, kjer sem ga začel, to je v levem spodnjem kotu, moram za risanje naslednjega kvadrata preskočiti celo stranico kvadrata in razdaljo med kvadratoma, kar je skupaj 30 korakov. Pri tem uporabim ukaz *svinčnik dvignjen*.



b) Vrstica

Rešitev:

Opomba: Vrstico narišem tako, da desetkrat narišem kvadrat, med risanjem vsakega pa naredim 30 korakov. Pomembno je, kje zaključim z risanjem vrstice. Pri risanju naslednje vrstice se moram postaviti na začetek vrste, 30 korakov višje ali nižje. Da me postaviš višje ali nižje, uporabi ukaz *zamenjaj y za 30* oziroma *za* -30. Če želim priti na začetek vrstice, moram iti 300 korakov v levo (10*30). Začetek vrstice pa lahko določiš tudi drugače. Razmisli, kako!

c) Polje kvadratov

Rešitev:

UPORABA SPREMENLJIVKE

Spremenljivko lahko uporabimo za shranjevanje podatkov. Uporabna pa je tudi zato, ker se njena vrednost med izvajanjem programa lahko poljubno spreminja. Zato se tudi imenuje spremenljivka.

V tem poglavju lahko spoznaš, v katerih primerih spremenljivka pride prav in kako jo uporabiti.

5. primer: Z uporabo spremenljivke lahko nariševa spiralo na spodnji sliki.

Kako je nastala? Pozorno poglej v središče spirale:

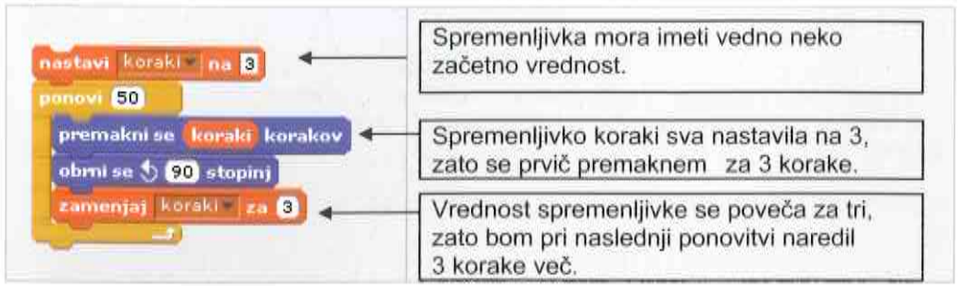
- začnem s tremi koraki v desno in se obrnem za 90°,
- naredim 6 korakov in se zopet obrnem za 90°,
- naredim 9 korakov in se zopet obrnem za 90°,
- to ponovim 50-krat.

Povzemiva: ko rišem spiralo, 50-krat narišem črto in se obrnem za 90°. Velikost črte ni nikoli enaka, ampak je vsakič večja za 3 korake. Pri tej težavi nama bo prav prišla spremenljivka.

Odpri program Svinčnik. Izberi skupino *Spremenljivke* Spremenljivke. Klikni na gumb *Ustvari spremenljivko* in v pogovorno okno, ki se odpre, napiši ime spremenljivke. Ime izberi smiselno, glede na vsebinski pomen spremenljivke, na primer *koraki*.

Naredila sva spremenljivko *koraki*. Število korakov pri risanju ne bo več določeno z nekim številom, ampak z vrednostjo spremenljivke *koraki* 

. Napišiva ukazni blok, ki bo risal spiralo, in poglejva, kako se število korakov v zanki vsakič poveča za 3.



Kako s spremenljivko vplivamo na število korakov, si lahko pogledava tudi v spodnji tabeli.

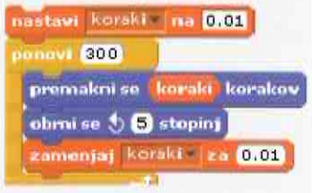
	koraki Vrednost spremenljivke	premakni se koraki korakov
Na začetku	3	
1. v zanki	3	Figura se premakne za 3 korake.
2. v zanki	6	Figura se premakne za 6 korakov.
3. v zanki	9	Figura se premakne za 9 korakov.

10. naloga: Odpri program Svinčnik in ga shrani kot Spirale. S pomočjo spremenljivke nariši še tri različne spirale. Prva spirala je podobna kot pri 5. primeru, a manjša.

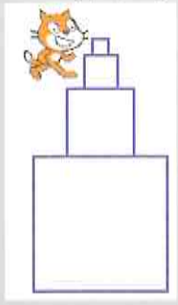
Rešitev:

Rešitev:

Rešitev:



11. naloga: Odpri program Svinčnik in ga shrani kot Stolp. Z uporabo spremenljivke nariši stolp s štirimi kvadratnimi bloki; vsak naslednji blok ima dvakrat krajšo stranico. Pri štirih kvadratih uporaba spremenljivke še ni nujno potrebna. Kaj pa, če je kvadratov deset ali še več?



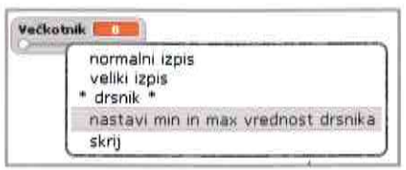
Rešitev:

6. primer: Pri tem primeru bova spoznala, kako lahko uporabnik programa sam izbere, kateri večkotnik želi narisati.

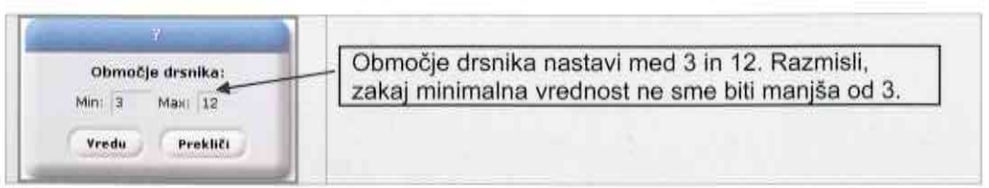
Ustvari novo spremenljivko in jo poimenuj Večkotnik. Odkljukaj potrditveno polje pred spremenljivko  . Na odru lahko sedaj spremljava, kolikšna je vrednost spremenljivke.

Z desno tipko miške na odru klikni na gumb *Večkotnik*. Odpre se pogovorno okno, v njem izberi drsnik.

Poleg spremenljivke se tako prikaže še drsnik. S pomočjo drsnika lahko določaš vrednost spremenljivke.



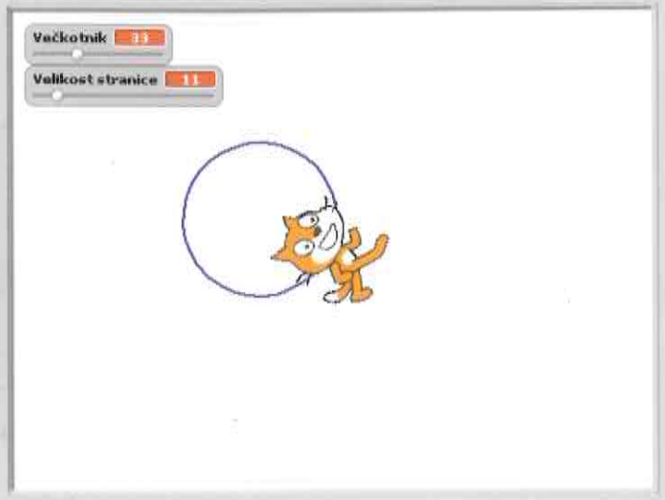
Določiš lahko tudi največjo in najmanjšo izbiro vrednosti spremenljivke.



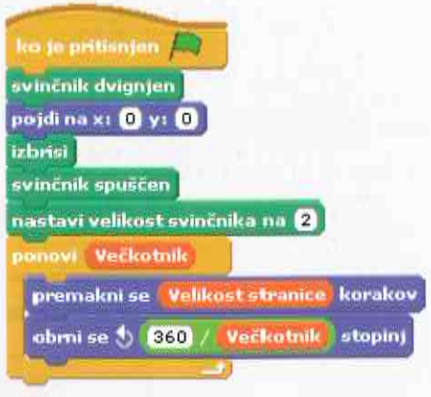
Napišiva program, ki nariše poljuben večkotnik. Število kotov naj določi uporabnik programa s pomočjo drsnika (število kotov sva že omejila med 3 in 12). Primer je spodnja slika, kjer je uporabnik označil število 5 in narisal petkotnik.

Podatek, ki se bo spreminjal, bo število kotov v večkotniku. Če si še enkrat pogledava nalogo 5, bova zlahka prišla do rešitve.

12. naloga: Nadaljuj kar s primerom 6. Shrani ga kot Drsnik in preoblikuj program tako, da lahko uporabnik poleg števila kotov določa tudi velikost stranice večkotnika. Smiselno izberi območje obeh drsnikov.



Rešitev:



ŠTAMPILJKA

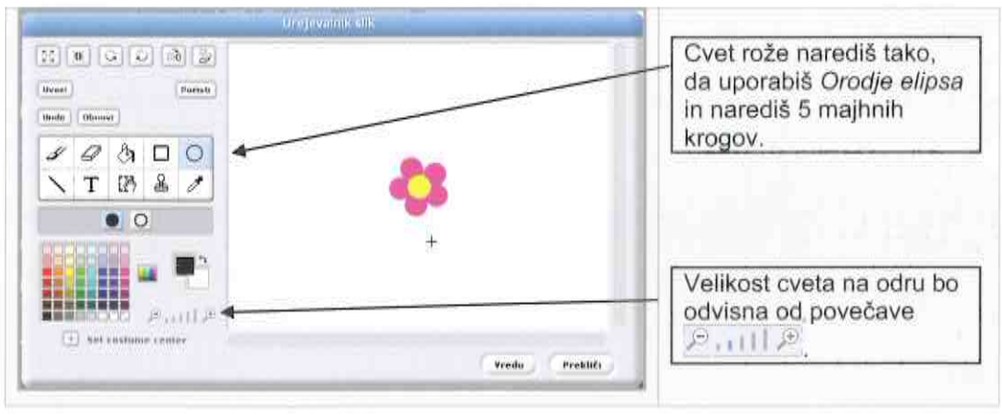
Če želiš spremljati gibanje figure, lahko uporabiš tudi ukaz *štampiljka*. Namesto črt bodo na platnu nastajali vzorci z različnimi motivi.

Zaženi program Scratch in iz skupine *Svinčnik* izberi ukaz *štampiljka*. Začela bova s preprostim primerom.

7. **primer:** Napisala bova program, ki bo naredil vrsto rož.

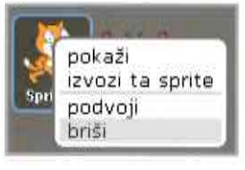
Zato morava narediti novo figuro. Narediš jo tako, da izbereš *Nariši novo skupino slik*.

Odpre se okno *Urejevalnik slik*. V njem nariši majhen cvet. Figuro poimenuj Roža.



Če se ti cvet na odru zdi prevelik, ga lahko pomanjšaš z uporabo ukaza *nastavi velikost na* nastavi velikost na % ali z orodjem *Pomanjšaj (Shrink)*.

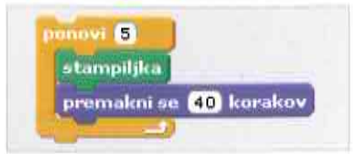
V tej in naslednjih nalogah se bom začasno umaknil z odra, zato me izbriši. Klikni z desno tipko miške na ikono Praskača ter v izbirnem oknu izberi *briši*.



Ker bova uporabljala ukaze, ki so povezani s koordinatnim sistemom, v Uvodu še enkrat preberi poglavje Premikanje. Nato odčitaj koordinati središča cveta.

Novonastala figura je v središču odra.

Napiši spodnji program, dvakrat klikni na ukazni blok in poglej, kaj se bo zgodilo.



Pričakovala sva, da bo imel nastali vzorec 5 rož, na platnu pa jih vidiva 6. To težavo lahko hitro rešiva.

Klikni na zadnjo rožo v vrsti in pokazalo se bo, da se jo da premikati. Roža je torej pustila le pet odtisov, kot sva določila v programu, zadnja roža pa ni odtis, ampak figura.

Število korakov, ki določajo razdaljo med rožami, pri naslednjih nalogah določiš glede na velikost tvoje figure. To velja tudi za druge parametre, ki so odvisni od velikosti figure.

13. naloga: Napiši program, ki s klikom na zeleno zastavico počisti ekran, figuro postavi v sredino platna in jo usmeri v desno. Program poimenuj Štampiljka. Program ima podobno vlogo kot program Svinčnik, torej pripravo platna za žigosanje.

Rešitev:

14. naloga: Odpri program Štampiljka in ga shrani kot Rožni krog. Napiši program, ki nariše rože v krogu.

Pomoč: Rožo na vsakem koraku zasukaj in premakni.

Rešitev:

15. naloga

Odpri program Štampiljka in ga shrani kot Rože v vrsti.
Napiši program, ki nariše tri vrste rož.

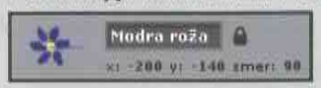
Rešitev 1:

Rešitev 2:

Opomba: Lažje kot preračunati, kje se bo začela koordinata x nove vrstice, je odčitati začetno lego figure: x = 0.

16. naloga: Odpri program Štampiljka in ga shrani kot Rožni vrt. Napiši program, ki nariše rožni vrt.

Pomoč: Rožni vrt ima dva različna motiva, zato nariši še eno figuro, modro rožo. Poimenuj jo Modra roža. Postavi jo v levi spodnji kot odra in odčitaj koordinati.

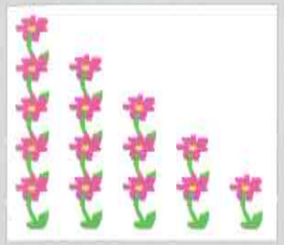


V tem primeru je x enak -200, y pa je enak -140. Za rožnat cvet je potrebno koordinate preračunati glede na lego modrega cveta. Postavimo rožnat cvet 40 korakov višje in 40 korakov desno. To pomeni, da bo koordinata x rožnatega cveta enaka $-200 + 40 = -160$ in koordinata y enaka $-140 + 40 = -100$. Vsaki figuri napiši program in ga zaženi.

Rešitev:

Rešitev:

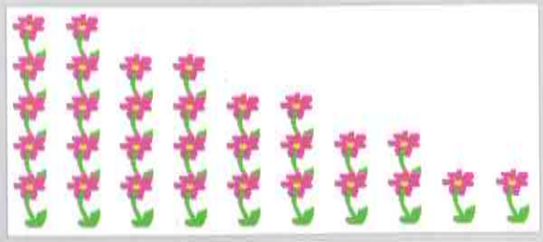
17. naloga: Odpri program Štampiljka in ga shrani kot Trikotnik rož. Lahko narišeš nov motiv ali pa pustiš enega od narisanih. Napiši program, ki naredi vzorec v obliki trikotnika.



Pomoč: Podobno kot v prejšnjih primerih bo zunanja zanka risala vrstice. Ponovila se bo 5-krat. Notranja zanka pa bo skrbela za risanje rož v posamezni vrstici. V prvi vrsti je le ena roža, v drugi vrsti sta dve, v tretji tri in tako naprej. Zanka, ki riše rože v vrstici, se bo torej izvajala najprej enkrat, nato dvakrat, potem trikrat itd. Ker je v vsaki vrstici različno število rož, uporabi spremenljivko. Razmisli, kako jo uporabiti.

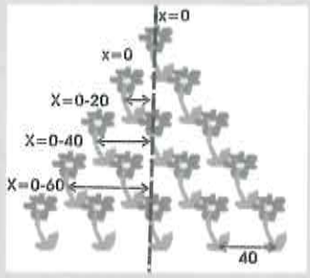
Rešitev:

18. naloga: Odpri program Trikotnik rož in ga shrani kot Trikotnik rož 2. Stari program izbriši ali pa preoblikuj v novega, ki bo narisal trikotnik rož nekoliko drugače. Ker je zadnja vrsta že zelo dolga, pazi, da rože ne bodo prevelike.



Rešitev:

19. naloga: Odpri program Trikotnik rož in ga shrani kot Piramida. Stari program preoblikuj v program, ki bo narisal piramido rož.



Pomoč: Program bo skoraj enak prejšnjemu, razmisli le, kako določiti koordinato x prve rože v novi vrstici. Kot vidiš, je koordinata x v vsaki vrstici za 20 manjša.

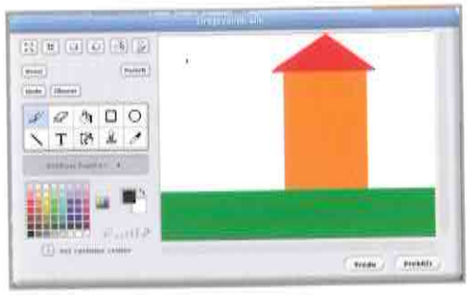
Vrsta	Koordinata x	Vrednost spremenljivke	Pravilo
1.	x 0	1	
2.	x = 0 – 20	2	x = 0 – 1*2
3.	x = 0 – 40	3	x = 0 – 2*20
4.	x = 0 – 60	4	x = 0 – 3*20
			x = 0 – vrednost spremenljivke*20

Med vrednostjo spremenljivke in določanjem pravila opaziš zamik. Če pogledaš prejšnji nalogi, se vrednost koordinate x določi predhodno, torej tudi z zamikom.

Rešitev:

NALOGE Z OZADJEM

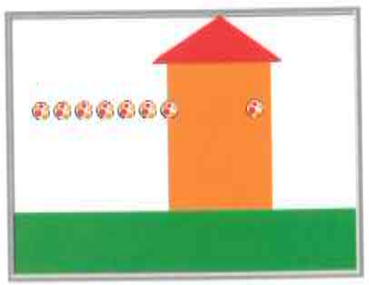
Pri naslednjih nalogah se bova poigrala z barvami. Za ta namen bova narisala različna ozadja. Začniva s prvim ozadjem. Klikni na ikono Ozadje (Stage). Izberi jeziček *Ozadja* in klikni na gumb *Uredi*. Odpre se okno Urejevalnik slik. Nariši ozadje s travnikom in hišico, podobno kot je na spodnji sliki.



Izbriši figuro Praskača in uvozi novo s klikom na gumb *Izberi nov sprite iz datoteke*.

V mapi Stvari (Things) izberi figuro žoga (beachball1).

8. primer: Napisala bova program, ki žogo 10-krat premakne za 30 korakov v desno, odtis pa pusti le, če je barva ozadja bela.



V skupini *Zaznavanje* poišči ukaz, ki prepozna barve na odru:  in s pomočjo *kapalke* določi ustrezno barvo.

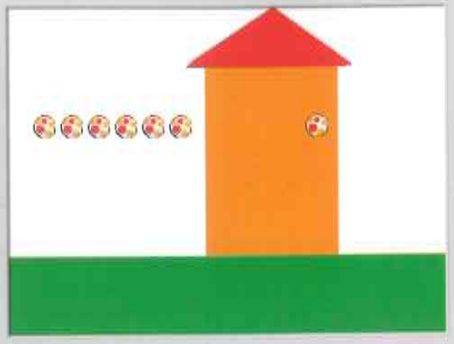
Sestavi ukazni blok, ki naredi 10 odtisov žog. To že znaš.

Ker želiš odtis žoge le na belem ozadju, dodaj še pogoj: če se žoga dotika bele barve, pusti odtis. Premik žoge naj se zgodi v vsakem primeru.

Napiši še program, ki žogo pomanjša, postavi jo na primerno izhodišče ter počisti vse sledi na platnu.

K temu pripni ukazni blok, ki riše žoge, in poglej, kaj nastane. Kot vidiš, žoga pusti odtis tudi tam, kjer se le delno dotika bele barve. To lahko poskusiš rešiti v naslednji nalogi. Ne pozabi, da je zadnja žoga v vrsti figura in ne odtis. Program shrani z imenom Žoga.

20. naloga: Odpri program Žoga in ga shrani kot Žoga 1.
Program žoga preuredi tako, da žogo 10-krat premakne za 30 korakov, odtis pa pusti le, če je barva ozadja v celoti bela.
Pomoč: Pomagaj si z operatorjem ne.

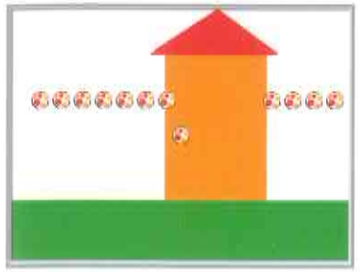


Rešitev:

PONOVI DOKLER, ZA VEDNO ČE

Doslej sva se srečevala s problemi, ko je bilo za neko dejanje natančno določeno, kolikokrat naj se ponovi, npr. 3-krat, 5-krat. Pogosto pa ne veš vnaprej, kolikokrat se bo neko dejanje ponovilo. Če na primer želiva priti do konca nogometnega igrišča, ne bova rekla: »Narediti morava še 100 korakov!« Prav je: »Korakajva toliko časa, dokler ne prideva do konca igrišča.« Če na vprašanje *Ali sva prišla do konca igrišča?* lahko odgovoriva **Da**, je pogoj izpolnjen in se ustaviva, drugače pa korakava naprej. Vprašava pa se lahko tudi drugače. Kako, bova videla v primeru 9.

9. primer: Program bova napisala tako, da bo žoga puščala odtise na beli površini, dokler ne pride do konca odra. Ko pride do konca odra, se postavi točno v sredino.



Nalogo lahko rešiva na dva načina: z uporabo ukaza *ponovi dokler* ali z ukazom *za vedno če*. Odpri program Žoga in poskusi rešiti zastavljeno nalogo. Če ne gre, si pomagaj s spodnjima rešitvama.

