Priručnik za nastavnike za izvannastavnu aktivnost
iz informatike od 5. do 8. razreda

OD BITA DO ROBOTA

skupina autora

Robotika

Inkscape

PHP

Logo

Stop-motion animacija

Wordpress

Ulaganje u budućnost
Projekt je sufinancirala Europska unija
iz Europskog socijalnog fonda

OD BITA DO ROBOTA

skupina autora

priručnik za nastavnike za
izvannastavnu aktivnost iz
informatike od 5. do 8. razreda

AUTORI

Željka Orčić (1. poglavlje - Logo)
Darko Vencl i Vladimir Bićanić (2. poglavlje - Inkscape)
Goran Mučnjak (3. poglavlje - Robotika)
Željko Lozušić (4. poglavlje - Wordpress)
Nikolina Jurić (dodatni sadržaj - PHP i Stop-motion animacija)

GRAFIČKO OBLIKOVANJE

Grafička škola u Zagrebu

TISAK

Grafička škola u Zagrebu

U izradi priručnika sudjelovali su i Marijana Brdar (uređivanje), Danijela Nežak i učenici Grafičke škole u Zagrebu (prijelom), Gordana Fileš (lektura).

Priručnik je izdan u sklopu projekta ICT4SCF (Informatička znanja za uspješnu i kreativnu budućnost) sufinanciranog sredstvima Europske unije iz Europskog socijalnog fonda pri Daljnjem razvoji i provedbi hrvatskoga kvalifikacijskog okvira.

skupina autora
Od bita do robota
Izvannastavna aktivnost iz informatike
za osnovne škole

Zagreb, Slatina, 2014.

Ova publikacija je ostvarena uz financijsku potporu Europske komisije.
Ova publikacija odražava isključivo stajalište autora publikacije i Komisija se ne može smatrati odgovornom prilikom uporabe informacija koje se u njoj nalaze.

Ulaganje u budućnost
Projekt je sufinancirala Europska unija
iz Europskog socijalnog fonda

Priručnik za nastavnike za izvannastavnu aktivnost
iz informatike od 5. do 8. razreda

OD BITA DO ROBOTA

skupina autora

Robotika
Inkscape
PHP
Logo
Stop-motion animacija
Wordpress

Sadržaj

1.0 Logo - programski jezik 12
1.1 Što je Logo? 12
1.1.1 Uvod u programiranje 12
1.1.2 Osnovne naredbe za crtanje 13
1.1.3 Pisanje programa u editoru 18
1.1.4 Crtanje jednakostraničnog trokuta 22
1.1.5 Bez traga 25
1.1.6 Potprogrami u programskom jeziku Logo 26
1.1.7 Ah, ta ponavljanja! Upotreba petlji 29
1.1.8 Crtanje mnogokuta 33
1.1.9 Crtam crtu duljine koje želim 36

2.0 Inkscape 44
2.1 Uvod 44
2.2 Vektorska grafika 44
2.3 Inkscape – programsko sučelje 45
2.4 Geometrijski oblici 46
2.4.1 Pravokutnici, kvadrati i 3d objekti 46
2.4.2 Elipse, kružnice i isječci kružnice 47
2.4.3 Mnogokuti, zvijezde i spirale 49
2.5 Krivulje 51
2.5.1 Bezierova krivulja 51
2.5.2 Izrada i uređivanje krivulje 51
2.5.3 Operacije s krivuljama (crop, weld, intersect ...) 53
2.5.4 Efekti na krivuljama (lpes) - primjeri 55
2.5.5 Poveznice (konektori) 57
2.6 Tekst 58
2.6.1 Unos, promjena, uređivanje i oblikovanje teksta 58
2.6.2 Tekst na krivulji i u obliku 58
2.7 Atributi i alati 59
2.7.1 Ispune objekata 59
2.7.2 Konture objekata 60
2.7.3 Alat tweak, spray, eraset i paint bucket 60
2.8 Efekti i filteri 61
2.8.1 Standardni filteri 61
2.8.2 Korisnički (napredni filteri) filteri 61
2.9 Skiciranje (trace) slike 62
2.9.1 Osnovno skiciranje slike 62
2.9.2 Napredno skiciranje slike 63
2.10 Dodatne mogućnosti programa 63
2.10.1 Raspoređivanje objekata 64
2.10.2 Rad s bojama (rgb, grayscale...) 64
2.10.3 Interpolacija i ekstrudiranje objekata, dodavanje efekta na objekte 65
2.10.4 Uvoz i spremanje slika u različitim formatima 67
2.10.5 Rasterizacija vektorskih objekata 67
2.10.6 Renderiranje 68
2.10.7 Vizualizacija krivulje 71
2.11 Svg na internetu 72
2.11.1 Jednostavan svg prikaz 72
2.11.2 Položaj svg-a 73
2.11.3 Dodavanje veze 73
2.11.4 Dodavanje javascript-a 74
2.11.5 Jednostavne animacije 75