1. Rešitev:

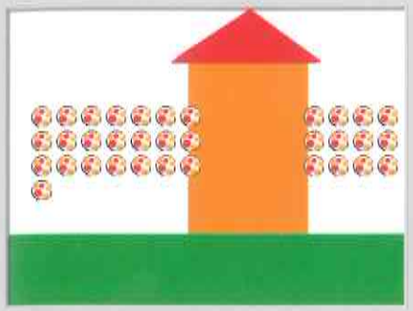
Zanka *Ponovi dokler* se bo izvajala toliko časa, dokler žoga ne bo prišla do roba platna. Torej če na vprašanje *Ali se dotika roba?*, lahko odgovoriva z **da**, se izvajanje zanke konča. Program se nato nadaljuje z ukazom *pojdi na x: y:*

2. rešitev

Ravno obratno je pri drugi rešitvi, kjer se zanka izvaja toliko časa, dokler je pogoj resničen. Ko na vprašanje *Ali se ne dotikam roba*? odgovoriš z **ne**, se izvajanje zanke konča.

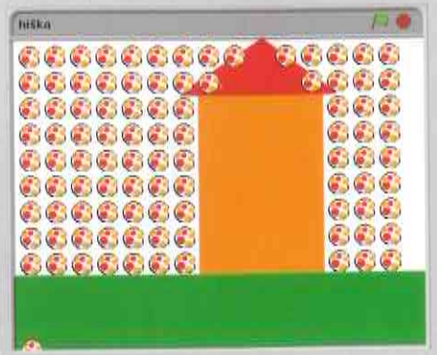
Ukaza se razlikujeta še po eni stvari. Pri ukazu *ponovi dokler* lahko na koncu zanke pripenjava nove ukaze, pri ukazu *za vedno če* pa tega ne moreva. Pri drugi rešitvi zato še vedno manjka ukaz, ki postavi žogo v središče zaslona. To se da rešiti drugače. V zanki na koncu dodatno preveriva, ali se figura dotika roba. V tem primeru jo postaviva na sredino zaslona in končava. Opaziva, da isti pogoj (ali je figura na robu) preverjava pri vsaki izvedbi zanke dvakrat, kar ni smiselno. Zato je v tem primeru bolje uporabiti prvo rešitev. Prvo rešitev shrani kot Žoga 2.

21. naloga: Odpri program Žoga 2 in ga shrani kot Žoga 3.
Verjetno ne bo težko napisati podobnega programa kot prej, če želiš imeti tri vrste žog od začetka do konca zaslona.



Rešitev:

22. naloga: Odpri program Žoga 3 in ga shrani kot Žoga 4.
Dopolni program tako, da bo celotna bela površina platna polna odtisov žog.



Pomoč: Risati začni v levem zgornjem kotu zaslona. Žoga se bo nato premikala od leve proti desni strani toliko časa, dokler ne bo prišla do dna.

Rešitev1:

Opomba: Rešitev je podobna prejšnji, le da namesto treh ponovitev risanja vrstic ponavljaš risanje vrstic, dokler y ni manjši od -180.

Rešitev2:

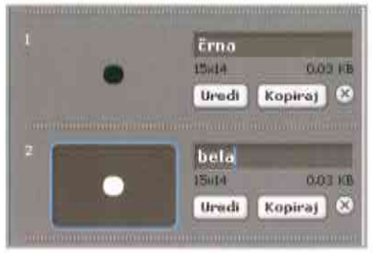
Opomba: Kot vidiš, lahko zanko *ponovi dokler* uporabiš le enkrat. V zanki se žoga premika po 30 korakov v desno, dokler ni y koordinata manjša od –180. Za skok v novo vrstico znotraj zanke skrbi pogoj, ki preverja, ali je žoga prišla do desnega roba odra. V tem primeru žogo premakne na začetek vrstice.

ČE, DRUGAČE PA

Odpri nov projekt in nariši novo ozadje s tremi različnimi barvami.

Nariši novo figuro, majhno črno piko, in jo poimenuj Pika. Figuro Praskača lahko izbrišeš.

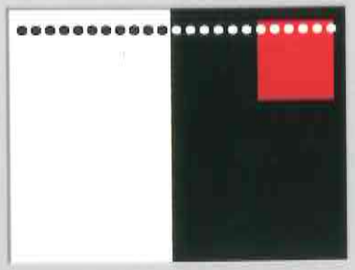
V naslednjih nalogah bova na platno risala črne in bele pike. Pika bo zato imela dve obleki, črno in belo. Klikni na ikono Pika in izberi jeziček *Obleke*. Klikni na gumb *Kopiraj*. Nastane nova obleka, ki je enaka prvi. Klikni na gumb *Uredi* in v Urejevalniku slik s pomočjo orodja *Zapolni* piko prebarvaj na belo. Obleki poimenuj črna in bela.



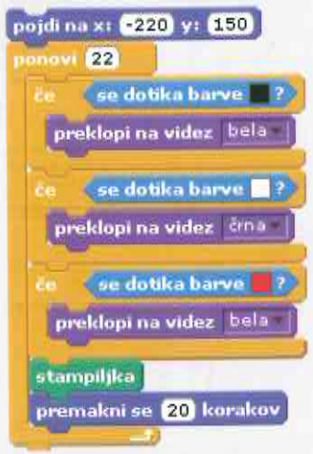
Z ukazom *preklopi na videz* lahko izbiraš med črno in belo piko.

Naslednje naloge bova rešila z uporabo stavkov *če* in *če drugače pa*. Poskusi jih rešiti samostojno.

23. naloga: Napiši program, ki nariše celo vrsto pik. Pike naj bodo na belem ozadju črne, na črnem pa bele. Program shrani kot 101 dalmatinec. Naredi dovolj velik korak, sicer se bo figura dotikala sama sebe in ne bo ustrezno spremenila barve.



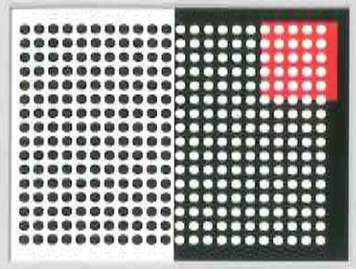
Rešitev 1:



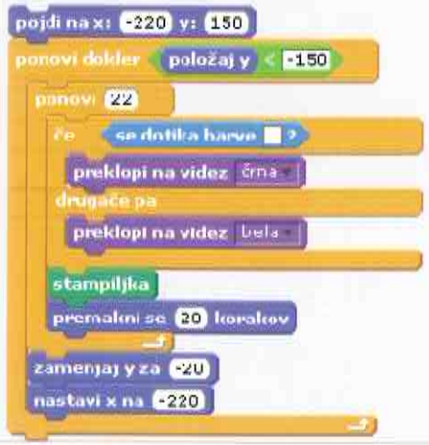
Rešitev 2:

Opomba: Rezultat je pri obeh rešitvah enak, vendar se program pri drugi rešitvi izvede hitreje, saj se pogoj preverja enkrat namesto trikrat.

24. naloga: Odpri program 101 dalmatinec in ga shrani kot 102 dalmatinca. Preuredi program tako, da bo ves zaslon prekrit s pikami.



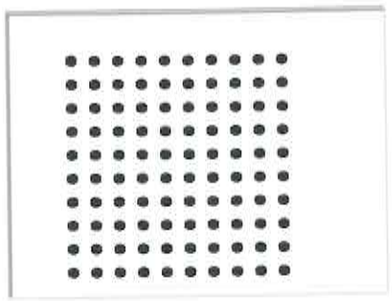
Rešitev:



ENA VRSTA RDEČA, DRUGA ČRNA ...

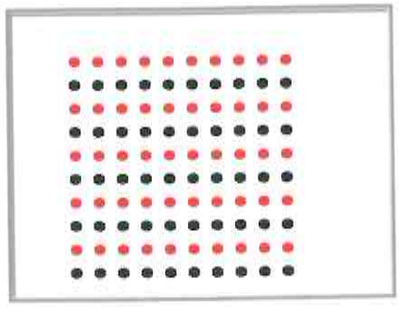
10. primer: Odpri program Pikice in ga shrani kot 103 dalmatinci. Izbriši barvno ozadje: klikni na ikono Ozadje (Stage), izberi jeziček *Ozadje* in klikni na gumb *Uredi*. V oknu Urejevalnik slik barvno ozadje izbriši tako, da klikneš na gumb *Počisti*.

Začni s preprostim primerom. Za začetek naredi 10 vrst po 10 črnih pik. Začni v levem spodnjem kotu.



Problem verjetno ni pretežak.

Naslednja naloga bo precej težja. Narisati želiva izmenično eno vrsto črnih in eno vrsto rdečih pik. Preprosta rešitev bi bila, če bi narisala eno vrsto črnih in eno vrsto rdečih pik in nato vse skupaj petkrat ponovila.



Vendar bova raje uporabila naslednjo idejo, ki bo uporabna še marsikdaj. Gledano od spodaj navzgor, je vsaka liha vrstica črna, vsaka soda pa rdeča. Torej če je vrstica liha, bova uporabila ukaz *preklopiva na videz črna*, če je vrstica soda, pa ukaz *preklopiva na videz rdeča*.

Pri določanju, ali je vrstica soda ali liha, si bova pomagala s spremenljivko. Če bo vrednost spremenljivke 0, bo vrstica liha, če bo vrednost spremenljivke 1, bo vrstica soda.

Piki naredi še rdečo obleko in poglej spodnji rešitvi. Na videz sta obe pravilni, vendar pa je desna rešitev napačna. Razmisli, zakaj.

Pravilna rešitev:

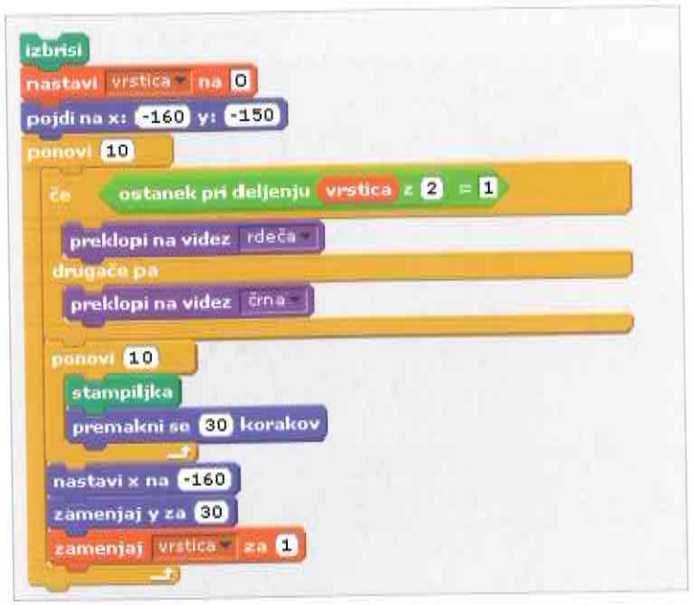
Napačna rešitev:

Če pogledaš rešitev, vidiš da se v pogojnem stavku vrednost spremenljivke poveča za ena ali zmanjša za ena in tako skače med 0 in 1.

Med vrednostjo 0 in 1 se lahko premikaš tudi drugače. Poglejva si primer ostanka pri deljenju z 2:

$1:2=0$, ostanek je 1;
$2:2=1$, ostanek je 0;
$3:2=1$, ostanek je 1;
$4:2=2$, ostanek je 0.

Opaziva, da je ostanek pri deljenju izmenično 1 ali 0. Razmisliva, kako bi to lahko uporabila s pomočjo operatorja *ostanek pri deljenju*  . Če ne gre, poglej spodnjo rešitev.



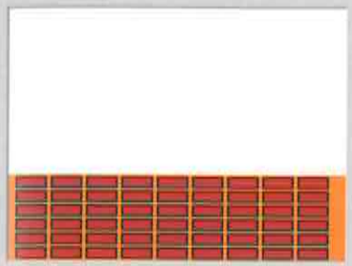
Pri tej rešitvi ni nujno, da uporabiš ukaz *če drugače pa*. Lahko tudi dvakrat uporabiš ukaz *če*. V tem primeru pogoj preverjaš dvakrat, zato je bolje uporabiti *če drugače pa*.

Za vajo lahko narediš še podobno nalogo, le da uporabiš tri različne barve (uporabi ostanek pri deljenju s 3).

Pridobljeno znanje lahko uporabiš za risanje zidu iz opek pri naslednjih nalogah.

25. naloga: Zaženi Scratch in nariši novo ozadje.

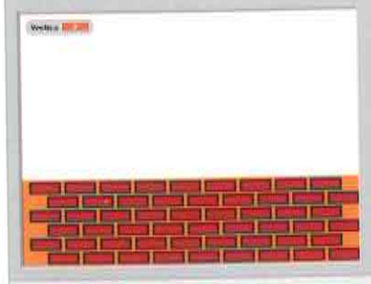
Izbriši figuro Praskač in nariši majhno opeko. Napiši program, ki bo narisal nekaj vrst opek. Število vrstic in število opek v vrstici je odvisno od velikosti opeke. Program shrani kot Zid.



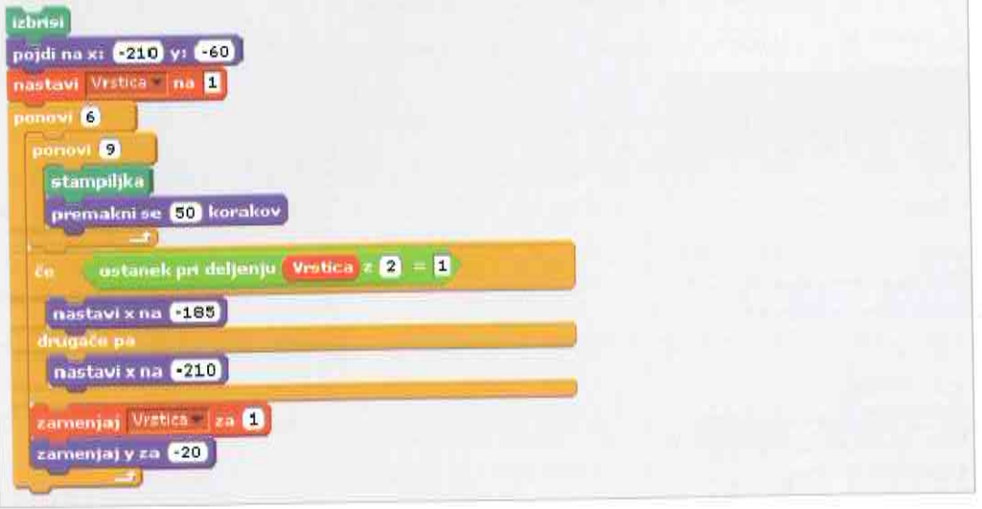
Rešitev:

26. naloga: Odpri program Zid in ga shrani kot Zid 1.
Program preuredi tako, da bo vsaka druga vrsta opek v zidu zamaknjena.

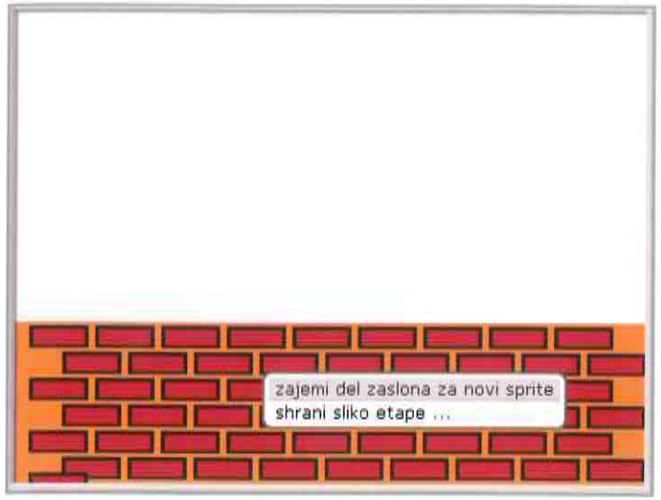
Pomoč:
Rišeš dve različni vrsti opek. Spomni se, kako sva v 10. primeru narisala različni barvi pik.



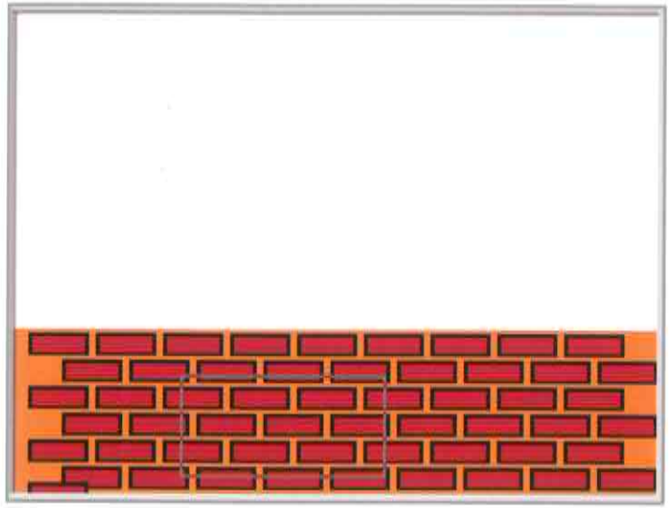
Rešitev:



In še majhen trik. Iz vzorca, ki nastane, lahko narediš novo figuro. Če z desno tipko miške klikneš kjerkoli na oder, se odpre izbirno okno.



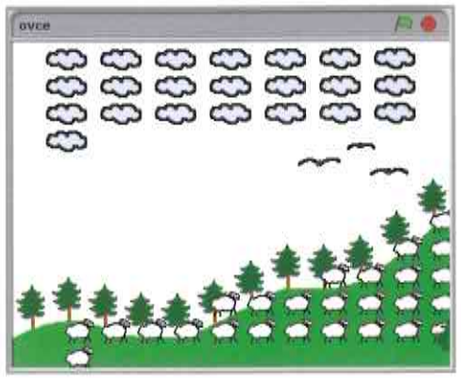
Izberi *zajemi del zaslona za novi sprite* in označi področje za novo figuro.



Ko miško spustiš, iz označenega dela zaslona nastane nova figura. 

PIKA NA I, PREDVAJAJ IN KO PREJMEM

11. primer: Naredila bova sliko s štirimi različnimi motivi. S klikom na zeleno zastavico se bodo vzorci začeli risati v določenem vrstnem redu, tako da bo vse skupaj učinkovalo kot animacija.



Ker je problem zahtevnejši, bo najbolje, da ga razdeliva na več korakov:

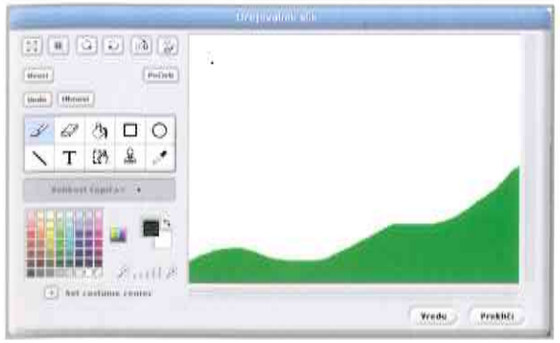
1. narisala bova štiri različne **motive** (figure);
2. narisala bova **ozadje**;
3. vsakemu motivu bova napisala **program**;
4. naredila bova **animacijo** tako, da se postopno prikazujejo motivi; najprej se prikažejo oblaki, nato smreke in na koncu hkrati še ovce ter ptice.

MOTIVI

Zaženi Scratch, izbriši figuro Praskač in nariši štiri različne figure majhne velikosti: oblak, smreko, ovco in tri ptice.

OZADJE

Nariši ozadje, travnik.



PROGRAMIRANJE

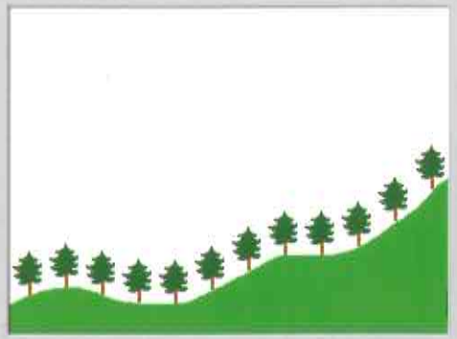
Motiv oblaka

27. **naloga:** Napiši program, ki nariše tri vrste oblakov.

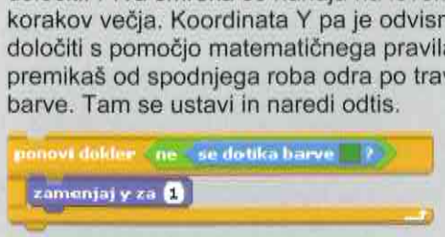
Rešitev:

Motiv smreke (1. način)

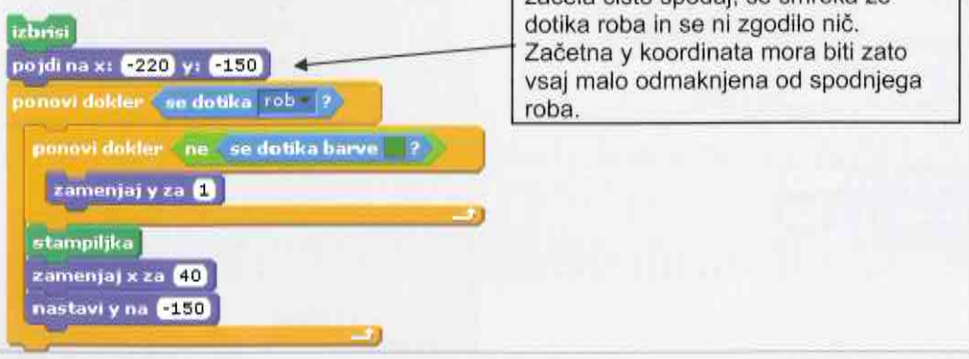
28. naloga: Napiši program, ki na vrhu travnika nariše smreke.



Pomoč: Vsaki smreki moraš določiti koordinati. Najprej določi koordinato x. Te ni težko določiti. Prva smreka se nahaja na levem robu odra. Nato je koordinata x vsakič za 40 korakov večja. Koordinata Y pa je odvisna od razgibanosti travnika in je ne moreš določiti s pomočjo matematičnega pravila. Lahko jo na primer določiš tako, da smreko premikaš od spodnjega roba odra po travniku navzgor, dokler se ne dotika več zelene barve. Tam se ustavi in naredi odtis.



Rešitev:



Če sva pri določanju koordinate y začela čisto spodaj, se smreka že dotika roba in se ni zgodilo nič. Začetna y koordinata mora biti zato vsaj malo odmaknjena od spodnjega roba.

Opomba: Določanje koordinate y je precej zamudno. Na koncu poglavja si poglej nekoliko težjo, vendar učinkovitejšo rešitev.

Motiv ovce

29. naloga: Napiši program, ki bo po celotni površini travnika narisal ovce.

Pomoč: Ovce se pasejo po celotnem platnu, od leve proti desni, dokler ne pridejo do dna. Odtis pustijo le v primeru, če pridejo na travnik.

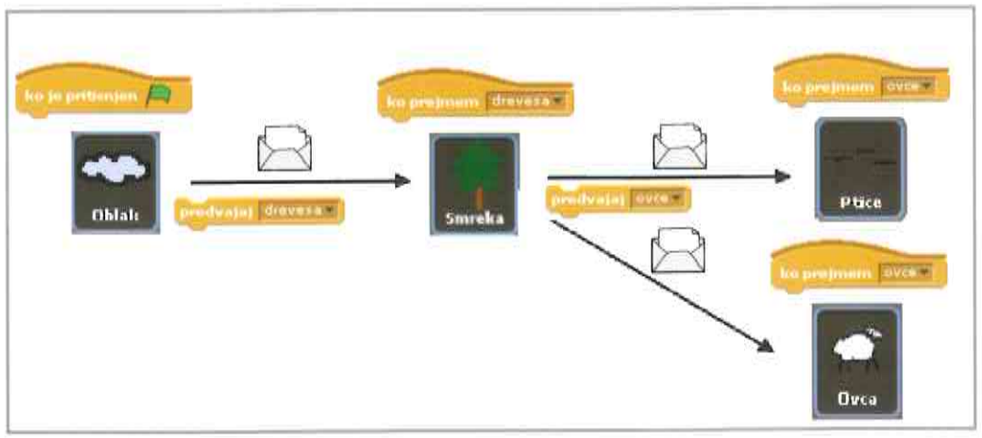
Rešitev:

Ker figuri ne menjaš smeri in je ne obračaš, je ves čas obrnjena v desno. Zato sta ukaza *premakni se 40 korakov* ali *zamenjaj x za 40* v tej nalogi enakovredna.

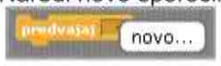
ANIMACIJA

Ukaza *predvajaj*  in *ko prejmem*  skrbita, da se stvari pri animaciji odvijajo po predvidenem vrstnem redu.

Vsaka figura mora vedeti, kdaj se začne njeno izvajanje oziroma kdaj se pojavi na odru. Eden od načinov, kako figura ve, kdaj se pojaviti na odru, je, da prejme sporočilo z ukazom *ko prejmem*. Sporočilo, ki ga lahko pošlje vsaka figura (lahko ga pošlje tudi ozadje), pa je poslano kot ukaz *predvajaj*. Vsako sporočilo ima neko ime, ki ga smiselno izbereš. To prikazuje spodnja shema.



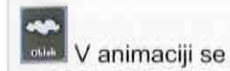
Naredi novo sporočilo. Klikni na ukaz *predvajaj* in v izbirnem oknu, ki se odpre, izberi novo.



Odpre se pogovorno okno, kamor napišeš ime sporočila, na primer drevesa.

To sporočilo lahko prejme poljubna figura s pomočjo ukaza *ko prejmem*.

Dopolniva ukazne bloke tako, da bo nastala animacija.


V animaciji se oblaki prikažejo na zaslonu prvi, in sicer s klikom na zeleno zastavico. Zato na vrhu programa dodaj še ukaz

Ko je risanje oblakov končano, program pošlje naprej sporočilo smrekam. Sporočilu daj ime ***drevesa***.


Program se bo začel izvajati, ko bo prejel sporočilo ***drevesa***. Ko se bo končal, pa bo poslal sporočilo z imenom ***ovce***, ki ga bodo prejele ovce in ptice.


Ovce naj se pojavijo, ko prejmejo sporočilo ***ovce***.


Ptice se pojavijo, ko prejmejo sporočilo ***ovce***, torej hkrati z ovcami. Na začetku programa so ptice skrite, ko prejmejo sporočilo ***ovce***, pa se prikažejo.

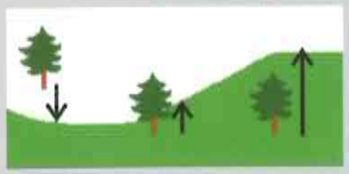
Motiv smreke (2. način)

30. naloga: Smreki poišči manj zamudno določanje koordinate y.

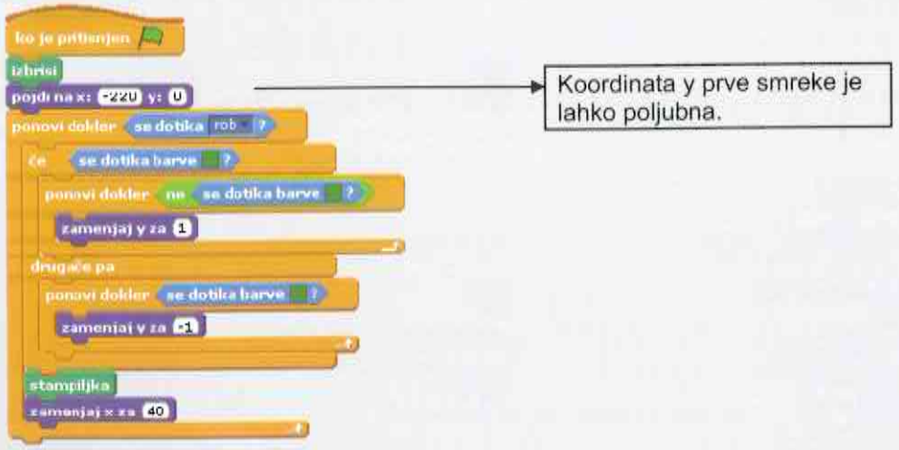
Pomoč: Smreko premikaš po 40 korakov v desno. Tako določiš koordinato x.

Sedaj pa preveri, na kakšni barvi se nahaja smreka. Če se nahaja na beli barvi, jo moraš spuščati toliko časa, da se dotakne zelene barve. Če pa se delno ali v celoti nahaja na zeleni barvi, se premika navzgor toliko časa, dokler se zelene barve ne dotika več.

Najprej preveri, ali se smreka dotika zelene barve, ker s tem zajameš tiste smreke, ki so v celoti na zeleni površini ali pa delno na zeleni in delno na beli površini. Te se dvigajo. Nato ostanejo le tiste smreke, ki so v celoti na beli površini. Te pa se spuščajo.



Rešitev:



RAČUNALNIŠKE IGRE

V tem delu bova izdelala deset računalniških iger. Za vsako igro bova naredila glavne junake in druge elemente in jim napisala programe. V eni igri bom glavni junak kar sam.

Igre bova naredila po korakih, od načrtovanja do sestavljanja ukaznih blokov in končne podobe. Posamezne probleme bova opisala tako, da bo program pogosto mogoče sestaviti tudi brez moje pomoči. Na koncu vsake igre te v poglavju *Računalniški maček* čakajo tudi naloge, ki so zastavljene kot izziv za samostojno programiranje. Te naloge nimajo rešitev, saj so namenjene temu, da lahko daš duška svoji domišljiji, rišeš, se igraš z glasbenimi učinki in najdeš povsem originalno pot.

Igre si sledijo tako, da se znanje nadgrajuje in utrjuje, podobno kot v šoli, le da je vse skupaj bolj zabavno, predvsem pa si sam svoj učitelj. Pri izdelovanju lastnih iger se učiš programiranja, hkrati pa razvijaš domišljijo, logiko in spoznavaš multimedijske veščine. Skratka, postajaš pravi računalniški maček.

AKVARIJ

OPIS IGRE

Akvarij je preprosta igra. Glavni junak, rak, lovi ribe, ki se naključno gibajo po prostoru. Ko rak ujame ribo, ta za nekaj trenutkov izgine.

NAČRT

Najprej narediva načrt. Določiva glavne figure in druge elemente igre ter poglejva, kako so elementi med seboj povezani.

Elementi igre	Dejanje	Program napišemo elementu
Rak	- Premika se v vse smeri. - Upravljamo ga s puščicami na tipkovnici ←,↑,→,↓.	
Riba	- Se naključno premika po odru.	
Rak in Riba	Če rak ribo ujame, **riba**: - spusti zvok »pop«, - izgine, - se ponovno prikaže.	
Stage Ozadje	- Je kulisa. - V ozadju se sliši brbotanje vode.	

Programiranje igre bova razdelila na 6 korakov:

1. naredila bova figuro **Rak** in določila njeno gibanje;
2. naredila bova figuro **Riba**, ki se naključno premika po prostoru;
3. določila bova povezavo med **Rakom in Ribo**;
4. igri bova dodala **ozadje;**
5. igro bova popestrila z **zvoki**;
6. dodala bova **še nekaj rib;**
7. pogledala bova, kako lahko svoj projekt naložiš na **spletno stran programa Scratch**.

RAK

Rak je glavni junak te igre. S puščicami na tipkovnici ga lahko vodimo gor, dol, levo in desno. V oknu nove figure klikni na gumb *Izberi nov sprite iz datotek* in v mapi Živali (Animals) izberi raka (crab1-a).

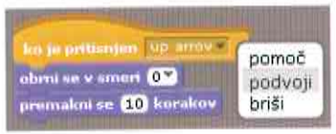
Nastala je nova figura z imenom Sprite2. Preimenuj jo v Rak.

Izbriši mojo figuro, saj v tej igri ne bom nastopal. Klikni z desno tipko miške na ikono Praskača ter v izbirnem oknu izberi *briši*.

Zdaj lahko sestaviva program. Začniva s premikanjem Raka v desno. Želiva, da se ob pritisku na desno puščico rak obrne v desno smer ter naredi nekaj korakov. V skupini *Upravljanje* izberi ukaz *ko je pritisnjen*. V izvlečnem meniju izberi *puščica desno*.

Nato dodaj še ukaz za obrat desno in ukaz za premik.

Sestavi še ukazne bloke za premikanje Raka v ostale smeri in rezultat primerjaj s spodnjo rešitvijo. Manj dela bo, če uporabiš ukaz podvoji.



Rak se zdaj s pritiskom na puščice premika po odru. Verjetno se tudi tebi zdi, da na majhnem odru Rak zaseda preveč prostora. Zato je najbolje, da se ob začetku igre njegova velikost zmanjša na 60 % njegove začetne velikosti. Igra se bo pričela s klikom na *zeleno zastavico*.

RIBA

V oknu nove figure klikni na gumb *Izberi nov sprite iz datotek* in v mapi Živali (Animals) tokrat izberi eno od rib. Imenuj jo Riba.

Riba se bo ves čas igre naključno gibala po odru, in to na naslednji način:

- premaknila se bo za nekaj korakov,
- nato bo naredila majhen zasuk, ki bo naključen (med 0° in 15° v levo ali desno),
- kadar bo prišla do roba odra, se bo obrnila.

Riba se bo začela premikati na začetku igre s klikom na zeleno zastavico.

Ukazni blok za naključno gibanje ribe lahko sestaviš iz spodnjih ukazov. Hitrost njenega gibanja lahko spreminjaš tako, da izbiraš med različnim številom korakov ali med različno dolgimi premori.

Klikni na *zeleno zastavico* in poglej, kako se Riba premika. Svoj program primerjaj še s spodnjo rešitvijo.

Če je bil Rak malo prevelik, je Riba gotovo precej prevelika za platno. Pomanjšaj jo na 30 % njene velikosti.

KO RAK ULOVI RIBO

Vsakič, ko Rak ulovi Ribo, ta za nekaj sekund izgine, nato pa se ponovno prikaže. Čeprav je Rak tisti, ki lovi Ribo, se ob njunem dotiku videz spremeni Ribi. Zato morava pogoj, ki preverja, ali se Riba dotika Raka, napisati Ribi.

S pomočjo spodnjih ukazov dodaj pogoj, ki preverja, ali se Riba dotika Raka. Če je pogoj izpolnjen:

- se Riba skrije,
- počaka 2 sekundi in
- se ponovno prikaže.

Preveri, ali je tvoja rešitev enaka kot spodnja.

Če se zgodi, da igro končaš ravno v trenutku, ko Riba izgine, se pri ponovnem zagonu programa Riba ne bo prikazala, če ukaza *pokaži* ne dodaš pred zanko.

Na prvi pogled je tudi spodnja rešitev videti pravilna, vendar ni. Razmisli, zakaj.

Pogoj, ki preverja, ali se Riba dotika Raka, se preveri le enkrat in to na začetku igre, ko zaženemo program.

OZADJE

Da se ribe ne bodo gibale po dolgočasni belini, jim poišči slikovitejše ozadje. Klikni na ikono Ozadje (Stage) in izberi jeziček *Ozadja*. Klikni na gumb *Uvozi* in v mapi Narava (Nature) izberi primerno ozadje, na primer ozadje Pod vodo (underwater).

ZVOK

Igro bova popestrila tako, da ji bova dodala nekaj vodnih zvokov. Med predvajanjem igre se bo v ozadju slišalo brbotanje vode. Tudi zato morava napisati program. Ta zvok ni vezan na določeno figuro ali na ozadje. V takem primeru je smiselno, da program napiševa ozadju.

Klikni na ikono *Ozadje* (*Stage*) in izberi jeziček *Zvoki*. Klikni na gumb *Uvozi* in v mapi Učinki (Effects) poišči zvok Mehurčki (Bubbles). Klikni na jeziček *Skripte* in napiši program za predvajanje zvoka Mehurčki (Bubbles).

Preveri kaj se zgodi, če namesto ukaza *predvajaj zvok Bubbles in počakaj* uporabiš ukaz *predvajaj zvok Bubbles*.

Dodajva še en zvok, ki pa bo vezan na določeno figuro. Vsakič, ko se bo Rak dotaknil Ribe, se bo ta oglasila: »Pop.«

V seznamu figur klikni na ikono Ribe, s tem jo ponovno izbereš, in izberi jeziček *Zvoki*. Tudi glas *pop* uvoziva iz mape Učinki (Effects). Razmisli, kateremu ukaznemu bloku je treba dodati ukaz za predvajanje zvoka *pop*.

ŠE NEKAJ RIB

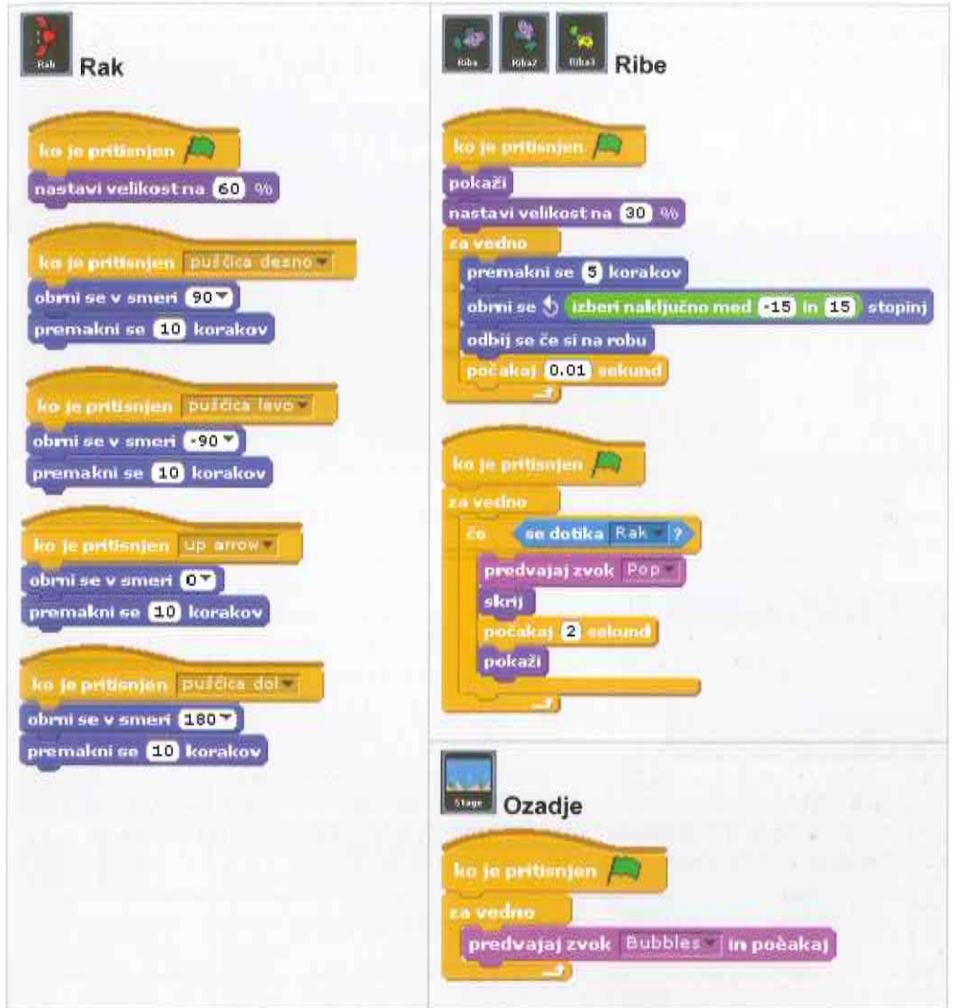
V akvarij dodajva še nekaj rib. Nove ribe bodo imele enak program, lahko pa imajo drugačen videz. To lahko narediva na dva načina. Lahko podvojiva ribo s programom vred ali pa narediva ribo z drugačnim videzom in podvojiva le program. Narediva vsak primer enkrat.

Najprej podvojiva Ribo s programom vred. Z desno tipko miške klikni na ikono Riba in v pogovornem oknu izberi *podvoji*. Nastala je nova, popolnoma enaka riba. Ribi se razlikujeta le po imenu. Novonastalo ribo preimenuj v Riba2.

Tretjo ribo bova naredila tako, da bova podvojila le program. Uvozi novo ribo drugačnega videza. Označi ikono Riba in ukazne bloke povleci v ikono nove ribe. Pri tem je treba biti pazljiv, saj se program prenese le, če se okoli ikone pokaže svetel okvir, kot je razvidno na spodnji sliki. Tretjo ribo poimenuj Riba3.

Za vsako od rib še poglej, kaj se je zgodilo z njenimi zvoki.

Ker je to prva igra, ki sva jo naredila, ti bo morda prav prišla spodnja preglednica, kjer so napisani programi za vse figure in ozadje.



SVETOVNI SPLET

Ena od prednosti Scratcha je, da lahko izdelke, narejene s tem programom, deliš s prijatelji na svetovnem spletu. Če želiš svoj projekt naložiti na spletno stran programa Scratch, ga odpri v programu Scratch in v menijski vrstici klikni na ikono *Share this project*.

Odpre se bo pogovorno okno, v katerem opišeš svoj projekt. Da ga lahko naložiš, moraš vtipkati svoje uporabniško ime in geslo.

Če še nimaš svojega računa, si ga lahko zdaj ustvariš.

Izpolni obrazec in klikni na gumb *V redu*. Tvoj projekt se naloži na spletno stran, kar potrdi ustrezno sporočilo.

Če želiš svoje projekte videti na svetovnem spletu, moraš imeti na svojem računalniku nameščen program Java. To običajno že imaš, v nasprotnem primeru pa ga dobiš brezplačno na naslovu http://www.java.com/en/download/index.jsp.

RAČUNALNIŠKI MAČEK

Naloga: Igri Akvarij dodaj strupeno ribo, ki lovi Raka. Na svojem lovu se mora Rak ribi izogniti, sicer se igra konča. Izvajanje programov končaš z ukazom 

.

Namig: Premikanje strupene ribe je podobno premikanju ostalih rib. Razlika je le v obračanju. Obračanje strupene ribe ni naključno, ampak usmerjeno k raku. Pomagaj si z ukazom *obrni se proti* 

.

ZAČARANI GOZD

OPIS IGRE

V igri Začarani gozd vitez lovi prijaznega duhca, ki se naključno premika po gozdu. Vsakič ko ga vitez ujame, se velikost duhca nekoliko zmanjša. Ko ga vitez ulovi desetič, se duhec obrne proti njemu in ga močno prestraši.

NAČRT

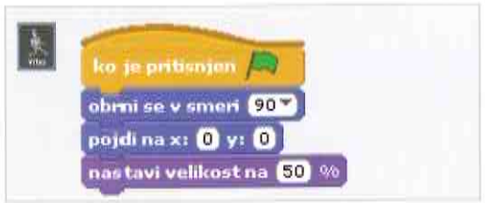
Elementi igre	Dejanje	Program napišemo elementu
 Vitez	- Premika se v vse smeri. - Upravljamo ga s puščicami na tipkovnici ←,↑,→,↓.	
 Duhec	- Se naključno premika po odru.	
 in 	Kadar se Vitez dotakne Duhca, se **Duhec** pomanjša.	
Spremenljivka Rezultat	Šteje, kolikokrat se Vitez dotakne Duhca.	
Cilj igre	Ko ima Rezultat vrednost 10, **Duhec** zavpije in te prestraši.	