3.0 Robotika 78
3.1 Uvod 78
3.2 Strujni krug 79
3.3.1 Strujni krug s izmjeničnim tipkalom 80
3.3 Izmjenično tipkalo 80
3.3.2 Paralelni spoj tipkala (logički sklop ili – „or“) 81
3.3.3 Serijski spoj tipkala (logički sklop i – „and“) 82
3.3.4 Upravljanje elektromotorom pomoću dva tipkala 83
3.4 Postupak izrade vodiča 84
3.4.1 Provjera ispravnosti vodiča univerzalnim instrumentom 85

3.5 Sastavljanje trošila (žaruljica) 85
3.5.1 Strujni krug s izvorom napajanja (baterija), tipkalom i trošilom (žaruljica) 86
3.5.2 Spajanje više žaruljica s tipkalima 87
3.6 Semafor 89
3.7 Strujni krug s izvorom napajanja (baterija), tipkalom i motorom (pokretanje motora) 91
3.8 Shema spajanja tipkala i motora robotskih kolica (upravljanje vozilom pomoću tipkala) 92
3.8.1 Primjer spajanja motora, mjenjačke kutije i kotača na osovini 93
3.9 Elementi za slaganje vozila - robotskih kolica 94
3.9.1 Primjeri slaganja robotskih kolica (korak po korak) 95
3.10 Sučelje za spajanje robotičkih sustava 98
3.10.1 Primjeri robota i robotskih kolica sa sučeljem 99
3.11 Sučelje programa RoboPro 100
3.11.1 RoboPro - Program elements – basic elements 102
3.12 Primjeri programa koji se mogu raditi s učenicima 104

4.0 Wordpress
4.1 Uvod 110
4.1.1 Dinamička mrežna stranica - uvod 110
4.1.2 *WordPress* - predstavljanje platforme 110
4.2. Instalacija i konfiguriranje Apache mrežnog poslužitelja i MySQL-a 112
4.2.1. Postavljanje Apache mrežnog poslužitelja, priprema MySQL-a i PHP-a 112
4.2.2. Kreiranje MySQL *Database WordPressa* 113
4.2.3. Instalacija i konfiguriranje postavki *WordPressa* 115
4.3. Izrada teme WordPressa i rad na localhostu 117
4.3.1. Kreiranje datoteka 117
4.3.2. Izrada teme Wordpressa 118
4.3.3. Uređivanje teme na lokalnom hostu 126
4.4. WordPress online 127
4.4.1. Kreiranje baze podataka WordPress MySQL 127
4.4.2. Instalacija WordPressa 128
4.4.2.1. Kopiranje datoteka na mrežni poslužitelj 128
4.4.2.2. Konfiguriranje postavki 128
4.4.2.3. Pokretanje instalacijske skripte 129
4.4.3. Podešavanje osnovnih postavki WordPressa 130
4.4.4. Sigurnosne postavke WordPressa 132
4.4.5. Arhiviranje podataka 135

5.0 PHP 140
5.1 Uvod 140
5.2 Povijest PHP-a 141
5.3 Instalacija PHP-a 141
5.4 Sintaksa PHP-a 142
5.4.1 Komentari u PHP-u 143
5.5 PHP varijable 144
5.5.1 Deklariranje varijable 144
5.5.2 Vrste varijabli 144
5.5.2.1 *String* varijable u PHP-u 145
5.5.2.2 *Integer* varijable u PHP-u 145
5.5.2.3 Logičke varijable 145
5.6 Konstante 147
5.7 PHP operatori 147
5.7.1 Aritmetički operatori 147
5.7.2 Operatori dodjeljivanja 148
5.7.3 Operatori usporedbe 149
5.7.4 Logički operatori 149
5.7.5 Operator spajanja – točka (.) 150
5.8 Uvjetne naredbe u PHP-u 151
5.8.1 If naredba 151
5.8.2 If…else naredba 152
5.8.3 If…elseif…else naredba 152
5.8.4 Switch nareba 154
5.9 Nizovi u PHP-u 155

5.9.1 Jednodimenzionalni cjelobrojni nizovi 156
5.9.1.1 Pristup jednodimenzionalnom cjelobrojnom nizu 156
5.9.2 Jednodimenzionalni asocijativni niz 157
5.9.3 Višedimenzionalni nizovi 157
5.10 Petlje u PHP-u 158
5.10.1 For petlja 158
5.10.2 While petlja 159
5.10.3 Do ... while 160
5.10.4 For...each petlja 160
5.10.4.1 Pristup jednodimenzionalnim asocijativnim nizovima pomoću for...each petlje 161
5.10.4.2 Pristup višedimenzionalnim asocijativnim nizovima pomoću for...each petlje 162
5.11 Funkcije 162
5.12 PHP i HTML forme - POST i GET metode slanja i prihvata podataka 163