Programiranje igre bova razdelila na naslednje korake:

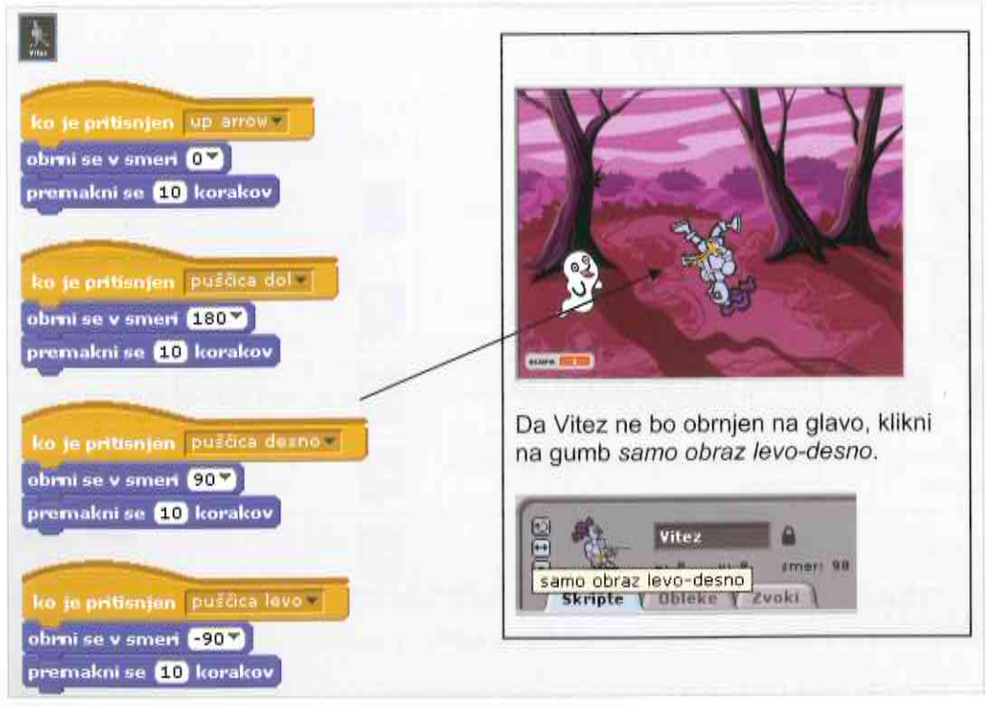
1. naredila bova figuro **Vitez** in določila njeno gibanje;
2. naredila bova figuro **Duhec**, ki se bo naključno premikala po prostoru;
3. uvedla bova spremenljivko, ki bo štela, **kolikokrat vitez ulovi Duhca**.

VITEZ

Vitez je glavni junak igre. S puščicami na tipkovnici ga lahko vodiš v vse smeri. Uvozi novo figuro. V mapi Domišljija (Fantasy) poišči viteza (knight1). Figuro poimenuj Vitez. Na začetku igre se Vitez pojavi na sredini odra, obrnjen v desno. Viteza pomanjšaj na 50 % njegove velikosti.



Kako upravljati figuro v vse smeri s tipkami na tipkovnici, že znava. Kar loti se dela.



Da Vitez ne bo obrnjen na glavo, klikni na gumb *samo obraz levo-desno*.

DUHEC

Uvozi novo figuro iz datoteke. V mapi Domišljija (Fantasy) poišči duhca (ghost2-a). Figuro poimenuj Duhec. Kot veš, ima figura lahko različne pojavne podobe, ki jim rečemo obleke. Duhec bo imel dve obleki, prijazno in strašljivo. Klikni na jeziček *Obleke* in obleko ghost2-a preimenuj v prijazen.

Izbriši ghost2-a in napiši *prijazen*.

Sedaj klikni na gumb *Uvozi* 

in v mapi Domišljija (Fantasy) poišči še enega duhca (ghost2-b). Obleko poimenuj *strašljiv*. Dobila sva obleki *prijazen* in *strašljiv*.

Izberi jeziček *Skripte* in napiši program za naključno gibanje Duhca, ki naj bo podobno gibanju ribic v nalogi Akvarij.

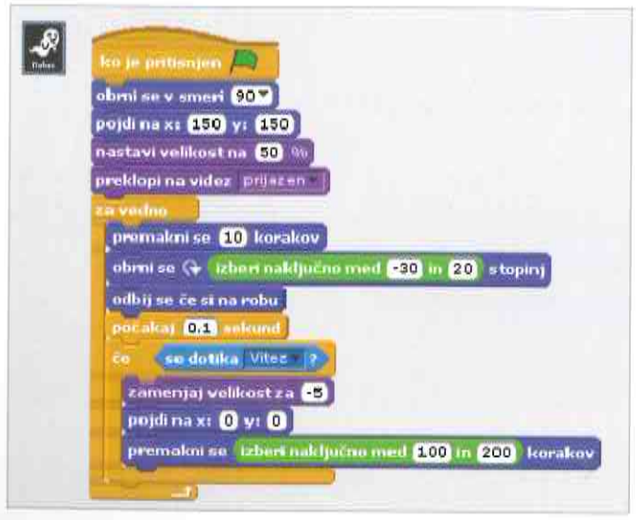
Dodajva pogoj, ki preverja, ali se Duhec dotika Viteza. Če je pogoj izpolnjen:

- se Duhec pomanjša (na primer za 5 enot),
- Duhec skoči stran: premakne se na sredino odra in naredi naključno velik premik med 100 in 200 korakov.

Napišiva še, kako se Duhec pojavi na odru na začetku igre:

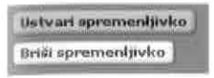
- na začetku naj bo odmaknjen od Viteza, na primer v desnem zgornjem kotu,
- obrnjen v desno,
- pomanjšan na 50 %,
- izbran naj bo videz prijazen.

Združi vse tri ukazne bloke in igro preizkusi.



KOLIKOKRAT JE VITEZ ŽE UJEL DUHCA

Igra je doslej zelo podobna prejšnji. Vendar pa bo ta igra imela tudi konec. Zaključila se bo, ko bo Vitez 10-krat ujel Duhca. Da bova vedela, kdaj končati igro, bova uvedla spremenljivko, ki bo štela, kolikokrat je Vitez ujel Duhca. Izberi skupino *Spremenljivke* in klikni na gumb *Ustvari spremenljivko*.

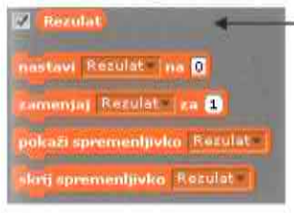


Odpre se pogovorno okno, kamor vpišeš ime spremenljivke, na primer Rezultat. Ime spremenljivke je smiselno izbrati glede na to, za kaj jo potrebuješ. Ime spremenljivke je lahko napisano z malo ali z veliko začetnico. Načeloma ga bova pisala z malo, le v primeru, da se bo prikazalo na odru, pa z veliko.

Kadar sva prepričana, da bo spremenljivko uporabljala le določena figura, izberiva možnost *Samo za ta sprite*. V tem primeru morava predvideti, katera figura bo uporabljala spremenljivko.

Če izbereva *Za vse sprite*, spremenljivko lahko uporabljajo vse figure. Tako ni treba vsega predvideti vnaprej, vendar pa se pri velikem številu spremenljivk zmanjša preglednost.

Izberi možnost *Za vse sprite* in klikni *V redu*. Spremenljivka je narejena in prikažejo se ukazi, ki so vezani na to spremenljivko.



Če odkljukaš potrditveno polje, se vrednost spremenljivke pokaže na odru.

Spremenljivko morava zdaj le še pravilno uporabiti:

- spremenljivka mora imeti na začetku vedno neko vrednost. V najinem primeru 0. (Duhec se še ni dotaknil Viteza.);
- če se Duhec dotakne Viteza, se vrednost spremenljivke poveča za 1.

Ko je vrednost spremenljivke Rezultat enaka 10, se igra konča:

- Duhec se poveča na 100 %,
- videz zamenja za strašljiv,
- obrne se proti Vitezu in
- predvaja zelo glasen in strašen buu in to pove tudi v oblačku;
- konča se tudi izvajanje vseh programov, kar omogoča ukaz *končaj vse*.

Glasen in strašen buu morava še posneti. Izberi jeziček *Zvoki* in klikni na gumb *Snemaj*.

Odpre se pogovorno okno Snemalnik zvoka. Klikni na rdeč gumb in na ves glas zavpij: »Buu.«

Klikni V redu.

Izpiše se ime *snemam1*. Preimenuj ga v Buu. Če tvoj računalnik nima mikrofona, lahko zvok uvoziš, podobno kot v igri Akvarij, ali pa na svetovnem spletu poiščeš kakšen strašen zvok.

Zdaj lahko sestaviva še končni ukazni blok.

Igri dodaj ozadje. Primerno ozadje najdeš lahko v mapi Narava (Nature), na primer gozd (woods). V ozadju se lahko sliši tudi strašljiva glasba. Poišči jo.

RAČUNALNIŠKI MAČEK

Naloga: Naredi podobno igro, kot je Začarani gozd. V njej naj kuža lovi kost. (Figure lahko uvoziš.) Kuža naj vsakič, ko mu kost uide, dvakrat zalaja. Ko kuža petkrat ujame kost, se ta poveča na začetno velikost, nato pa razpade. Pomagaj si z ukazom iz skupine *Izgled*

LOVEC METULJEV

OPIS IGRE

Igre, v katerih z neba padajo predmeti, glavni junak pa jih mora uloviti, preden padejo na tla, so precej pogoste. Z neba lahko padajo tudi predmeti, ki so nevarni in se jim mora glavni junak izogibati. Poskusiva narediti takšno igro, ki jo bova poimenovala Lovec metuljev. V njej se maček trudi, da bi ulovil čim več metuljev, ki padajo z neba.

NAČRT

Elementi igre	Dejanje	Program napišemo elementu
Praskač Praskač	- S puščicama ←, → ga premikamo levo in desno. - Če pritisnemo puščico ↑, skoči.	
Metulj Metulj	Navpična pada.	
Konec Konec	Skrit čaka na sporočilo, kdaj naj se igra konča. Takrat se pokaže in igro zaključi.	
Praskač in Metulj	Če se Metulj dotakne Praskačeve mreže, se skrije, nato se pojavi kot nov Metulj.	
Metulj in Konec	Če se Metulj dotakne tal, pošlje figuri Konec sporočilo, naj se igra konča.	Metulj in Konec
Spremenljivka Rezultat	Šteje ujete metulje.	
Cilj igre	- Preprečiti metulju, da pade na tla. - Ujeti čim več metuljev.	

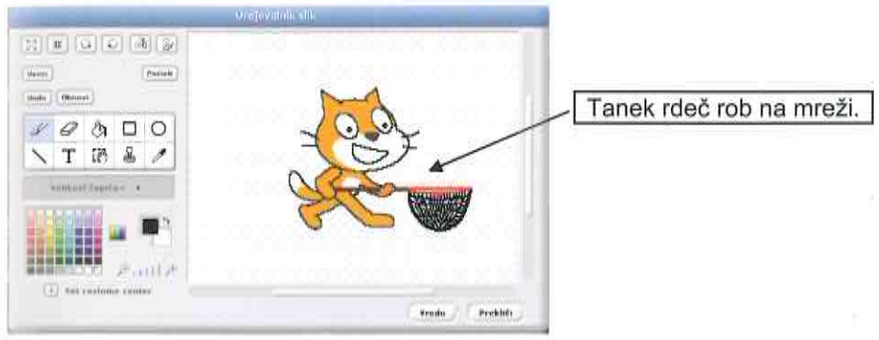
Programiranje igre bova razdelila na naslednje korake:

1. **Praskaču** bova narisala mrežo za metulje in določila njegovo premikanje;
2. naredila bova figuro **Metulj**, ki pada z neba;
3. dodala bova napis, ki bo označil **Konec igre** (bolj znano ti verjetno zveni Game Over).

PRASKAČ

Glavno vlogo v tej igri bom prevzel kar sam, saj me je lovljenje metuljev vedno zabavalo. Prikazal sem se že ob zagonu programa Scratch, zdaj me moraš le še poimenovati. Torej Praskač. S puščicami na tipkovnici me premikaš levo in desno, lahko pa tudi skačem.

Kot veš, imam že dve različni obleki, potrebujem le še mrežo. Klikni na jeziček *Obleke*. Izberi prvo obleko, klikni na gumb *Uredi* in v urejevalniku slik nariši mrežo za metulje. Rob mreže naj bo rdeče barve. Ta bo pomagala zaznati dotik metulja.



Enako mrežo nariši tudi drugi obleki.

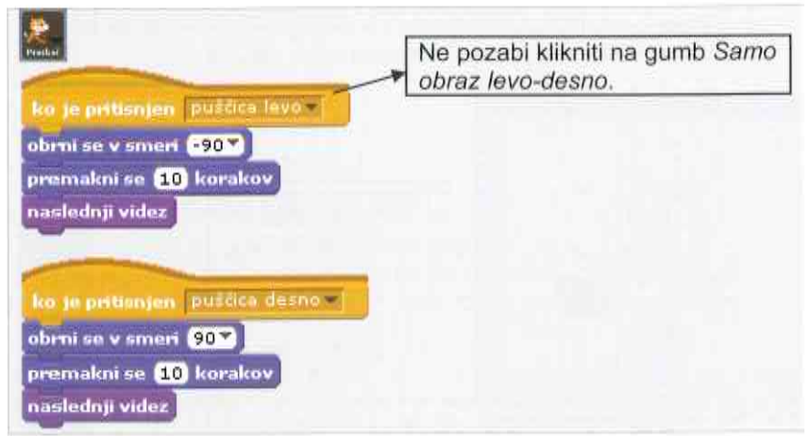
Če me postaviš na dno odra, bom imel več časa, da ulovim padajočega metulja.

S kazalcem miške me zato premakni na dno odra in odčitaj koordinati x in y.

Začetni koordinati: x = 0, y = –100.

Na začetku me postavi na določeno izhodišče. Lahko me tudi pomanjšaš.

Napiši program, s katerim me lahko vodiš levo in desno s puščicami na tipkovnici. Če uporabiš ukaz *naslednji videz*, bom med hojo menjal obleki. Poskusi.



Ne pozabi klikniti na gumb *Samo obraz levo-desno*.

Da bom lažje ulovil metulja, mi moraš omogočiti, da bom lahko tudi skakal. To narediš s pritiskom na *puščico gor*. Napiši ukazni blok. Ko skočim:

- se za nekaj korakov premaknem navzgor,
- naredim kratek premor in
- se spustim.

Preizkusi, kako skačem. Tole je bilo pa malo prehitro. Poskusiva drugače. Na primer tako:

METULJ

Metulj se pojavi na vrhu odra in navpično pada. Če ga ulovim v svojo mrežo, se Metulj skrije, nato pa se ponovno pojavi na vrhu, kot nov Metulj, ki ponovno pada proti tlom. Če se Metulj dotakne tal, se igra konča. Uvozi novo figuro. V mapi Živali (Animals) izberi enega od metuljev in ga poimenuj Metulj.

Metulja najprej pomanjšaj ter določi, kje naj se pojavi na začetku:

- koordinata y naj bo določena nekje na vrhu odra,
- koordinata x pa naj zavzema naključne vrednosti med levim in desnim koncem odra (približno med 200 in -200).

Dodaj ukazni blok za Metuljevo padanje. Navpično padanje figure na odru pomeni neprestano spreminjanje koordinate y. Sprememba koordinate y naj bo majhna, saj bo drugače Metulj padal prehitro.

Vsakič, ko se Metulj spusti nekoliko nižje, je potrebno preveriti dva pogoja:

- ali se je Metulj dotaknil mreže in
- ali se je Metulj dotaknil tal,

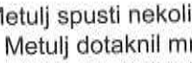
saj se v tem primeru padanje prekine.

Sestavi ukazni blok za preverjanje prvega pogoja in razmisli, kaj se zgodi Metulju, če je pogoj izpolnjen:

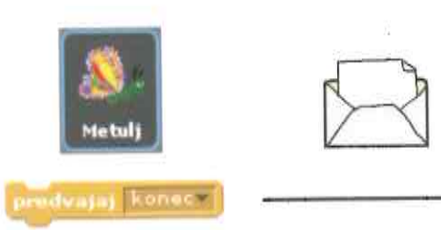
- Metulj se skrije,
- nato se premakne na vrh odra (kot na začetku igre) in
- se čez nekaj časa zopet prikaže.

Če pogoj vstaviva v zanko *za vedno*, Metulj začne ponovno padati. Ni težko ugotoviti, zakaj.

Če mi ne uspe uloviti Metulja, ta pade na tla in igra je končana. Zgornjemu ukaznemu bloku je tako treba dodati še pogoj, ki preverja, ali je Metulj padel na tla. Igra se konča tako, da se prikaže napis *Konec igre*. Ker bo napis narejen kot figura, morava določiti, kdaj naj se pojavi na odru.

Ukaza *predvajaj*  in *ko prejmem*  bosta omogočila, da se bodo dogodki odvijali po predvidenem vrstnem redu.

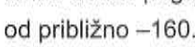
Ko se bo Metulj dotaknil tal, bo s pomočjo ukaza *predvajaj* poslal sporočilo, naj se igra konča. To sporočilo bo prejela figura z napisom *Konec igre* in se pojavila na odru. Vsakemu sporočilu morava izbrati tudi smiselno ime. V tem primeru je ustrezno ime *konec*.



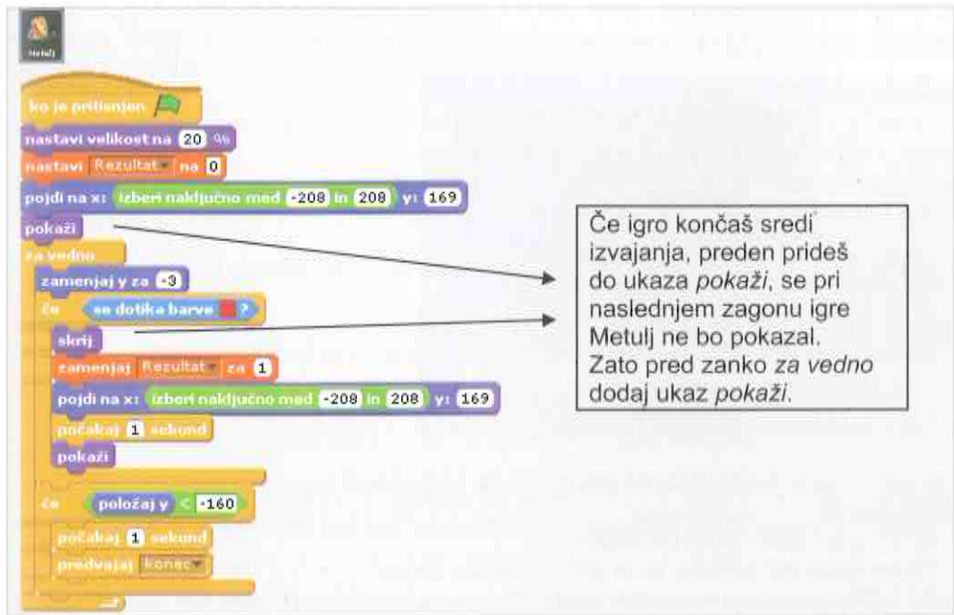
Ko Metulj pade na tla, pošlje sporočilo *konec*.

Napis se pokaže, ko figura Konec prejme sporočilo *konec*.

Naredi sporočilo z imenom *konec*. Klikni na ukaz *predvajaj* in v izbirnem oknu, ki se odpre, izberi novo.  Odpre se pogovorno okno, kamor napišeš ime sporočila.

Sporočilo *konec* lahko dodaš pogoju, ki preverja, ali je Metulj padel na tla oziroma če je položaj y manjši od približno –160. Uporabi ukaz  in sestavi pogoj.

Igra bo zanimivejša, če bova vedela, koliko metuljev mi je uspelo ujeti, preden se je igra končala. Zato narediva novo spremenljivko z imenom *Rezultat*, ki šteje ulovljene metulje. Kako jo uporabiti, zdaj gotovo že veš. Spremenljivko dodaj k ustreznim ukaznim blokom. Posamezne ukazne bloke lahko tudi združiš.



KONEC IGRE

Igra se konča tako, da se prikaže napis *Konec igre*. Napis bova izdelala sama, in sicer kot novo figuro. Klikni na gumb *Nariši novo skupino slik*. Odpre se pogovorno okno Urejevalnik slik. Izberi orodje *Text tool* in napiši *Konec igre*.

Izberi še orodje *Zapolni* in ozadje pobarvaj črno. Pri orodju *Zapolni* lahko izbiraš med različnimi možnostmi polnitve.  Pomembno je, da je pobarvan cel zaslon, saj bo tako figura prekrila ostale figure in ozadje. Figuro z napisom *Konec igre* imenuj Konec.

Sedaj sestavi ukazne bloke, ki bodo omogočili, da se bo napis prikazal v pravem trenutku. Pri tem upoštevaj:

- ko se igra začne, mora biti napis skrit;
- figura Konec se prikaže, ko prejme sporočilo *konec*;
- če želiš, da prekrije vse ostale figure, dodaj ukaz *pojdi v ospredje*.

Igri dodaj še ozadje. Primerno ozadje lahko najdeš v mapi Narava (Nature). Lahko ga tudi narišeš.

RAČUNALNIŠKI MAČEK

1. **naloga:** V igri Lovec metuljev dopolni zaključek igre. Preden se pojavi napis *Konec igre*, naj Metulj pove (v oblačku), koliko točk je bilo zbranih.

2. **naloga:** Naredi več različnih metuljev.

Namig: Metulji naj padajo počasneje. Dobro je, da se na začetku igre ne pojavilo vsi metulji hkrati. Ustrezno uporabi ukaz *počakaj*.

3. **naloga:** Praskaču podari več življenj.

Namig: Naredi še eno spremenljivko, ki bo štela Praskačeva preostala življenja. To pomeni, da se bo vrednost spremenljivke manjšala. Ko bo vrednost spremenljivke nič, se igra konča.

4. **naloga:** Naredi povsem novo igro z ekološko vsebino. Ekoloških iger na internetu še ni zaslediti. Mogoče bo pa tvoja prva.

Opis igre: Z neba padajo nevarni odpadki. Če te zadenejo, si izgubil. Vendar niso vsi odpadki nevarni, nekateri so tudi uporabni, ker se jih da koristno predelati. Če v mrežo uloviš takšne, dobiš točke.