6.0 Stop motion animacija

6.1 Uvod 168
6.1.1 Osnovni pojmovi stop motion animacije 168
6.2 Povijest 170
6.3 Vrste stop motion animacije 170
6.4 Izrada stop motion animacije 171
6.4.1 Sinopsis i storyboard 172
6.4.2 Pozornica 174
6.4.3 Snimanje 176
6.4.4 Montaža 177
6.4.5 Dodavanje zvuka 186

1. poglavlje

LOGO - programski jezik

5. razred

1.0 Logo - programski jezik

1.1 Što je Logo?

Dobro došli u svijet programiranja! Dobro došli u Logo-svijet! Na prvi pogled programski jezik Logo je drugačiji od ostalih „klasičnih" programskih jezika i mnogi će reći i kompliciraniji. No, u radu s djecom shvatila sam da taj programski jezik djeca usvajaju brzo i s veseljem.

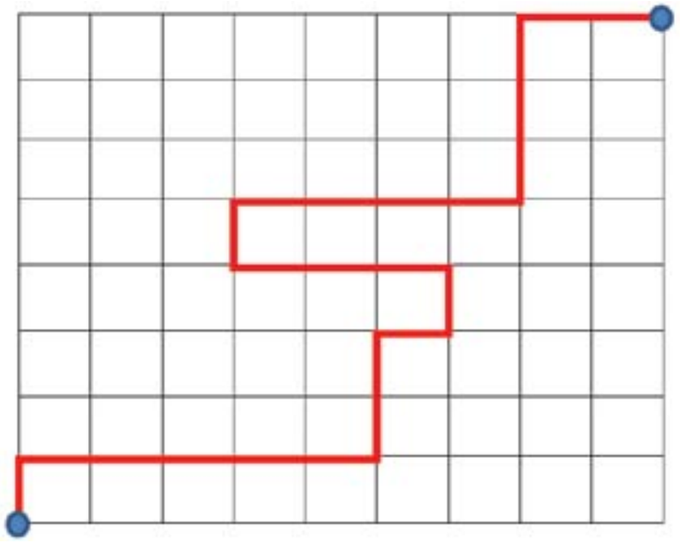
Djecu najviše privlači mogućnost crtanja u Logu. Na pristupačan i zabavan način Logo uvodi u složen proces programiranja i algoritamskog načina razmišljanja. Osim toga, učeći Logo, djeca jednostavno, uglavnom vizualizacijom, kroz igru i zabavu, ponavljaju i usvajaju mnoge matematičke pojmove. U narednim ću poglavljima opisati kako su djeca uz moje upute usvajala programski jezik Logo.

1.1.1 Uvod u programiranje

Djeca su u svakodnevnom kontaktu s računalom i to uglavnom igrajući neku od igara na računalu. Pitala sam djecu znaju li kako igre nastaju, tko ih je izradio i imaju li ideju za vlastitu igru te dobila mnoštvo interesantnih i zabavnih odgovora.

Uvodna igra (na školskom igralištu ili u učionici):
Na školskom igralištu nacrtaju se ili označe dvije točke koje predstavljaju start i cilj. Odabere se jedan učenik kojeg učitelj osnovnim naredbama navodi od starta do cilja: idi **naprijed** dva koraka, okreni se **desno**, okreni se **lijevo**, vrati se **nazad** tri koraka). Slijedeći upute učenik ujedno crta kredom svoj put. Nakon što nekoliko učenika nacrta put i drugi učenici usmenim uputama navode ostale učenike da nacrtaju svoj put od starta do cilja.

U učionici učenici opisuju put od starta do cilja koristeći se riječima: naprijed, desno i lijevo



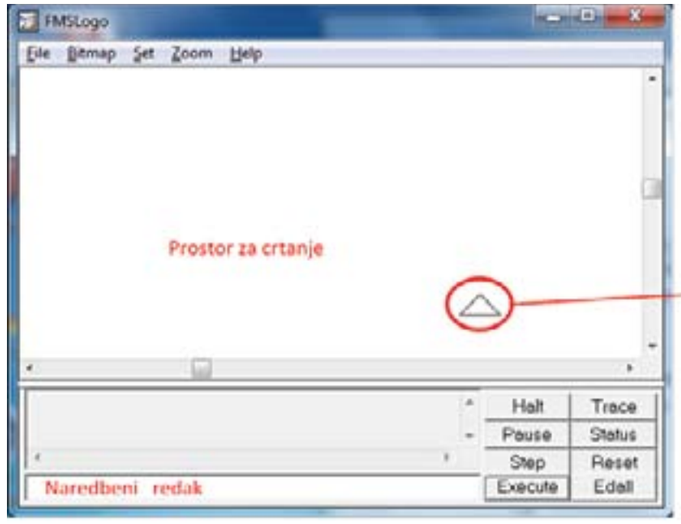
NAPRIJED 1
DESNO
NAPRIJED 5
LIJEVO
NAPRIJED 2
DESNO
NAPRIJED 1
LIJEVO
NAPRIJED 3
DESNO
NAPRIJED 1
DESNO
NAPRIJED 5
LIJEVO
NAPRIJED 3
DESNO
NAPRIJED 2

Ovim postupkom opisali su put od starta do cilja, odnosno napisali algoritam. Algoritam je precizno opisan način rješavanja nekog problema. Kad računalu zadajemo neki algoritam, moramo znati jezik koji računalo „razumije". Takvi jezici se zovu programski jezici. Kao što se svaki strani jezik sastoji od riječi, tako i programski jezik sadržava riječi, rečenice i pravila pisanja (pravopis). Riječi u programskom jeziku zovu se naredbe. Opis algoritma pomoću niza naredbi napisanih po pravilu pisanja u određenom programskom jeziku zove se program.

Logo je samo jedan od mnogih programskih jezika. Neki od ostalih programskih jezika su BASIC, C, C++, i drugi. Programski jezik Logo ima mnogo različitih inačica – PC Logo, MSWLogo, FMSLogo, Terapin Logo i druge. Osnovne naredbe koje će ovdje biti opisane koriste se svim inačicama, a ponegdje se razlikuju samo u načinu pisanja.

1.1.2 Osnovne naredbe za crtanje

Pokrenimo Logo! Otvara se prozor:



U naredbeni redak upisujemo naredbe. Naredbe u programskom jeziku Logo su pokrate i kratice engleskih riječi koje opisuju tu naredbu. Naredbe se u naredbeni redak mogu upisati malim ili velikim slovima. U jednom retku može se napisati i nekoliko naredbi koje moraju biti odvojene razmakom. Nakon što smo upisali naredbu, pritiskom na tipku Enter izvršavamo upisanu naredbe.

Evo nekoliko osnovnih naredbi i njihovog značenja. Upišite ih u naredbeni redak:

HT (HideTurtle – sakrij kornjaču): Trokut je nestao iz prostora za crtanje. U prvoj inačici Loga umjesto trokuta bila je nacrtana kornjača.

ST (ShowTurtle – pokaži kornjaču): Trokut se pojavio u prostoru za crtanje.

FD 100 (ForwarD – naprijed): Trokut se pomaknuo naprijed za 100 točkica (piksela). Nacrtao je crtu duljine 100 točkica.

CS (ClearScreen – obriši ekran): Prostor za crtanje je obrisan, a trokut se ponovo nalazi u početnom položaju („gleda" prema gore).

BK 100 (Back – natrag): Trokut se pomaknuo natrag za 100 točkica. Nacrtao je crtu duljine 100 točkica.

Pomicanjem trokuta naprijed ili natrag za određeni broj točkica trokut se pomiče u zadanom smjeru i ostavlja trag (crta po prostoru za crtanje).

RT 90 (RighT – desno): Trokut se okreće udesno za 90°.

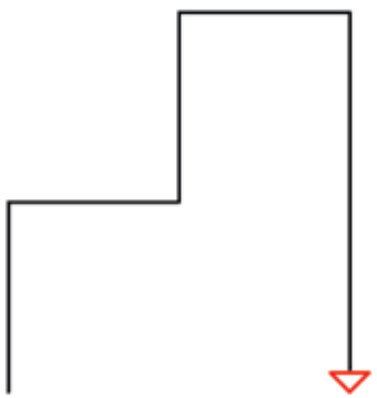
LT 90 (LefT – lijevo): Trokut se okreće ulijevo za 90°.

U točki u kojoj se nalazi, naredbama **RT** i **LT**, okrećemo trokut udesno ili ulijevo za zadani okret izražen u stupnjevima. Učenicima je za početak dovoljno objasniti da puni okret iznosi 360, ne moramo ni spominjati kutove jer se mjera kuta iz matematike uči tek u drugom polugodištu 5. razreda.