Namig: Naredi več figur, ki predstavljajo različne vrste odpadkov: steklenice, papir, plastično embalažo, akumulatorje, baterije, katran …

5. naloga: V igri si lahko junak tudi ti. To lahko narediš zelo preprosto. Najprej moraš postati figura. Gotovo imaš kakšno svojo sliko na računalniku, lahko pa se tudi fotografiraš posebej za ta namen. Tako kot vsako figuro, tudi to narediš tako, da klikneš na gumb *Izberi novi sprite iz datoteke*. Nato poiščeš svojo sliko. Sliko je treba zdaj obdelati. Klikni na jeziček *Obleke* in izberi gumb *Uredi*. Odpre se Urejevalnik slik, kjer se prikaže tvoja slika. Izberi primerno povečavo , nato pa z orodjem Radirka naredi svojo figuro.

ČEBELICA MAJA

OPIS IGRE

Glavna junakinja igre je čebelica Maja, ki opraŠuje cvetje. Ko Maja opraši cvetlico, se ta razcveti. Ko zacvetijo vsi cvetovi, lahko Maja vstopi v naslednjo sobo. V njej mora prazne celice satovja napolniti z medom. Ko napolni vse celice, dobi nagrado, lonček medu.

NAČRT

Namesto preglednice si bova v tej igri raje podrobneje pogledala potek igre. Doslej so se igre začenjale s klikom na zeleno zastavico. Tokrat bo nekoliko drugače. Če klikneš na zeleno zastavico, se pokažejo navodila ter gumb za začetek igre. Ko klikneš nanj, se pokaže soba, v kateri čebela opraŠuje cvetice. Drugačno pri tej igri je tudi to, da nima le ene sobe.

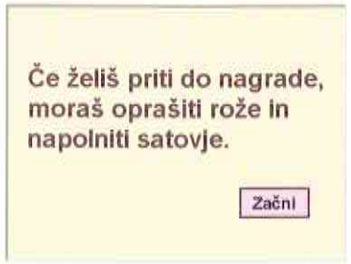
Naloge se bova lotila po korakih:

1. pogledala si bova podrobnejši **potek igre**;
2. narisala bova različna **ozadja**;
3. naredila bova figuro **Začni**, ki je gumb za začetek igre;
4. narisala bova figuro **Čebela**, ki se bo premikala s pomočjo miške;
5. naredila bova več figur **rož**, ki bodo menjale obleke,
6. narisala bova figuro **Soba2**, ki bo služila kot vhod v naslednjo sobo;
7. naredila bova več figur **celic medu**;
8. naredila bova še zaključek, nagrado, **lonček medu**.

POTEK IGRE

Igra bo imela štiri dejanja: navodilo, rože, satovje in nagrado. Pri določanju, kdaj se odvija posamezno dejanje, si bova pomagala s pošiljanjem sporočil in s spremenljivkami. Poglejva, kako bova tej igri pravilno določila potek.

Navodilo



S klikom na **zeleno zastavico** se prikažejo:

- navodila, ki so napisana na ozadju ter
- gumb Začni, ki je narejen kot figura.

Ko klikneš na gumb Začni, gre naprej sporočilo ***začetek***.

Prva soba

Ko prejmejo sporočilo ***začetek***, se prikažejo:

- tri rože v obleki popek,
- novo ozadje, travnik,
- gumb Naslednja soba, ki je figura.

Čebelica opršuje cvetlice, ki spremenijo obleko v cvet. Spremenljivka z imenom *rože* šteje, koliko cvetlic je zacvetelo.

Ko je vrednost spremenljivke enaka 3, lahko klikneš na figuro Naslednja soba, ki pošlje sporočilo ***naslednja soba***.

Druga soba

Ko prejmejo sporočilo ***naslednja soba***, se prikažejo:

- novo ozadje, satovje,
- tri celice satovja v obleki rjava.

Čebela polni prazne celice satovja, ki spremenijo obleko v rumeno. Spremenljivka z imenom *med* šteje, koliko celic se je napolnilo z medom.

Ko je vrednost spremenljivke enaka 3, gre naprej sporočilo ***nagrada***.

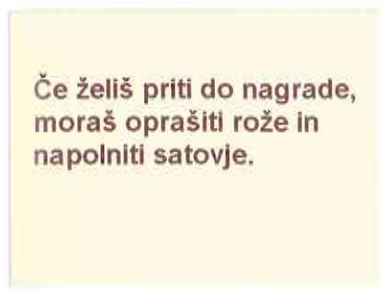
Nagrada

Ko prejme sporočilo ***nagrada***, se prikaže lonec medu in igra je končana.

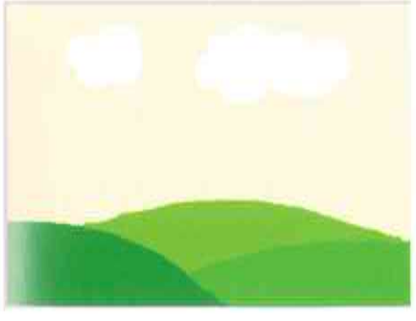
OZADJA

Klikni na ikono *Ozadje* (*Stage*) in izberi jeziček *Ozadja*. Narisala bova tri različna ozadja. Izberi gumb *Risar* 

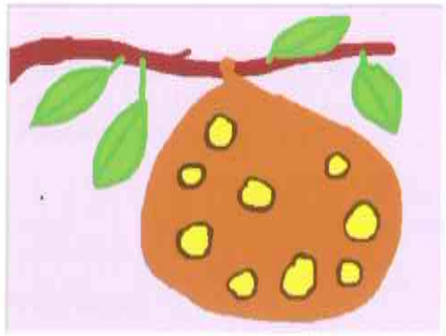
in odprl se bo Urejevalnik slik. Na prvem ozadju bodo napisana navodila.



Zopet klikni na gumb *Risar* in nariši drugo ozadje. Na drugem ozadju bo narisana scena za prvo sobo: travnik in nebo z oblaki.



Nariši še tretje ozadje, kjer bo narisano satovje. Na satovju nariši nekaj celic medu, vendar pusti dovolj prostora za celice medu, figure, ki jih lahko dodaš kasneje.



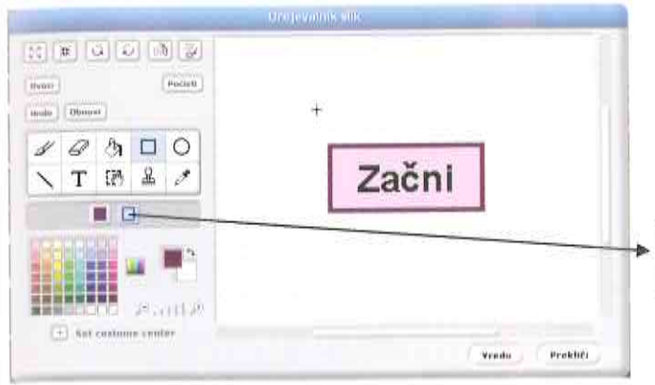
Ozadja poimenuj: navodila, travnik in satovje.

Ker potek igre poznava, lahko program za ozadje napiševa kar v celoti. Iz poteka je razvidno, da se:

- prvo ozadje, navodila, pojavi s klikom na *zeleno zastavico*,
- drugo ozadje, travnik, se pojavi, ko prejme sporočilo *začetek* in
- tretje ozadje, satovje, se pojavi, ko prejme sporočilo *satovje*.

ZAČNI

Hkrati z navodili se na začetku igre pokaže tudi gumb z napisom *Začni*. Narejen je kot figura, saj tako zazna, kdaj sva nanjo kliknila s kazalcem miške. Klikni na gumb *Nariši novo skupino slik*. Odpre se pogovorno okno *Urejevalnik slik*. Gumb narišeš s pomočjo *Orodja za risanje pravokotnikov*. Izberi še orodje Besedilo (*Text tool*) in v pravokotniku napiši Začni. Figuro imenuj kar Začni.



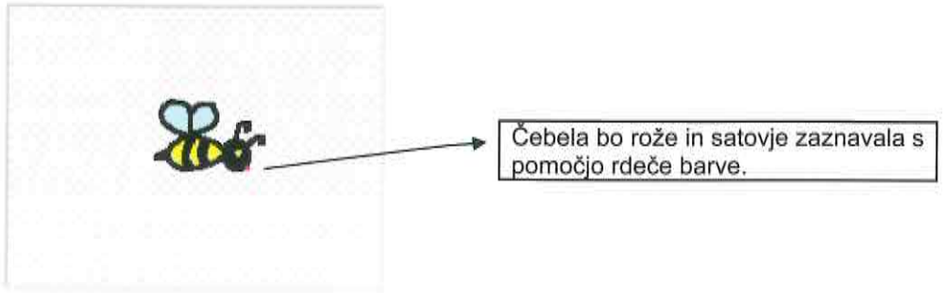
Gumb *Začni* nato s pomočjo miške premakni na primerno mesto pod besedilo. Ker se med igro ne bo premikal, mu v programu ni treba določiti začetne lege. Če ozadje navodila ni prikazano, klikni na zeleno zastavico.

S klikom na gumb *Začni* se igra začne. Razmisliva, kako bova napisala program. Gumb *Začni*:

- se pokaže takoj na začetku, ko klikneš na zeleno zastavico,
- nato čaka, dokler nanj ne klikneš z miško,
- takrat pošlje naprej sporočilo *začetek* in
- se skrije.

ČEBELICA MAJA

Čebelica Maja je glavna junakinja igre. V prvi sobi opršuje cvetlice, v drugi pa polni satovje z medom. Narisala jo bova kot novo figuro v Urejevalniku slik in poimenovala Čebela.



Doslej sva junake vodila s tipkami na tipkovnici, Čebelo pa bova vodila s pomočjo miške in se tako naučila še drugačnega načina vodenja figur. Uporabiva ukaz obrni se proti kazalec miške in napišiva program za vodenje čebele. Pri tem upoštevaj:

- Čebela se vedno obrača v smeri kazalca miške;
- vsakič, ko se obrne, naredi tudi nekaj korakov.

Opaziva, da se pojavi težava, če je kazalec miške točno na sredini figure. Problem rešiš tako, da dodaš pogoj - čebela sledi kazalcu miške le, če je od njega nekoliko oddaljena, na primer za 5 enot razdalja do kazalec miške > 5 .

Narediva dve spremenljivki, ki bosta šteli, koliko rož je oprašenih in koliko celic z medom je napolnjenih. Prvo imenujva rože, drugo pa med. Da bodo spremenljivko lahko uporabljale vse figure, izberiva možnost *Za vse sprite*. Na začetku naj bo vrednost obeh spremenljivk enaka 0. Ukaza, ki nastavita vrednost spremenljivke na 0, bova dodala enemu od ukaznih blokov Čebele, čeprav bi ju lahko dodala tudi kje drugje. Pomembno je le, da sta vrednosti spremenljivk, ko se igra prične, nastavljeni na 0.

Tudi Čebeli morava določiti, kdaj se pojavi na odru:

- na začetku je Čebela skrita.
- pokaže se, ko prejme sporočilo »začetek«.
- Čebela je na odru postavljena pred vse figure, torej v ospredje (pred cvetlice ali celice medu).

Že napisanim ukaznim blokom dodaj še naslednje.

ROŽE

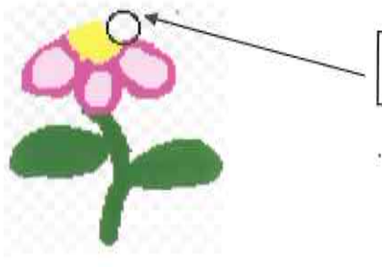
Čeprav bo rož več, je dovolj, da na začetku narišeš le eno figuro rože. V *Urejevalniku slik* nariši rožo in jo poimenuj Roža1.

Roža bo imela dve obleki, cvet in popek. Izberi jeziček *Obleke* in obleko1 preimenuj v popek. Obleko nato prekopiraj. Prekopirano obleko preimenuj v cvet.

Jeziček *Obleke*.

Gumb *Kopiraj*.

Klikni na gumb *Uredi* in v *Urejevalniku slik* nariši novo obleko.



Z radirko zbriši del slike in nariši popek.

Roža ima tako dve obleki, ki ju poimenujva *cvet* in *popek*.

Lotiva se programa:

- Roža je na začetku igre skrita;
- ko prejme sporočilo *začetek*, se pokaže v obleki popek;
- Roža čaka, dokler se je ne dotakne Čebela;
- nato Roža zamenja obleko *popek* v obleko *cvet*;
- vrednost spremenljivke *rože* se poveča za 1;
- Roža je v naslednji sobi zopet skrita.

Napiši ukazne bloke. Upoštevaj, da Roža zazna dotik Čebele s pomočjo barv. Rdeča pika na glavi Čebele se bo dotaknila rumene barve popka. Pomagaj si z ukazom

Ko je program napisan, Rožo dvakrat podvojiva. Rože nato s pomočjo miške razporediva po travniku. Če ni prikazano ozadje travnik, klikni na ikono *Ozadja (Stage)*, izberi jeziček *Ozadja* in klikni na ozadje travnik. Ker se rože med igro ne bodo premikale, jim v programu ni treba določiti začetne lege.

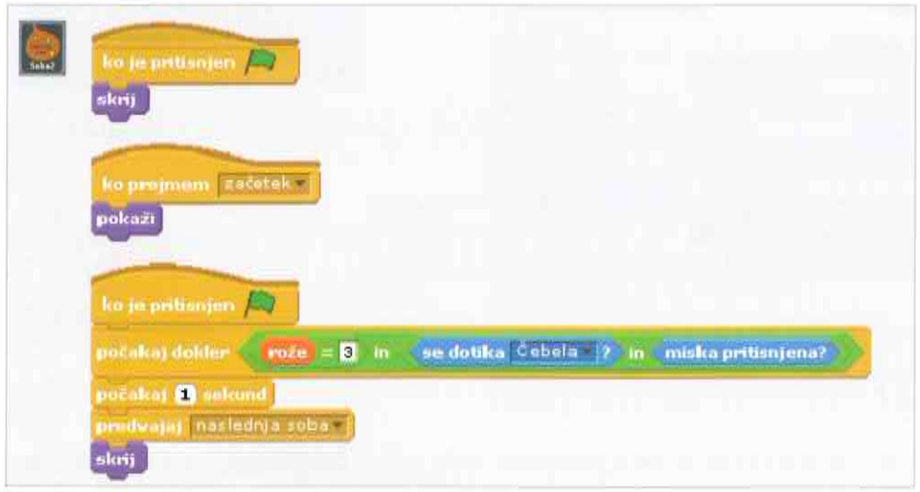
NASLEDNJA SOBA

Če želiva priti v drugo sobo, morava klikniti na gumb z napisom *Naslednja soba*. Tudi ta gumb bova naredila kot figuro v Urejevalniku slik. Imenujva ga Soba2. Gumb s kazalcem miške premakniva na primerno izhodišče, nekje na robu odra.

Gumb je na začetku skrit in se pokaže, ko prejme sporočilo *začetek*. Naloga gumba je, da pošlje naprej sporočilo »naslednja soba«. To se zgodi, ko so izpolnjeni trije pogoji:

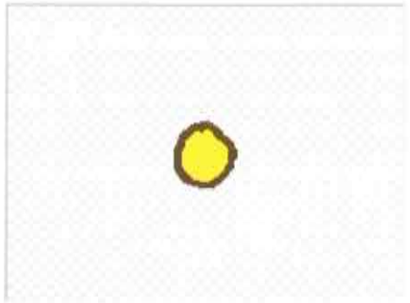
- Čebela je oprašila vse tri rože, torej vrednost spremenljivke rože je enaka 3,
- Čebela se dotika gumba in
- pritisnjen je kazalec miške.

Da bova lahko preverila, ali so vsi trije pogoji izpolnjeni, morava ukaze sestaviti:



CELICE

V drugi sobi bodo na satovju poleg celic, ki so že polne medu, tudi take, ki jih bo treba še napolniti. Te bova naredila kot figure in bodo imele podobno vlogo kot rože. V *Urejevalniku slik* narišiva prvo celico in jo imenujva Celica1.



Podobno kot rože bodo imele tudi celice dve obleki: rjavo in rumeno.

Tudi program bo podoben kot pri rožah, zato podrobnosti ni treba še enkrat razlagati.

Potrebujeva tri celice, zato Celico dvakrat podvojiva. Celice nato razporediva po satovju. Ker se celice med igro ne bodo premikale, jim v programu ni treba določiti začetne lege. Če ozadje satovje ni prikazano, klikni na ikono *Ozadje (Stage)*, izberi jeziček *Ozadja* in klikni na satovje.

Spremenljivka z imenom *med* šteje, koliko celic se je napolnilo z medom. Ko je vrednost spremenljivke enaka 3, gre naprej sporočilo *nagrada*. Ukazni blok ni vezan na določeno figuro. Dodala ga bova k Čebeli.

MED

Nariši še zadnjo figuro, lonček medu in jo poimenuj Med. Ta ima podobno vlogo kot figura z napisom *Konec igre* v igri Lovec metuljev. Podoben je tudi program.

Zdaj mora čebelica Maja le še izpolniti svoje naloge in na koncu jo čaka nagrada.

RAČUNALNIŠKI MAČEK

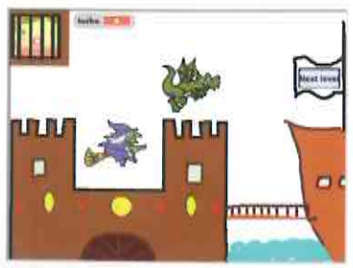
1. naloga: Lonček medu se v igri Čebelica Maja prikaže, ko prejme sporočilo *Konec igre*. To se da narediti tudi drugače, brez pošiljanja sporočila. Bi šlo s spremenljivko?

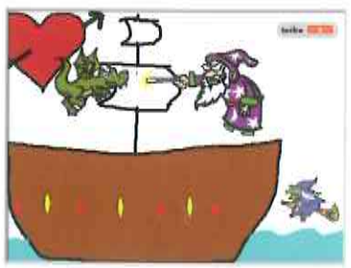
2. naloga: Izberi eno od že narejenih iger in ji dodaj navodila.

3. naloga: Igri naredi navodila brez uporabe figure Začni.
Namig - 1. možnost: Navodila se pokažejo za določen čas, nato pa se igra prične in pokaže se prva soba. Vsi ukazni bloki, ki so se prej zagnali, ko so prejeli sporočilo *Začetek*, se bodo zdaj zagnali že takoj na začetku, s klikom na zeleno zastavico. Zato jim bo takoj za ukazom potrebno dodati še en ukaz. Kateri?

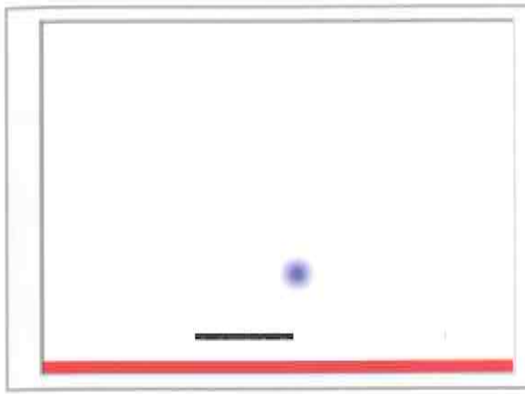
Namig - 2. možnost: Gumb *Začni* je lahko del ozadja, zaznaš ga s pomočjo barve.

4. Naloga: Uporabi domišljijo in naredi zanimivo zgodbo z več različnimi sobami. Potrebuješ idejo?





PONG



OPIS IGRE

Pong je igra, ki ponazarja namizni tenis. Cilj igre je s pomočjo loparja preprečiti, da bi žoga padla na tla.

V Wikipediji si o igri pong lahko prebereš naslednje:
»Pong je bila prva komercialno uspešna videoigra. Najbolj znano različico je leta 1972 izdalo podjetje Atari. Igra je bila simulacija namiznega tenisa, ki se angleško pogosto imenuje pingpong. Igra je bila na voljo kot samostojna naprava, priključena na televizor ali pa v obliki igralnih videoavtomatov (običajno na kovance) ...«

Skupaj s programom Scratch se naložijo tudi različni primeri iger in animacij. Najdeš jih v mapi Primeri (Examples). Ena od takih iger je tudi igra pong.

V menijski vrstici klikni na Datoteka (File) in izberi Odpri. V pogovornem oknu, ki se odpre, izberi mapo Primeri (Examples) in nato mapo Igre (Games). V mapi Igre izberi datoteko Pong in jo odpri.

Klikni na *zeleno zastavico* in igro zaženi. Igro najprej preizkusi, nato pa skušaj napraviti še svojo različico. Zato odpri nov projekt. V menijski vrstici izberi Datoteka (File) in nato Nova.

NAČRT

Elementi igre	Dejanje	Program napišemo elementu
Ozadje	Je kulisa, rdeča črta ponazarja dno.	
Lopar	Upravljamo ga z miško. Premika se levo in desno.	
Žoga	Se premika sama od sebe naravnost naprej, na robu se odbije.	
in	Če se žoga dotakne loparja, se odbije.	
in	Če se žoga dotakne rdeče črte, se igra konča.	
Cilj igre	Preprečiti, da bi žoga padla na tla.	

Programiranje igre bova razdelila na tri korake:

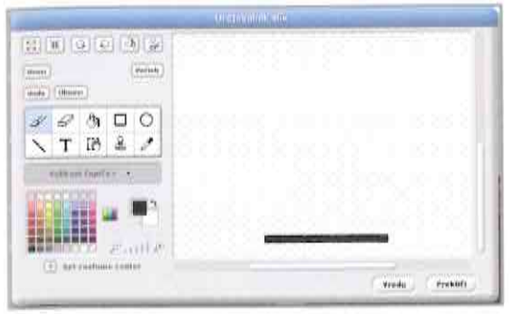
1. narisala bova **ozadje**;
2. naredila bova figuro **Lopar**, ki bo žogi preprečila, da pade na tla;
3. naredila bova figuro **Žoga**, ki se odbija od vseh površin, razen od rdeče.

OZADJE

V Urejevalniku slik nariši ozadje, na katerem je rdeča črta.

LOPAR

V urejevalniku slik nariši novo figuro, ki ponazarja lopar in jo poimenuj Lopar. Figuro Praskača izbriši.