Vježba 1: Upišimo sljedeće naredbe u naredbeni redak jednu po jednu i poslije svake naredbe pritisnimo **Enter** i objasnimo što se promijenilo nakon svakog koraka.

```
cs
fd 100
rt 90
fd 100
lt 90
fd 100
rt 90
fd 100
rt 90
fd 200
```

Crtež na kraju treba izgledati ovako:



Vježba 2: Pomoću osnovnih naredbi za crtanje u programskom jeziku Logo nacrtaj kvadrat stranice duljine 200 točkica.

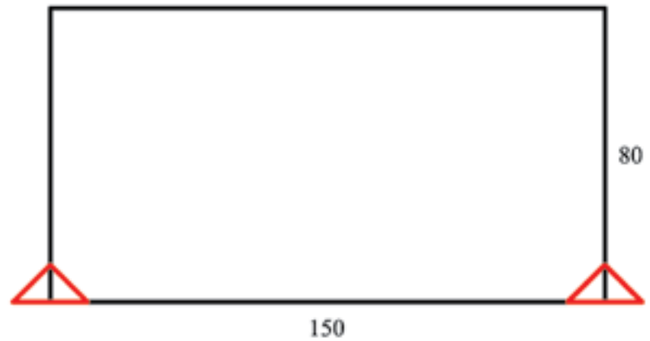
Opis algoritma:

Kvadrat je četverokut koji ima sve stranice jednake duljine i sve prave kutove. Koristeći se definicijom kvadrata i crtanjem kvadrata na papiru učenici opisuju algoritam crtanja. Dajemo slijedeće upute učenicima: „Nacrtaj stranicu duljine 200 točkica, okreni se za pravi kut, nacrtaj stranicu duljine 200 točkica, okreni se za pravi kut, nacrtaj stranicu duljine 200 točkica, okreni se za pravi kut, nacrtaj stranicu duljine 200 točkica."

Pisanje naredbi u naredbeni redak:

```
fd 200
rt 90
fd 200
rt 90
fd 200
rt 90
fd 200
rt 90
```

Vježba 3: Pomoću osnovnih naredbi za crtanje u programskom jeziku Logo nacrtaj pravokutnik čije su duljine susjednih stranica 150 i 80 točkica. Vježba se može zadati i crtanjem pravokutnika na ploču.



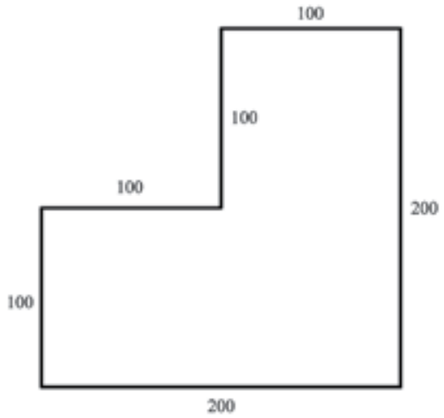
Napomena: Ako učenike pitamo, gdje sve može biti početak crteža, potaknut ćemo ih da uoče različite mogućnosti.

Vježba 4: U naredbeni redak upišite naredbe:

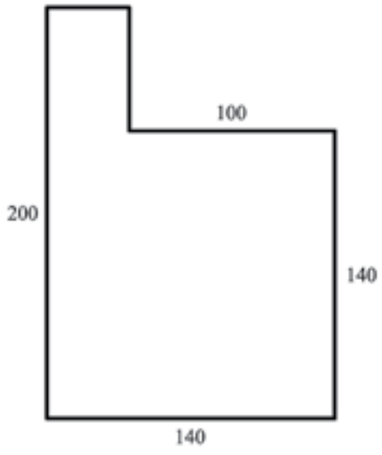
```
rt 90
fd 200
lt 90
fd 100
lt 90
home
```

Naredba **Home** vraća trokut u početni položaj u prostoru za crtanje.

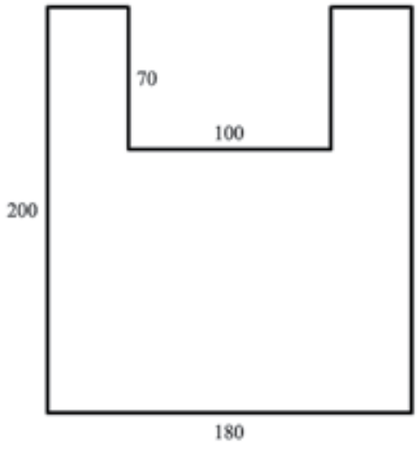
Vježba 5: Pomoću osnovnih naredbi za crtanje u programskom jeziku Logo nacrtaj likove na slikama:



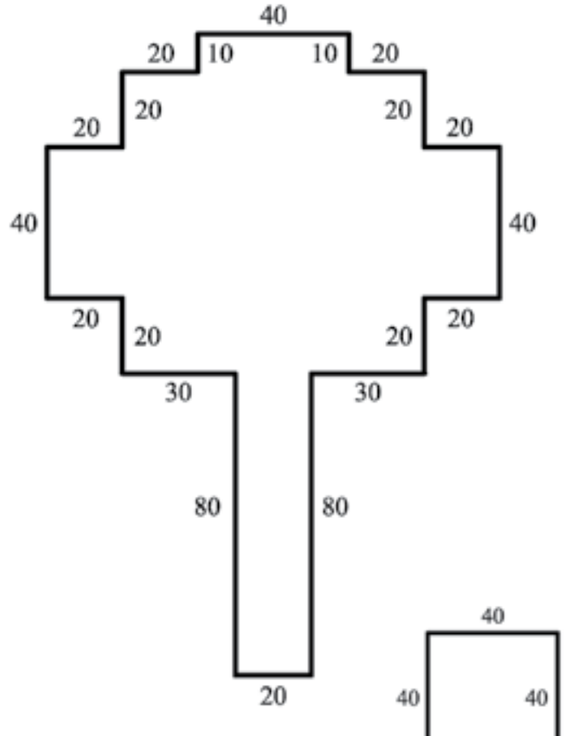

```
fd 100
rt 90
fd 100
lt 90
fd 100
rt 90
fd 100
rt 90
fd 200
rt 90
fd 200
```

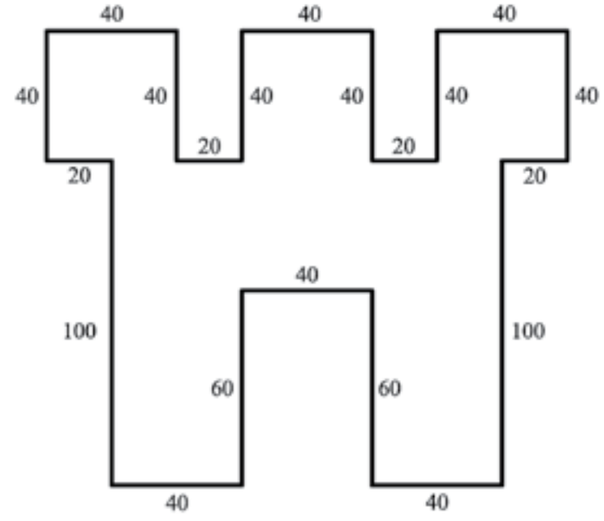


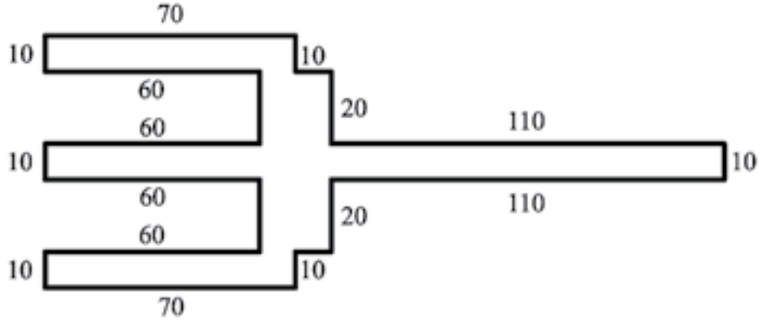

```
fd 140
lt 90
fd 100
rt 90
fd 60
lt 90
fd 40
lt 90
fd 200
lt 90
fd 140
```




```
fd 200 rt 90
fd 40 rt 90
fd 70 lt 90
fd 100 lt 90
fd 70 rt 90
fd 40 rt 90
fd 200 rt 90
fd 180
```