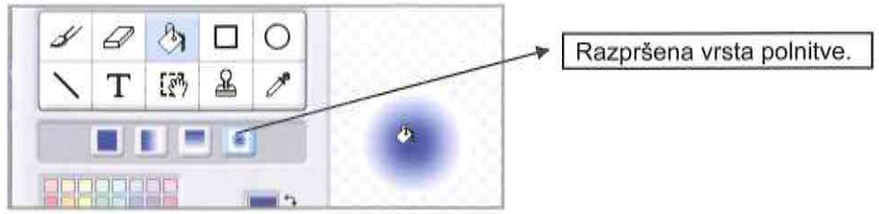
Lopar se premika levo in desno in preprečuje žogi, da bi padla na tla. Upravljaš ga z miško. Kako upravljati figuro z miško, sva spoznala v igri Čebelica Maja. Lopar se bo premikal le v vodoravni smeri. Ves čas naj se položaj loparja v koordinati x ujema s položajem miške. To bosta omogočila ukaza nastavi x na in miska x.

S kazalcem miške določiva še začetno lego Loparja nekje na dnu odra.

ŽOGA

V urejevalniku slik nariši figuro, ki bo ponazarjala žogo, in jo poimenuj Žoga. Uporabi orodje Elipsa in nariši moder krog.

Nato izberi orodje *Zapolni*, izberi ustrezno polnitev in klikni v središče kroga.

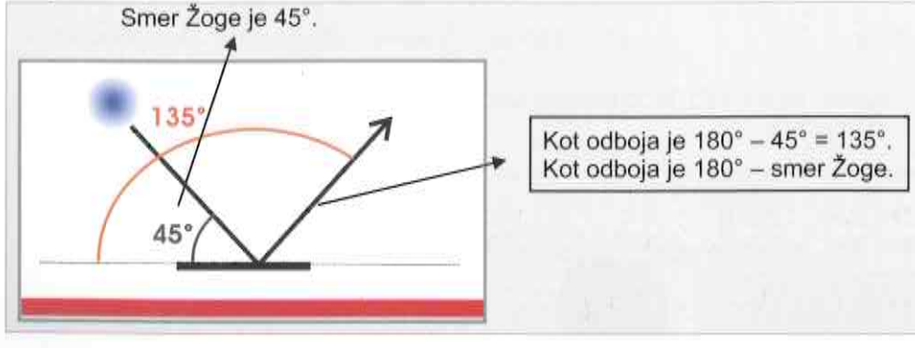


Žoga se ves čas igre premika po odru. Premika se naravnost naprej, dokler ne pride do roba ali do Loparja in se odbije. Če pade na tla, je igra končana. Gibanje Žoge je neprestano, podobno kot pri ribah v igri Akvarij. Vendar pa gibanje ni naključno.

Določiti morava:

- začetno lego Žoge nekje na vrhu odra,
- začetno smer, poševno navzdol (če bi se Žoga začela premikati navpično navzdol, bi se po odboju vračala po isti poti) in
- hitrost premikanja.

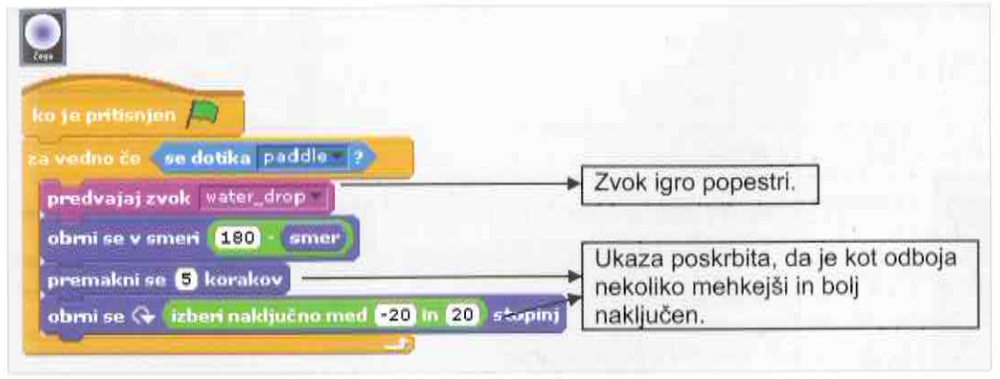
Ukazni blok dopolniva tako, da se bo žoga odbijala tudi od loparja. Kot, pod katerim se mora žoga odbiti, bova določila s pomočjo primera na spodnji sliki.



S pomočjo ukaza smer, ki poda smer žoge, napišiva še ukazni blok.

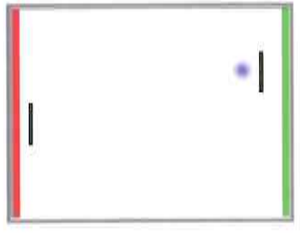
Igri narediva še zaključek. Ko Žoga pade na tla oziroma, ko se dotakne rdeče barve, je igra končana.

Če želiš, si lahko še enkrat pogledaš igro Pong iz mape Primeri. Kot vidiš, ima ukazni blok za gibanje Žoge dodanih še nekaj ukazov.



RAČUNALNIŠKI MAČEK

Naloga: Naredi podobno igro, ki pa naj ima dva igralca. Prvi premika lopar z miško, drugi s tipkovnico, na primer s črkama S in D. Igro izgubi tisti, ki ni ubranil žoge. Vsak od igralcev ima lahko tudi več življenj. Pomagaj si s sliko.



PAC-MAN

OPIS IGRE

Igralec upravlja Pac-Mana, ki se premika po labirintu in žre krogce.

Pac-Man je računalniška igra, ki so jo razvili leta 1980. Sodi med najuspešnejše dvodimenzionalne igre vseh časov. Igra ima 50 sob, vendar ni znano, da bi kdorkoli uspešno prišel do konca. Najina igra je poenostavljena verzija originalnega Pac-Mana, ki ga pri premikanju ovirajo pošasti ali duhovi. Tudi Igro Pac-Man lahko najdeš v Scratchevi mapi Primeri (Examples).

NAČRT

Elementi igre	Dejanje	Program napišemo elementu
Labirint	Labirint določa prostor, po katerem se Pac-Man lahko premika.	
Pac-Man	- Premika se naravnost naprej. - S puščicami na tipkovnici ←,↑,→,↓ mu določaš smer.	
Krogci	Več belih krogcev razporedimo po labirintu. Vsak krogec je ena figura.	
in	Pac-Man zaznava barve labirinta. Premika se lahko le po črni barvi.	
in	Ko se Pac-Man dotakne krogca, ta izgine.	
Cilj igre	Pac-Man požre vse krogce.	

Programiranje igre bova razdelila na štiri korake:

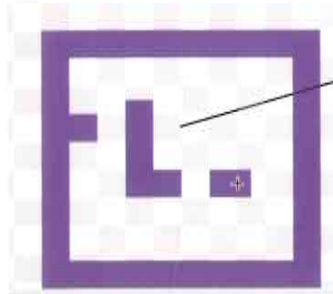
1. na ozadje bova narisala **labirint**;
2. naredila bova figuro **Pac-Mana** in določila njeno gibanje;
3. naredila bova več belih **krogcev**;
4. dodala bova **zvok**.

LABIRINT

Pac-Man se bo premikal po črnih hodnikih labirinta. Labirint bova narisala na ozadje. Klikni na ikono *Ozadje* (*Stage*) in izberi jeziček *Ozadja*. Izberi gumb *Risar*

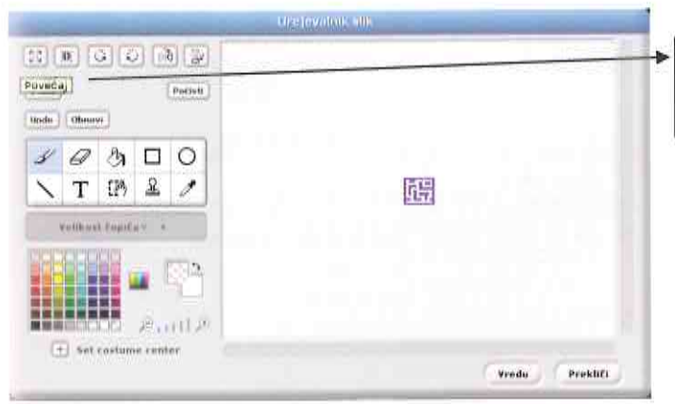
in odprl se bo Urejevalnik slik.

Labirint je lažje narisati tako, da najprej izbereš večjo povečavo in s pomočjo orodja *Črta* ali z *Orodjem za risanje pravokotnikov* narišeš stene labirinta.



Pri večji povečavi lahko s pomočjo kvadratkov narediva načrt labirinta.

Nato izbereš najmanjšo povečavo . Prikaže se dejanska velikost labirinta.

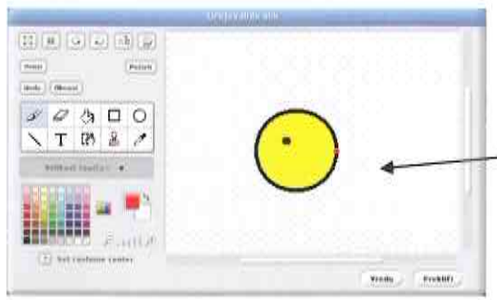


Labirint povečaš tako, da izbereš orodje *Povečaj* in večkrat klikneš na labirint.

Sliko še pobarvaj in dopolni z napisom.

PAC-MAN

V *Urejevalniku slik* nariši figuro Pac-Man in jo imenuj PacMan. Imel bo dve obleki: z odprtimi in zaprtimi usti. Najprej narišiva Pac-Mana z zaprtimi usti.



Rdeča pika bo pomagala pri določanju meje labirinta. Če je pika premajhna, se lahko pojavijo težave pri zaznavanju barve.

Narediva še obleko z odprtimi usti. Izberi jeziček Obleke in klikni na gumb Kopiraj.

Nastane enaka obleka, ki jo bo treba preoblikovati. Izberi obleko2 in klikni na gumb *Uredi*. Z orodjem *Črta* določi usta. Nato z radirko izbriši lok med črtama.

Izberi prozorno barvo in z orodjem *Zapolni* izbriši preostanek rumene barve. Ne smeš pozabiti na rdečo piko.

Pac-Man se bo samodejno premikal naravnost naprej, s puščicami na tipkovnici ga bova le usmerjala, če bo prišel do ovire.

Na začetku igre naj bo Pac-Man:

- približno na sredini Labirinta, na delu, ki je obarvan črno,
- postavljen v ospredje (da ga ne prekrije Labirint),
- primerno pomanjšan in
- neprestano naj odpira in zapira usta.

Tudi premikanje Pac-Mana se začne takoj na začetku igre. Pac-Man se premika ves čas, razen če naleti na oviro, vijolično barvo Labirinta. Oviro bo prepoznal s pomočjo zaznavanja barve.

Tukaj nama bo prišla prav rdeča pika; dokler se rdeča pika dotika črne barve labirinta, se Pac-Man počasi premika naprej. Ko se rdeča pika dotakne vijolične barve, se ustavi.

Napišiva še ukazne bloke za spreminjanje smeri s pomočjo puščic na tipkovnici.

KROGEC

Pac-Man v igri žre bele krogce. Ker bodo krogci med igro izginjali, jih je treba narediti kot figure. Naredila bova en krogec, mu napisala program in ga nato večkrat podvojila.

Nariši novo figuro, bel krogec. Poimenuj jo Krogec1. Program je enostaven. Na začetku ima Krogec1:

- določeno lego (določiva jo s pomočjo kazalca miške) in
- je viden.

Ko se Pac-Man dotakne krogca, ta izgine. Ob tem naj se sliši zvok hrustanja. Klikni na jeziček *Zvoki* in iz mape Efekti (Effects) uvozi primeren efekt, na primer zvok *Hrsk* (Rattle).

Ko je Krogec1 dokončan, ga večkrat podvojiva. Nove krogce s pomočjo kazalca miške razporediva po Labirintu.

ZVOK V OZADJU

V ozadju naj se sliši glasba. Postavi se v ozadje, klikni na jeziček *Zvoki* in v mapi Napevi (Music Loops) poišči zvok *DripDrop*. Napiši program, ki bo ves čas igre predvajal zvok.

Napiši še spodnji program in se odloči, katera verzija predvajanja zvoka ti je bolj všeč.

Naredila sva poenostavljeno verzijo Pac-Mana. Igro lahko na različne načine dopolniva. Nekaj idej, kako jo izboljšati, te čaka pri nalogah.

RAČUNALNIŠKI MAČEK

1. naloga: Igri dodaj pošast, ki se premika levo in desno na enem od hodnikov in tako Pac-manu oteži žretje krogcev, saj je nevarna. Odločitev, kako nevarna je, je tvoja.

2.naloga: V originalni igri Pac-Man se pošasti premikajo proti glavnemu junaku, vendar je napisati ustrezen program že pravi izziv. Vredno se je potruditi.

3. naloga: Igri dodaj več sob.

Namig: Naredi spremenljivko, ki šteje požrte krogce. Ko so vsi krogci požrti, se oblika labirinta spremeni (nariši več različnih ozadij), krogci pa se zopet prikažejo in razporedijo po Labirintu. Če narediš več sob, je treba krogcem v programu določiti začetno lego.

NORA DIRKA

OPIS IGRE

Igralec vodi avto po dirkalni stezi in skuša čim prej priti na cilj.

NAČRT

Elementi igre	Dejanje	Program napišemo elementu
Stage Ozadje	Je kulisa. Siva barva določa cesto.	
Avto Avto	- Premika se naprej, hitrost določa spremenljivka. - Obračaš ga s puščicami na tipkovnici ← in →.	
Cilj Cilj	S pomočjo štoparice pove čas vožnje.	
Avto in Cilj	Cilj zazna dotik avta, štoparica se ustavi.	
Spremenljivka hitrost	Določa hitrost vožnje: - se poveča, če pritisneš na↑, - se zmanjša, če avto zapelje na travnik.	

Programiranje igre bova razdelila na tri korake:

1. narisala bova **dirkalno stezo**;
2. naredila bova figuro **Avto** in določila njeno vodenje;
3. naredila bova figuro **Cilj**, na kateri se bo prikazal čas vožnje.

DIRKALNA STEZA

Dirkalno stezo bova narisala na ozadje. Najprej s *Čopičem* nariši temno sive, tanke robove ceste. Nato z orodjem *Zapolni* cesto pobarvaj svetlo sivo. Ostalo površino pobarvaj zeleno in nariši belo štartno črto. Lahko dodaš še drevesa in hiše.

AVTO

Avto lahko uvoziš iz mape Prevozi (Transportation) ali pa ga narišeš v *Urejevalniku slik*. Poskusi ga narisati. Klikni na gumb *Nariši novo skupino slik* in v *Urejevalniku slik* nariši majhen avto. Imenuj ga Avto.

Začni z majhnim ovalnim likom. Dodaj svetlejša lika, ki ponazarjata steklo, in nariši še lučki. Nato nariši še kolesa ter izpuh.

Avto vodiš s pomočjo puščic na tipkovnici:

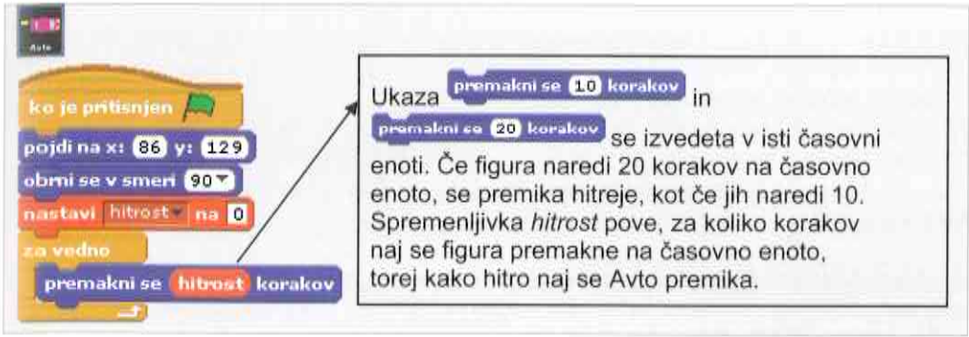
- smer določaš s puščicama levo in desno;
- hitrost povečujeva s puščico gor. Vsakič ko pritisneš na puščico gor, se hitrost Avta poveča.

Hitrost Avta se zmanjšuje, če z dirkalne steze zapelje na travnik.

Gibanje Avta je premikanje naprej za določeno število korakov. Za koliko korakov, bo določila spremenljivka, saj se hitrost Avta spreminja. Vrednost spremenljivke se bo večala, kadar bova pritisnila puščico gor, in manjšala, če bo avto zapeljal na travnik.

Naredi novo spremenljivko, ki jo bodo lahko uporabljale vse figure, in jo imenuj *hitrost*. Napiši ukazni blok za premikanje Avta ter določi njegovo začetno lego in hitrost:

- Avto je na začetku postavljen za startno črto;
- usmerjen je v desno;
- Avto se ves čas premika naprej, število korakov določa spremenljivka hitrost;
- na začetku Avto miruje, zato je vrednost spremenljivke *hitrost* na začetku enaka 0.



Napiši še ukazni blok za pospeševanje. Vsakič ko bo pritisnjena puščica gor, se bo hitrost Avta nekoliko povečala. To pomeni, da se bo povečala vrednost spremenljivke *hitrost*. Za koliko, ugotovi s poskusi.

Avto morava zdaj usmeriti po progi. S pritiskom na puščici levo ali desno se bo obrnil v določeno smer za nekaj stopinj. Obračanje Avta ne sme biti prehitro.

Razmisli, kako naj se hitrost Avta zmanjšuje, ko zapelje na travnik. Najbolj smiselno je, da se njegova hitrost zmanjšuje postopno. Dlje kot je na travniku, počasneje naj se premika. Problema se tu ne da rešiti tako preprosto kot pri večanju hitrosti. Poglejva si to na konkretnem primeru. Reciva, da je vrednost spremenljivke *hitrost* 10 in da se njena vrednost zmanjšuje po 2. Čez nekaj trenutkov bo tako vrednost spremenljivke hitrost 8, nato 6 in hitro bova prišla do 0 in celo do negativnih vrednosti. Tega pa ne želiva. Zato poiščiva boljšo rešitev.

Iz matematike veš, da število lahko pomanjšamo z odštevanjem ali pa tako, da ga pomnožimo s številom, manjšim od 1. Poglejva si primer, kaj se dogaja s številom 10, če ga večkrat zapored pomnoživa s številom 0,8.
$10 \cdot 0{,}8 = 8$; število 10 se zmanjša za 2
$8 \cdot 0{,}8 = 6{,}4$; število 8 se zmanjša za 1,6
$6{,}4 \cdot 0{,}8 = 5{,}12$; število 6,4 se zmanjša samo še za 1,28
Število korakov se tako manjša, vendar vsakič za manj korakov.

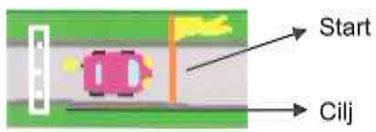
Napiši ukazni blok. Ob dotiku Avta z zeleno barvo se bo njegova hitrost manjšala tako, da bova spremenljivko *hitrost* množila s številom, manjšim od 1. S katerim številom pomnožiti spremenljivko, ugotoviš s poskusi.

CILJ

Ko se Avto dotakne ciljne črte, se na cilju izpiše čas, ki ga je izmerila štoparica.
Cilj ima naslednje naloge:

- zaznati, kdaj se Avto dotakne bele ciljne črte,
- pokazati mora čas vožnje in
- končati igro.

Nariši novo figuro, belo ciljno črto, in jo imenuj Cilj. S kazalcem miške določi Cilju ustrezno izhodišče za startno črto.



Med ukazi *Zaznavanja* je ukaz *ponastavi štoparico*, ki zažene štoparico ter ukaz *stoparica*, ki pove čas. Uporabi ju in napiši ukazni blok, ki bo na cilju prikazal čas vožnje:

- štoparico je na začetku treba nastaviti,
- Cilj čaka, da se ga dotakne Avto,
- takrat pokaže čas vožnje (uporabi ukaza *stoparica* in *reci*) in
- igra se konča.

Če želiš, da se vrednost štoparice prikaže na odru, na ukazni paleti odkljukaj potrditveno polje pred spremenljivko *stoparica* 

.

Podobno dirkalno igro najdeš tudi v mapi Primeri (Examples). Igra ima podoben program kot najina, le da je nekoliko zahtevnejši.

RAČUNALNIŠKI MAČEK

1. **naloga:** Avtu omeji čas vožnje. Če je prepočasen, naj se pojavi napis: »Poskusi znova, klikni na zeleno zastavico.«

2. **naloga:** Naredi več sob. V vsaki sobi je proga bolj zavita in na njej naj se pojavijo različne ovire. Za to, da lahko prideš v naslednjo sobo, moraš biti dovolj hiter.

NAPAD NA LUNO

OPIS IGRE

Luno ogrožajo napadalci iz vesolja. Nadzorovati moraš obrambno raketo, ki se nahaja na Luni. Tvoji sovražniki v raketah se ves čas skušajo prebiti mimo tebe, zato ti dela ne bo zmanjkalo. Igra je končana, ko je obrambna raketa zadeta, do takrat pa se moraš obraniti pred čim več sovražniki.

NAČRT

Elementi igre	Dejanje	Program napišemo elementu
Obramba	Obračaš jo s puščicami na tipkovnici ← in →.	
Izstrelek	Če pritisneš presledek, se sproži in premika naprej.	
Raketa	Leti proti Obrambi.	
in	Obramba določi Izstrelku smer in izhodišče.	
in	Ko se Raketa dotakne Izstrelka: - spremeni obleko , - se pojavi kot nova raketa.	
in	Ko se Raketa dotakne Obrambe: - spremeni obleko , - konča igro.	
Spremenljivka Rezultat	Šteje, koliko raket je bilo zadetih.	
Cilj igre	- Preprečiti, da te zadene raketa. - Zadeti čim več raket.	

Programiranje igre bova razdelila na naslednje korake:

1. narisala bova **ozadje**;
2. naredila bova figuro **Obramba**, ki se bo obračala levo in desno;
3. naredila bova figuro **Izstrelek**, ki jo bo Obramba usmerila proti napadalcem;
4. naredila bova figuro **Raketa**, ki napada Luno.