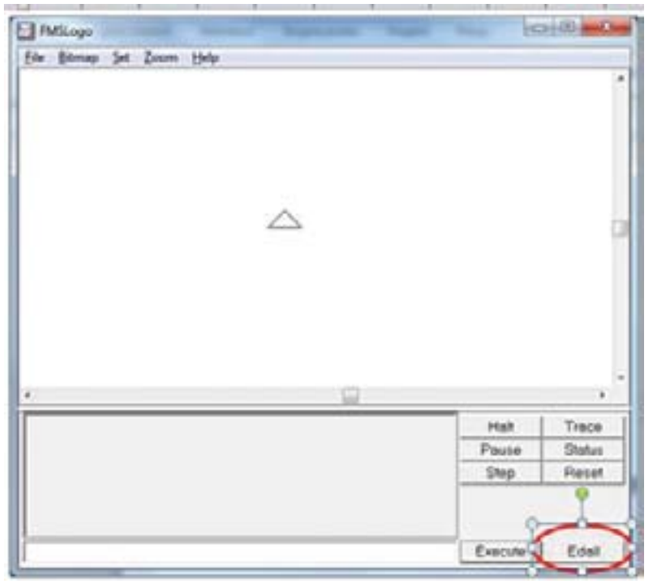




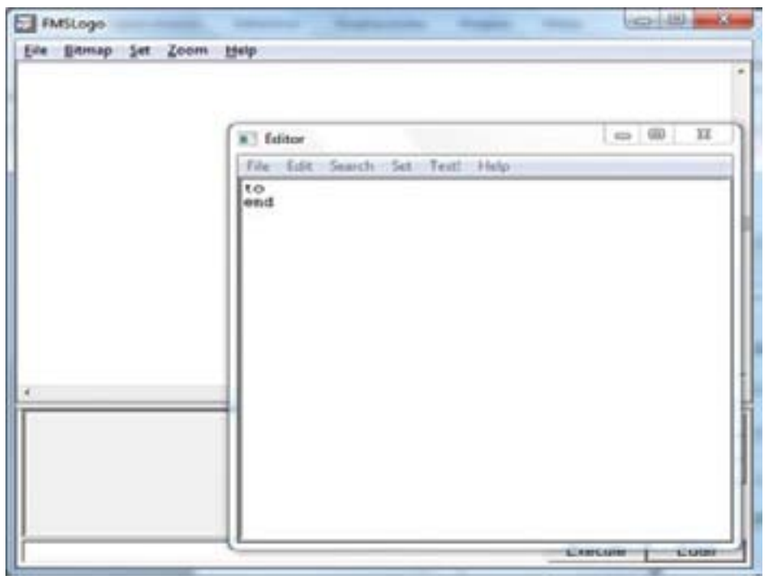


1.1.3 Pisanje programa u editoru

Motivacija: Ako smo pogriješili u pisanju neke naredbe, nacrtana crta nije bila one duljine ili položaja koji nam nije bio namjera. Te smo greške mogli ispraviti tako da smo obrisali ekran naredbom CS i sve pisali ispočetka. No, postoji način da ispravimo samo ono što smo pogriješili.

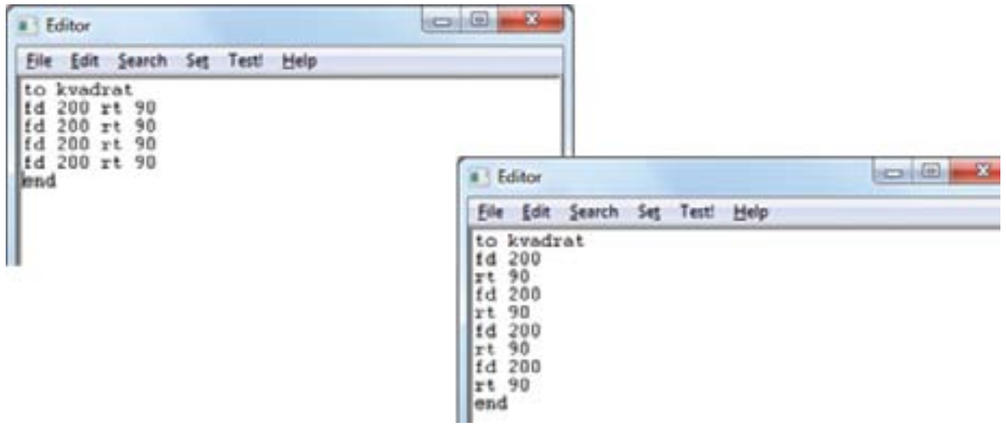


Pritiskom na tipku **Edall** u donjem desnom kutu prozora otvara se novi prozor koji nazivamo *editor*. U *editoru* pišemo programe.



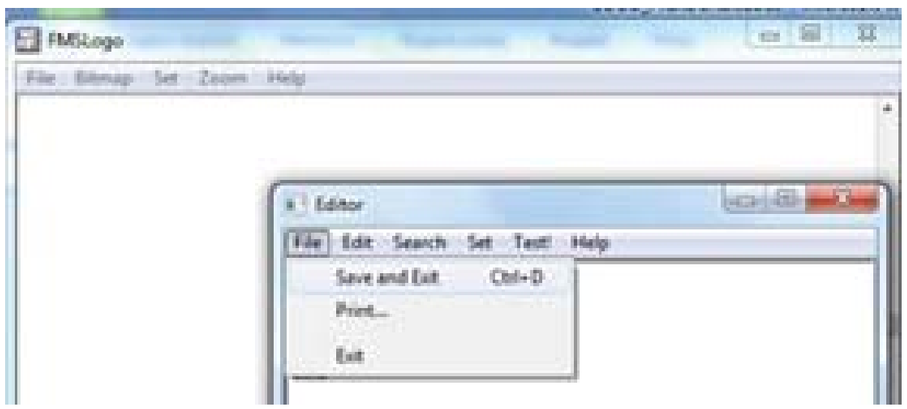
U *editoru* već postoje dvije naredbe: ***To*** i ***End***. Naredbom ***To*** počinje svaki program u programskom jeziku Logo, a naredbom ***End*** završava svaki program. Iza naredbe ***To*** treba upisati ime programa. U imenu programa ne smiju se nalaziti slova č,ć,ž,š kao ni razmaci! Ime programa možemo pisati i velikim i malim slovima.