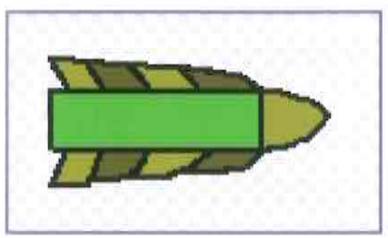
OZADJE

V urejevalniku slik nariši ozadje, Luno in zvezde.

OBRAMBA

Na Luni je postavljena obrambna raketa. Upravljaš jo tako, da jo obračaš s puščicama levo in desno.

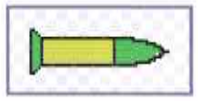
Nariši novo figuro, obrambno raketo, in jo poimenuj Obramba. Smer Obrambe naj bo enaka kot na spodnji sliki.



Obrambi določi začetno lego in smer ter napiši ukazni blok za njeno obračanje levo in desno. Vsakič ko pritisneš puščico levo ali desno, se obramba obrne za majhen kot.

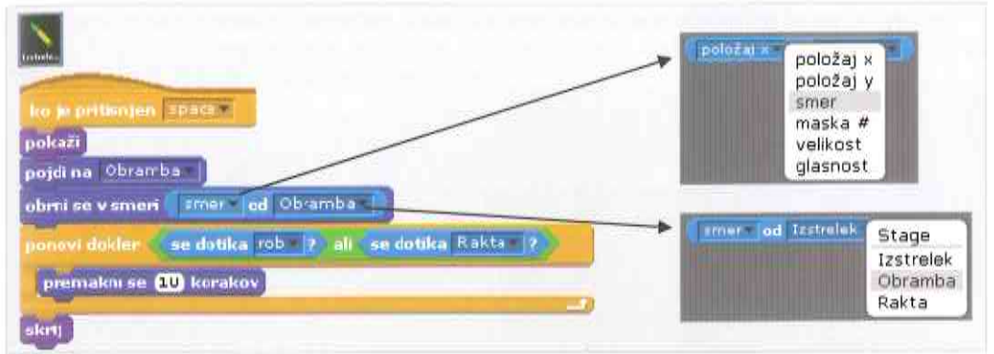
IZSTRELEK

S pritiskom na tipko *presledek* (space), bova v vesolje poslala izstrelek. Nariši novo figuro, podobno obrambni raketi, le nekoliko manjšo, in jo poimenuj Izstrelek. Tudi smer Izstrelka naj bo enaka kot na spodnji sliki.



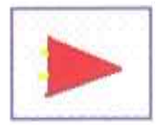
S pritiskom na tipko *presledek* Izstrelek pošlješ v vesolje. Razmisli, kako bova napisala ukazni blok:

- izstrelitev sproži pritisk na tipko presledek;
- začetno lego in smer Izstrelka določa Obramba s pomočjo ukazov pojdi na Obramba in smer od Obramba;
- po izstrelitvi se Izstrelek toliko časa premika naravnost naprej, dokler ne pride do roba odra ali ne zadene rakete (Če želiš uporabiti ukaz *se dotika Raketa*, moraš najprej narediti figuro Raketa.);
- Izstrelek se skrije, dokler ponovno ne pritisneš na tipko presledek.



RAKETA

Raketa poskuša zadeti Obrambo. Če jo zadene, pride do eksplozije in igra se konča. Obramba ji poskuša to preprečiti z izstrelki, ki so usmerjeni vanjo. Če je raketa zadeta, spremeni barvo in rezultat se poveča za 1. V Urejevalniku slik nariši novo figuro, majhno rdečo raketo. Imenuj jo Raketa.

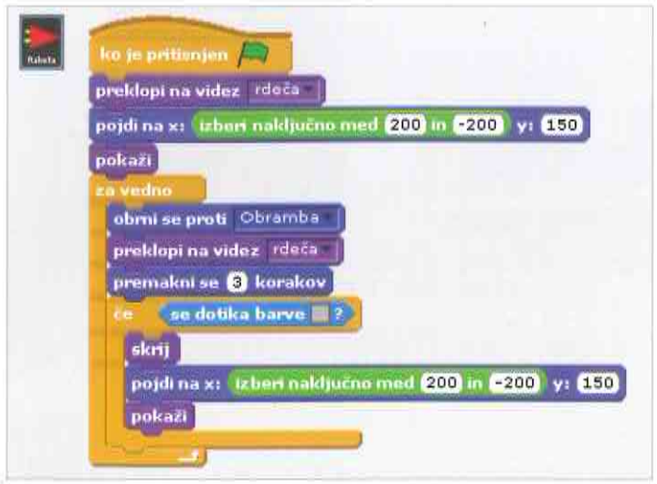


Raketa naj ima tri obleke. Klikni na jeziček Obleke in naredi še dve obleki, belo raketo in eksplozijo. Obleke poimenuj rdeča, bela in eksplozija, kot je razvidno s spodnje slike.

Raketa bo imela precej obsežen ukazni blok. Najprej določiva njeno osnovno gibanje, ki je zelo podobno gibanju Metulja v igri Lovec metuljev:

- Raketa rdeče barve se na začetku igre pojavi na vrhu odra, koordinata x je določena naključno med levim in desnim robom,
- Raketa se z vrha odra premika proti Obrambi,
- gibanje Rakete prekine dotik s sivo barvo Lune,
- Raketa izgine in se spet prikaže na vrhu odra.

Gibanje Rakete se od gibanja Metulja razlikuje le v smeri padanja. Raketa je usmerjena proti obrambi in ne navpično navzdol.



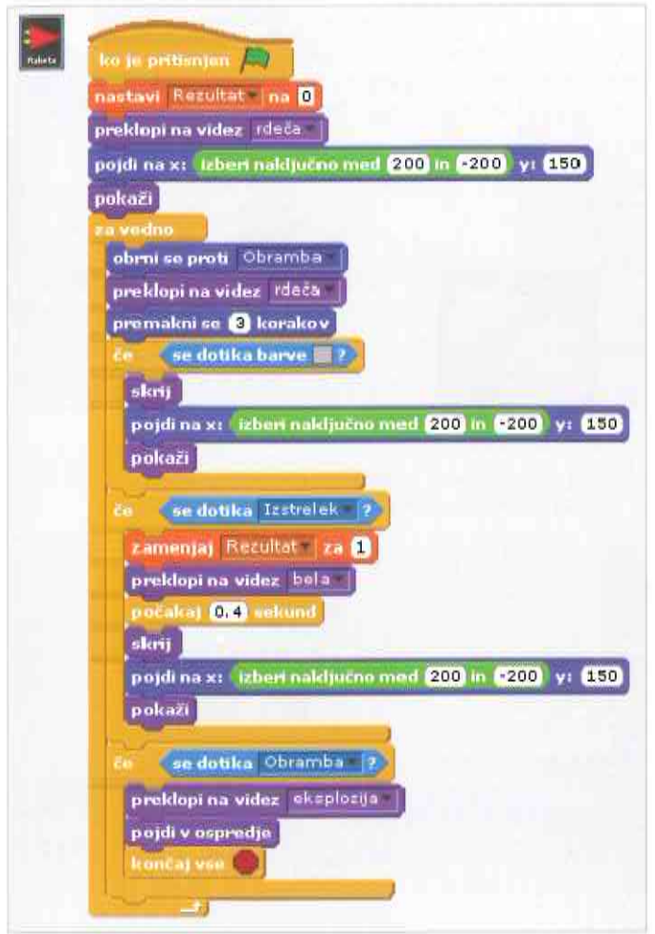
Poglejva, kaj se zgodi, če izstrelek zadene Raketo:

- Raketa za nekaj trenutkov spremeni barvo,
- nato izgine in se spet prikaže na vrhu odra.

Igri dodaj še zaključek, ko Raketa zadene Obrambo:

- Raketa eksplodira, zamenja obleko v eksplozijo,
- igra se konča.

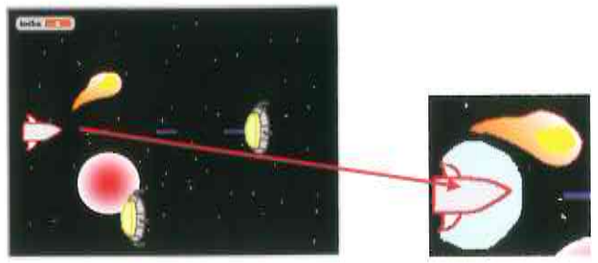
Združi vse ukazne bloke v enega ter dodaj spremenljivko Rezultat, ki bo beležila, kolikokrat je bila Raketa zadeta. Vrednost spremenljivke je na začetku enaka 0. Vsakokrat ko je raketa zadeta, se poveča za 1.



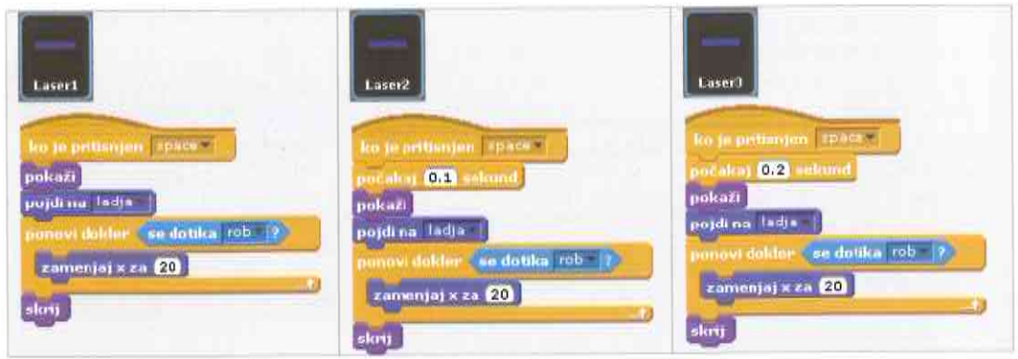
Igra je končana, lahko ji dodaš še primerne zvoke. Na svetovnem spletu lahko najdeš veliko brezplačnih zvočnih učinkov. V iskalnik vtipkaj na primer »free sound effects« in izberi med bogato ponudbo.

RAČUNALNIŠKI MAČEK

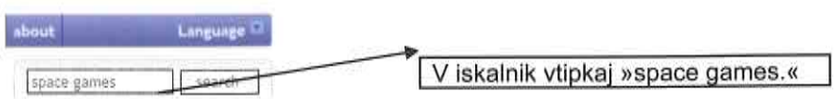
1. naloga: Naredi novo vesoljsko igro. V tej igri upravljaš vesoljsko ladjo, ki mora sestreliti čim več letečih krožnikov. Tako si tako nabiraš točke. Vesoljska ladja se ne obrača, ampak se premika gor in dol. Za vesoljsko ladjo je nevaren le komet. Pred njim se lahko ubrani, če se pokrije z neprebojnim ščitom.



2. naloga: Vesoljska ladja strelja tako, da zaporedoma izstreli tri laserske izstrelke.
Namig: Vsak izstrelek je ena figura. Tokrat si lahko pomagaš kar z ukaznimi bloki.

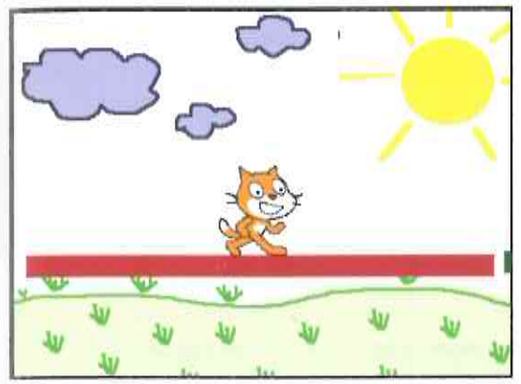


3. naloga: Ta naloga bo povsem drugačna. Na domači strani programa Scratch poišči vesoljsko igro, v kateri premikanje zvezd daje vtis letenja skozi vesolje.



Izberi tisto igro, ki ti je všeč, prenesi jo na svoj računalnik in si oglej, kako je narejeno premikanje zvezd. Poskusi to uporabiti.

SKRITI ZAKLAD



OPIS IGRE

Glavni junak te igre je maček, ki mi je zelo podoben. Kako tudi ne, saj je moj bratranec. Veseli me, da ga lahko spoznaš in ugotoviš, kako spreten je. Hodi po deskah, vtis gibanja pa daje premikajoče se ozadje. Maček mora na svoji poti proti cilju pobrati ključ, s katerim bo prišel do zaklada. Na poti ga čakajo tudi sovražniki, ki se jih mora rešiti.

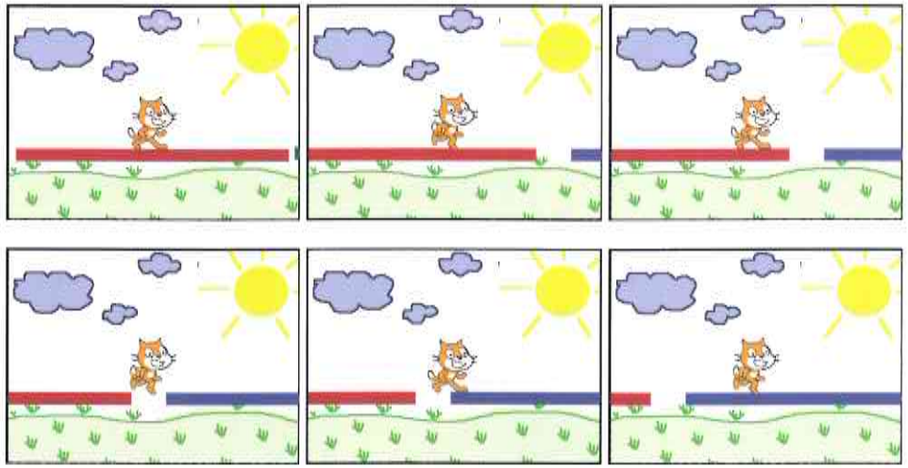
NAČRT IGRE

Namesto preglednice si bova v tej igri raje pogledala razlago, kako se naredi premikajoče se ozadje, saj je to bistvena posebnost te igre.

Programiranje igre bova razdelila na 7 korakov:

1. pogledala bova razlago, kako se naredi **premikajoče se ozadje**;
2. narisala bova **ozadje**;
3. figuri **Maček** bova določila njegovo upravljanje;
4. narisala bova premikajoče se ozadje, **deske**;
5. dodala bova figuro **Ključ**, ki jo mora maček pobrati na poti do cilja;
6. naredila bova figuro **Hudobca**;
7. igri bova naredila **zaključek**.

PREMIKAJOČE SE OZADJE



Igre s premikajočim se ozadjem ustvarjajo vtis, da se glavni junak premika, čeprav se v resnici premika le del ozadja. Oglej si vseh šest sličic in opazuj lego ozadja, mačka in barvnih desk, po katerih hodi. Če pozorno gledaš, opaziš:

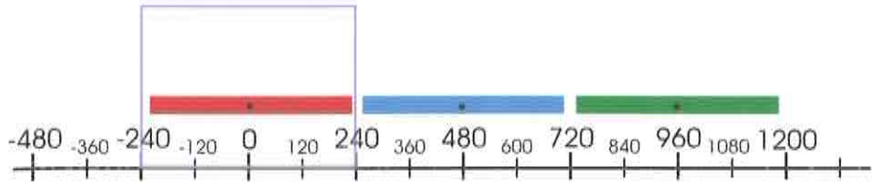
- da je ozadje - sonce, travnik in oblaki, ves čas nespremenjeno,
- da maček sicer dela korake, vendar se na vseh slikah nahaja točno na sredini odra,
- da se lega desk v vodoravni smeri spreminja in tako ustvari videz premikajočega se ozadja.

Lega desk med igro se spreminja, zato jih morava narediti kot figure. Ker se bodo deske premikale le levo in desno, si bova ogledala, kako se jim spreminja lega na osi x.

Položaj figure na odru je določen z dvema spremenljivkama: *položaj* x in *položaj y*. Spremenljivki povesta, kje na odru se nahaja središče figure. Če želiš, da je figura vidna, mora biti njen *položaj x* med -240 in 240. Figura ima lahko tudi drugačen *položaj x*, npr. 500 ali -400, vendar v tem primeru ne bo vidna.

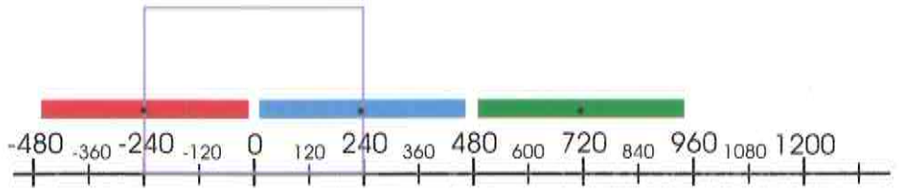
Vidni del osi x se nahaja med – 240 in 240. Na naslednjih slikah je os x podaljšana zunaj vidnega polja na levo in na desno. Nad osjo x so prikazane deske. Poglejva, kako bodo deske med igro spreminjale *položaj x*. Bodi pozoren na središče deske in pomisli, kako bi napisal splošno formulo za lego desk na osi x.

Na začetku igre je rdeča deska vidna v celoti.



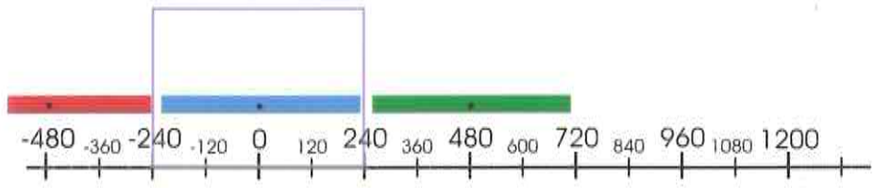
	Položaj x	Splošna formula
Rdeča deska	0	0 + 0*480
Modra deska	480	0 + 1*480
Zelena deska	960	0 + 2*480

Ko maček naredi nekaj korakov, se položaj desk spremeni. Vidi se del rdeče in del modre deske.



	Položaj x	Splošna formula
Rdeča deska	-240	-240 + 0*480
Modra deska	240	-240 + 1*480
Zelena deska	720	-240 + 2*480

Rdeča deska čez nekaj časa izgine, vidi se le še modra deska.



	Položaj x	Splošna formula
Rdeča deska	-480	-480 + 0*480
Modra deska	0	-480 + 1*480
Zelena deska	480	-480 + 2*480

Poišči povezavo med navidezno hojo mačka in premikanjem desk. Ko bo maček hodil na mestu, obrnjen v desno, se bodo deske premikale v levo. Za vsak njegov navidezni korak v desno bodo šle deske za nekaj, reciva za 10 enot, v levo.

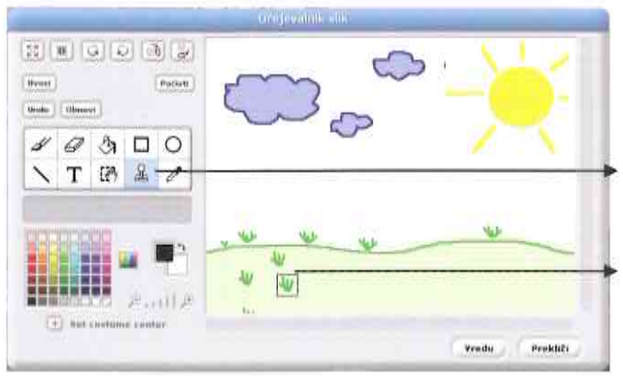
Vez med koraki in premikanjem desk bo spremenljivka, ki bo beležila, koliko korakov bo naredil maček. Poimenujva jo *os_x*. Z vsakim korakom, ki ga bo naredil maček, se bo vrednost spremenljivke pomanjšala za 10, če bo korakal v desno, in povečala za 10, če bo korakal v levo.

Na začetku, ko maček miruje, je vrednost spremenljivke *os_x* enaka 0. Po 24 korakih je vrednost spremenljivke -240. Po 480 korakih pa -480. Če pogledava zgornje tabele, najdeva povezavo med *položajem x* desk na sliki in spremenljivko *os_x*:

položaj rdeče deske je: *os_x*+ 0*480;
položaj modre deske je: *os_x*+ 1*480;
položaj zelene deske je: *os_x*+ 2*480.

OZADJE

Nariši preprosto ozadje, na katerem naj bodo travnik, sonce in oblaki.



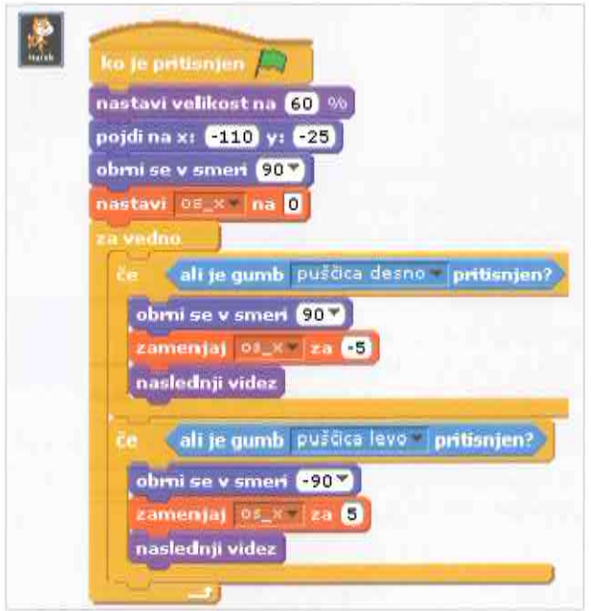
MAČEK

Maček se bo navidez premikal levo in desno. Vodila ga bova s puščicami na tipkovnici. Program za premikanje mačka levo in desno že poznaš. V tem primeru pa bo maček, namesto premika levo ali desno spremenil vrednost spremenljivki *os_x* za na primer 5 oziroma za – 5 enot. Točno za koliko naj se spremeni vrednost spremenljivke, bo najbolje preizkusiti kasneje, ko bodo narejene tudi deske.

Figuro že imaš, moraš jo le poimenovati. Imenuj jo kar Maček. Naredi še spremenljivko *os_x*, ki jo bodo lahko uporabljale vse figure.

Napiši ukazni blok za vodenje Mačka levo in desno s puščicami na tipkovnici. Maček hodi le na videz, vtis premikanja vzbuja spreminjanje njegove obleke. Z vsakim korakom pa se spremeni vrednost spremenljivke *os_x*. Določi tudi nekatere začetne vrednosti:

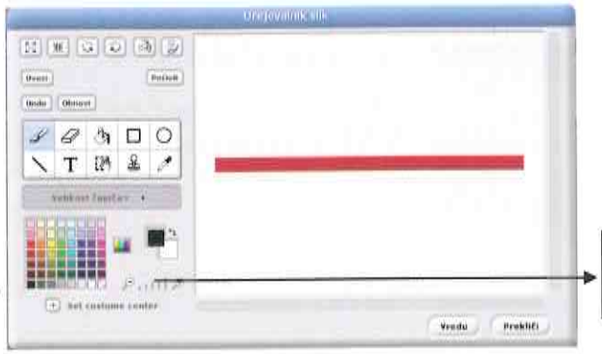
- Maček je na začetku igre postavljen v levi kot tik nad travnikom,
- obrnjen je v desno in
- je nekoliko pomanjšan,
- začetna vrednost spremenljivke *os_x* je enaka 0.



RDEČA DESKA

Maček zdaj že hodi na mestu, za vtis premikanja pa bodo poskrbele deske. Vsaka od desk bo narejena kot ena figura. Naredila bova tri deske: rdečo, zeleno in modro.

Začni z rdečo. Nariši novo figuro, tanek rdeč pravokotnik, in ga imenujva Scena1.



Izberi najmanjšo povečavo, saj mora deska segati skoraj do konca zaslona.

Desko s pomočjo kazalca miške premakni gor ali dol, da bo nameščena pod Mačkom. Rdeča deska bo na začetku igre vidna v celoti, zato bo njeno središče v sredini odra. Lega deske je določena s pomočjo spremenljivke *os_x*, ki se spreminja, ko Maček hodi. Če še enkrat pogledaš razlago o premikajočem se ozadju, ti ne bo težko napisati ukaznega bloka za premikanje rdeče deske:

- vedno, kadar Maček hodi, se deski spreminja koordinata x,
- koordinato x bova določila s pomočjo formule: lega rdeče deske je enaka: *os_x*+ 0*480.

Program preizkusi. Če se ti zdi premikanje deske prehitro ali prepočasno, temu primerno določi, za koliko enot naj se z vsakim korakom spremeni vrednost spremenljivke *os_x*.

MODRA DESKA

Naredi modro desko. Podvoji rdečo desko in v Urejevalniku slik desko prebarvaj v modro. Desko preimenuj v Scena2.

V ukaznem bloku popravi začetno vrednost deske. Zopet pokukaj v razlago.

ZELENA DESKA

Še enkrat podvoji figuro Scena1 in v urejevalniku slik desko prebarvaj v zeleno. Na desni strani nad desko nariši še majhno ključavnico.

V ukaznem bloku popravi le začetno vrednost koordinate x.

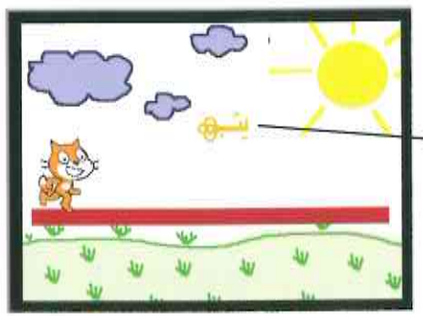
Igro še enkrat preizkusi. Kot vidiš, se delček desk vedno nahaja na robu zaslona. To lahko rešiš tako, da narišeš novo figuro v obliki okvirja. Imenuj jo Okvir.

Če ti ta rešitev ni všeč, si lahko pomagaš z ukazoma *skrij* in *pokaži*. Razmisli, kako se to naredi.

KLJUČ

Na svoji poti mora Maček pobrati ključ, saj lahko le z njim na cilju odklene ključavnico. Iz mape Stvari (Things) uvozi novo figuro, ključ (key), in jo primerno pomanjšaj. Imenuj jo Ključ.

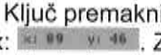
Podobno kot deske bo tudi Ključ del premikajočega se ozadja. Zato bo njegov program podoben kot pri deskah. Odločiti se moraš, kam ga postaviti. Najprej ga postavi točno na sredino odra, nad rdečo desko.



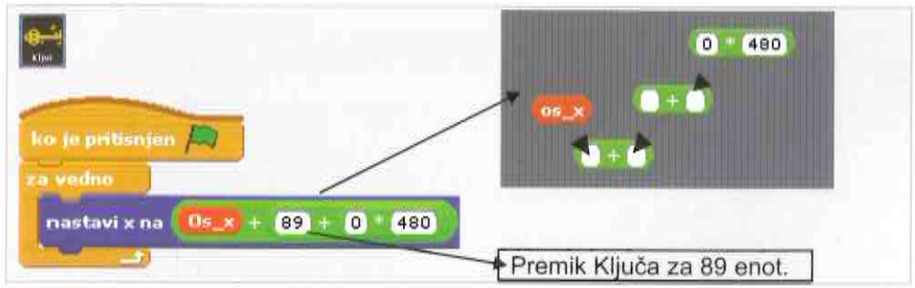
Odčitaj začetno lego na osi x:

Ključ se bo na začetku pojavil na sredini odra in se nato premikal skupaj z rdečo desko. Ker je lega Ključa na osi x enaka kot pri rdeči deski, lahko prepišeš kar njen ukazni blok.

Kaj pa, če želiš, da je Ključ na začetku igre pomaknjen bolj v desno?

Najprej Ključ premakni v desno in poglej, kako premik vpliva na njegov začetni položaj na osi x: 

. Zaradi premika je sedaj koordinata x enaka 89. Ta premik je treba upoštevati pri pisanju ukaznega bloka. Razmisli, kako.



Maček mora za to, da pride do ključa, narediti manjši skok. Skočil bo s pritiskom na *puščico gor*. Izberi figuro Mačka in napiši ukazni blok. To že znaš, oceniti pa moraš, kako visok naj bo skok, da bo Maček dosegel Ključ.

Vrniva se h Ključu. Ko se Maček dotakne Ključa, se ta skrije. Podatek o tem, da ima Maček Ključ, bo imela spremenljivka. Ta bo imela vrednost 0, dokler se Maček ne dotakne Ključa in vrednost 1, ko se Maček Ključa dotakne. Ključavnica se odklene le v primeru, če bo vrednost spremenljivke enaka 1.

Naredi novo spremenljivko, ki jo bodo lahko uporabljale vse figure. Imenuj jo *ima_ključ*.
Napiši ukazni blok, ki bo določil, kako se bo spreminjala vrednost spremenljivke *ima_ključ*:

- na začetku bo vrednost spremenljivke *ima_ključ* enaka 0,
- spremenljivka nato čaka, dokler se Maček ne dotakne ključa, nato se njena vrednost nastavi na 1.

Ključ je na začetku igre viden, ko se ga Maček dotakne, se skrije.

HUDOBEC

Maček mora na svoji poti preskočiti hudobno pošast, sicer igro izgubi. Nariši novo figuro, majhno pošast, in jo imenuj Hudobec. Natančno velikost Hudobca lahko določiš kasneje. Podobno kot Ključ bo tudi Hudobec del premikajočega se ozadja.

Ukazni blok za premikanje Hudobca bo podoben tistemu za premikanje Ključa. Hudobec se bo nahajal na desnem robu modre deske. Razmisli, kako določiš koordinate za njegov začetni položaj.

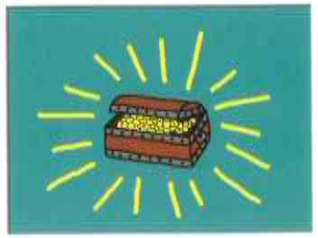
Če se Maček dotakne Hudobca, je igra izgubljena. Preden se konča, se Hudobec:

- večkrat poveča in pomanjša in s tem opozori, da se ga je Maček dotaknil,
- pošlje naprej sporočilo *žalosten konec*.

Ker se velikost Hudobca med igro spreminja, ne pozabi določiti njegove začetne velikosti.

ZAKLJUČEK

Igra ima dva zaključka. Če se Maček dotakne Hudobca, ta pošlje sporočilo *žalosten konec* in igra je izgubljena. Če pa Maček premaga ovire na poti, pride do zaklada. Poglejva si zaključek s srečnim koncem, ko Maček dobi zaklad. Nariši novo figuro, skrinjo zlata, in jo imenuj Zaklad.



Igra se srečno zaključi, če sta izpolnjena dva pogoja:

- Maček je na poti pobral Ključ,
- Maček se je dotaknil ključavnice (rumene barve).

Če sta izpolnjena oba pogoja, Maček pošlje sporočilo *konec*.

In še ukazni blok za Zaklad:

- na začetku je skrit,
- ko prejme sporočilo *konec*, se pokaže.

Verjetno ne bo težko narediti še zaključka, kadar je igra izgubljena.

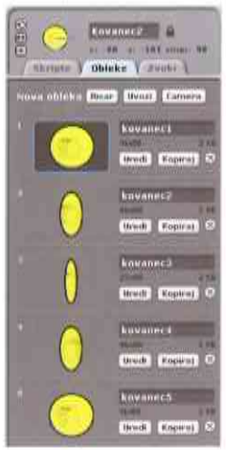
RAČUNALNIŠKI MAČEK

1. naloga: Igri Skriti zaklad naredi daljšo pot, torej več desk.

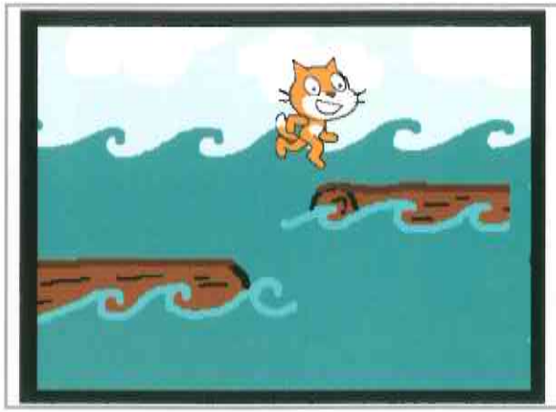
2. naloga: Maček lahko na svoji poti pridobiva točke z nabiranjem vrtečih se kovancev, ki jih sreča na poti.

Namig: Če želiš, da se kovanec vrti, mu nariši več oblek. Ukaz *naslednji videz* bo poskrbel za vrtenje.

3. naloga: Igri dodaj čarovnika, ki strelja. Maček mora izstrelek preskočiti.



DIVJA REKA



OPIS IGRE

Igra Divja reka je nekoliko preoblikovana igra Skriti zaklad. Igra se razlikuje po tem, da Maček namesto po deskah, skače s hloda na hlod. Če skoči v prazno, se potopi.

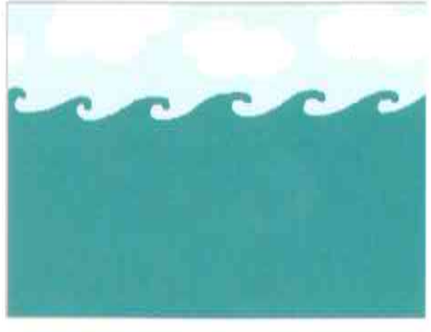
NAČRT IGRE

Ker je igra podobna Skritemu zakladu, odpri kar to igro in jo shrani z novim imenom Divja reka. Igro bova le nekoliko preoblikovala v naslednjih korakih:

1. narisala bova novo **ozadje**;
2. deske bova preoblikovala v plavajoče **hlode**;
3. napisala bova nove ukazne bloke za Mačka, saj se bo, če bo stopil v prazno, potopil.

OZADJE

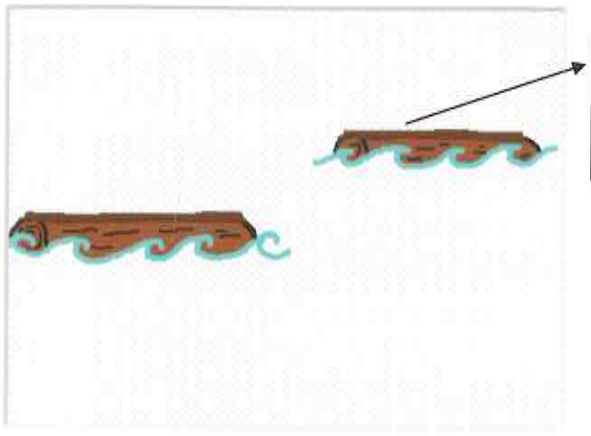
V Urejevalniku slik izbriši staro ozadje in nariši nekoliko drugačnega.



HLODI

Rdečo, modro in zeleno desko bova preoblikovala v plavajoče hlode. Maček bo skakal s hloda na hlod, če mu to ne bo uspelo, bo padel v vodo. Preden bova deske preoblikovala, je smiselno narediti načrt, kako bodo med potjo postavljeni hlodi, da skoki ne bodo prelahki, hkrati pa morajo biti izvedljivi. Postavi jih na primer takole:

Izberi rdečo desko in namesto nje nariši dva plavajoča hloda. Razmak med hlodoma naj bo dovolj velik. Čeprav sta v Urejevalniku slik narisana dva ločena hloda, je to še vedno ena figura.

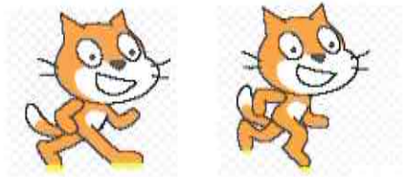


Ukazni blok lahko ostane enak. Podobno naredi še z modro in zeleno desko.

MAČEK

Maček v igri Skriti zaklad hodi levo in desno ter skače. V Divji reki pa bo tudi padal, če pod tačkami ne bo imel trdne podlage.

Izberi figuro Mačka, klikni na jeziček Obleke in pri obeh oblekah Mačku pod tačkami nariši rumene črte, ki mu bodo pomagale pri zaznavanju podlage.



Klikni na jeziček *Skripte*. Program za Mačka bova nekoliko spremenila. Dodala bova ukazni blok, ki poskrbi, da Maček vedno pada (spreminja koordinato y), kadar se ne dotika trdne podlage hloda (se z rumeno barvo tačk ne dotika rjave barve hloda).

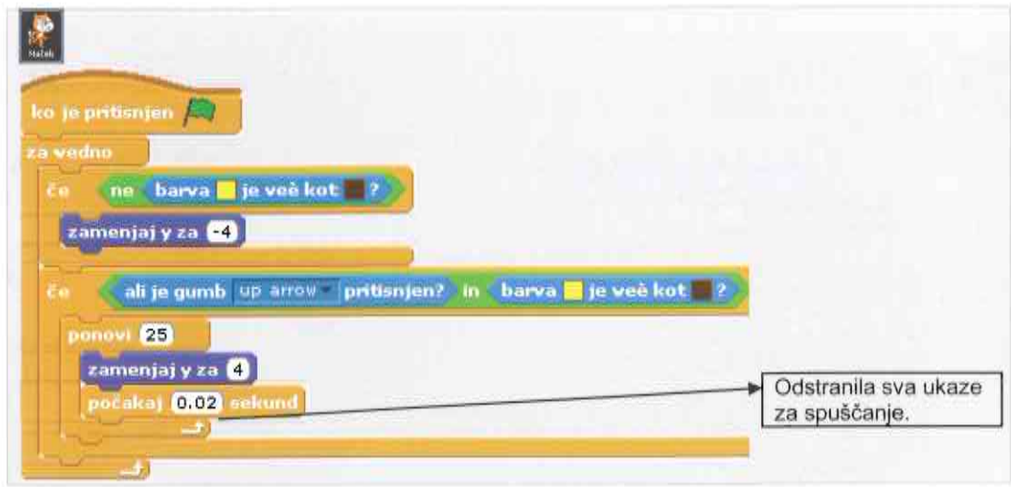
Preizkusi program. Opazuj, kaj se dogaja, ko Maček skače. Sedaj preoblikuj ukazni blok za skakanje. Dodaj mu pogoj, da se Maček lahko odrine le, če se dotika trdne podlage.

Zopet preizkusi program. Kot vidiš, Maček ne skače tako, kot sva si zamislila. Poskusi ugotoviti zakaj. Primerjaj ukazni blok za skakanje in za padanje.

Kadar je pritisnjena *puščica gor,* začneta na Mačka hkrati delovati dva ukazna bloka. Eden ga premika navzgor, drugi pa ga hkrati premika navzdol. Maček zato med premikanjem navzgor ne sme hkrati tudi padati. V uvodu knjige sva se naučila, kako lahko s pravilnim vrstnim redom sestavljanja ukazov vplivaš na potek dogodkov. Še enkrat si v Uvodu poglej poglavje Upravljanje in popravi program.

Da bo imel Maček več časa za skok na drug hlod, lahko skoči nekoliko višje in počasneje.

Preizkusiva program. Kot vidiš, se pojavlja nova težava. Ko Maček skače na drug, višje postavljen hlod, se pri spuščanju na njem ne ustavi, ampak pada dalje. Razmisli, zakaj. Ker je pri skoku točno določeno, koliko časa se Maček spušča, se lahko spusti pod rjavo črto hloda, ki prepreči njegovo padanje. Rešitev je preprosta. Odstraniti moraš le nekaj ukazov. Razmisli, katere, ali pa si pomagaj s spodnjo rešitvijo.



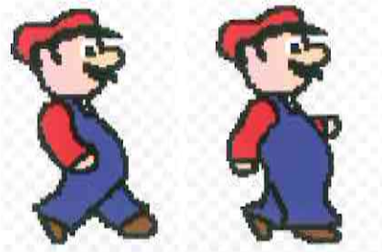
Narediva še en lepotni popravek. Kadar pritisneš puščico gor in držiš še puščico levo ali desno, maček v zraku premika noge, kar ni lepo. Razmisliva, kako bi to popravila. V spodnjem ukaznem bloku je napisana rešitev za premikanje v desno.

Igro lahko sedaj izboljšaš na različne načine. Verjetno bo treba popraviti tudi začetno lego Ključa in Hudobca.

RAČUNALNIŠKI MAČEK

Naloga: Naredi igro Super Mario. Verjetno jo poznaš, če ne, pa poglej na svetovni splet, kjer je veliko različnih in brezplačnih variacij te igre. Znanja imaš že več kot dovolj. Tudi domišljije ti gotovo ne bo zmanjkalo.

Pomoč: Figuro Mario lahko narišeš podobno, kot je prikazano na slikah spodaj.



Narediš mu lahko tudi več oblek: ko skače, se obrača in podobno. Če ne maraš kaj prida risati, lahko figure poiščeš tudi na svetovnem spletu. V iskalnik vpiši besedi: sprite in Mario. Presenetilo te bo, koliko figur je na voljo.

Tako, prišla sva do konca knjige. Gotovo si zdaj že pravi računalniški maček in lahko programiraš samostojno. Zato moja pomoč ne bo več potrebna. Prebudi svojo domišljijo in uživaj v ustvarjanju novih iger in drugih *projektov*.

RECENZIJE

Z velikim veseljem sem prijel v roke knjigo, ki bo nedvomno končala na polici marsikaterega učitelja računalništva v osnovni šoli, pa tudi tistih staršev, ki bi radi popeljali svoje otroke v čudoviti svet računalništva. Knjiga je posvečena dokaj novemu programskemu jeziku Scratch, ki je namenjen otrokom od 10. do 16. leta (po lastnih izkušnjah pa lahko tudi mlajšim). Knjiga najprej predstavi sam program. V drugem delu bralca sistematično uči osnov programiranja. Tretji del pa dopolnjujejo računalniške igre, ki dajejo mladini ustrezno motivacijo, hkrati pa kažejo domet tega simpatičnega orodja, ki je lahko ob primerni mentorski podpori učitelja ali starša precej več kot igrača.

Knjiga je pisana slikovito in v zelo všečnem slogu. Navdušuje me njena sistematičnost, postopnost in hkrati celovitost. Odlikuje jo tudi dobro in strokovno korektno izrazoslovje, ki je hkrati prilagojeno otroškemu razmišljanju. Predstavlja izjemno pomemben prispevek k popularizaciji računalništva, ki se s tem res lahko začenja v zgodnjih otroških letih. Pa tudi starejši - učitelji in starši, ki so kakorkoli povezani z računalništvom in z otroško zvedavostjo, bodo radi posegli po njej.

Prof. dr. Saša Divjak

Poznavanje osnov določenega programskega jezika je danes del informacijske pismenosti prav vsakega posameznika. Žal pa prvi koraki v svet programiranja običajno niso najlažji.
Že narobe postavljena vejica lahko povzroči, da program ne deluje. Prav tako je težko napisati programe, ki bi bili za začetnike zanimivi: z veliko grafike, akcije, uporabo animacij, zvoka ...

Avtorji jezika Scratch so želeli (in uspeli) sestaviti programski jezik in okolje, kjer začetniki zlahka napišejo privlačne in zanimive programe in ob tem spoznajo osnovne koncepte programskih jezikov.

Čeprav se je jezika zelo enostavno naučiti, pa je kljub temu zelo dobro, če si lahko pri tem pomagamo z dobrimi navodili, privlačnimi zgledi in kopico idej. In vse to nam prinaša knjiga Sonje Lajovic, ki pomeni dobrodošlo novost med računalniško literaturo. Upam, da bo ta knjiga omogočila številnim, da se bodo pogumno spustili v lep in zanimiv svet programiranja.

Mag. Matija Lokar

Zelo rada igram igre na računalniku, ko pa sem spoznala Scratch, sem ugotovila, da jih je še bolje delati. Nikoli se ne naveličam. Všeč mi je tudi, ker moraš veliko razmišljati, da narediš kaj dobrega. In potem zelo rada poskušam to, kar sem naredila.

Lara, 7. Razred