Vježba 1: Napiši program KVADRAT koji će nacrtati kvadrat stranice duljine 200 točkica.

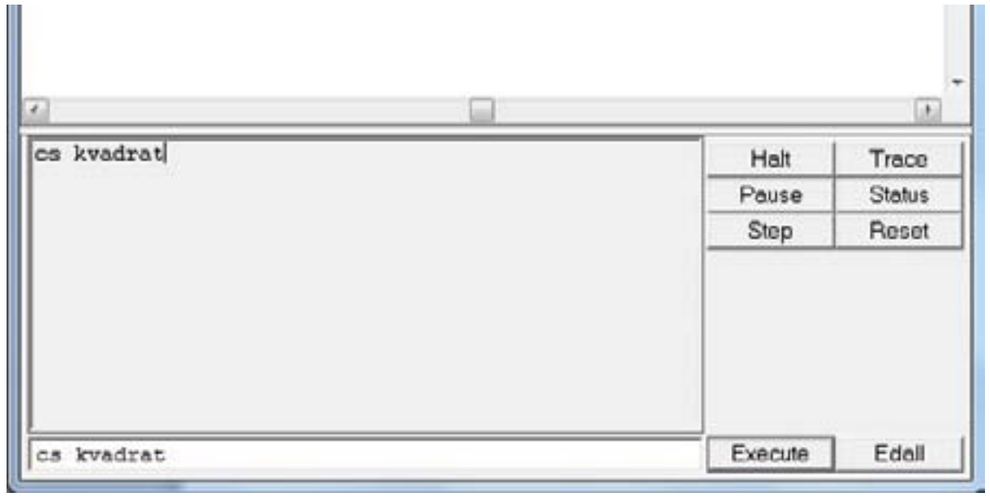


Učenici pomoću uputa pišu svoj prvi program. U *editoru* iza naredbe ***To*** upisujemo ime programa KVADRAT. U sljedećim recima pišemo naredbe koje su potrebne za crtanje kvadrata. Napomenimo da u istom retku može biti i više naredbi odvojenih razmakom. Također napomenimo da u jednom retku trebaju biti naredbe koje crtaju jedan dio slike.

Program je napisan i treba ga testirati. Iz trake izbornika izaberemo **File>Save and Exit** kako bismo izašli iz *editora* i vratili se u prozor u prostor za crtanje.



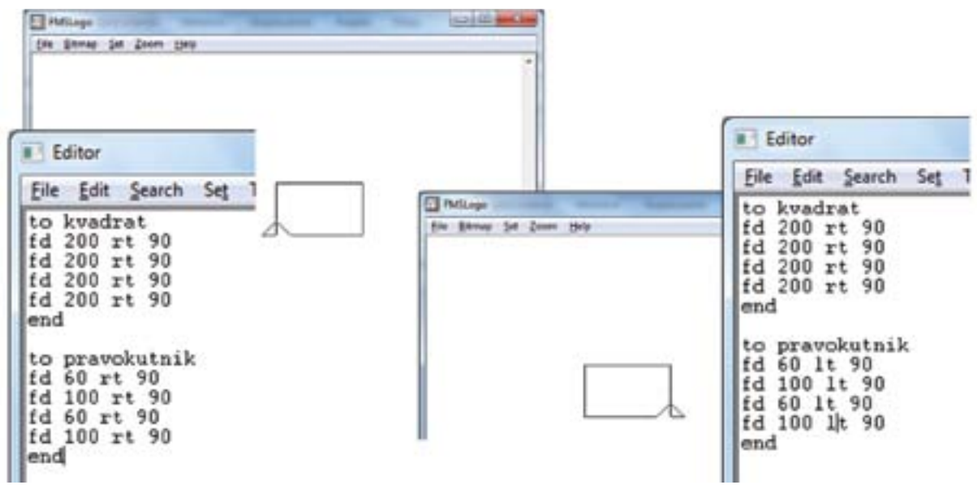
U naredbeni redak upisujemo ime programa. Interpretator programskog jeziks, naš program sada smatra naredbom. Kako nam se ne bi miješale prethodne slike, prije pokretanja programa obrišimo ekran naredbom **cs**.



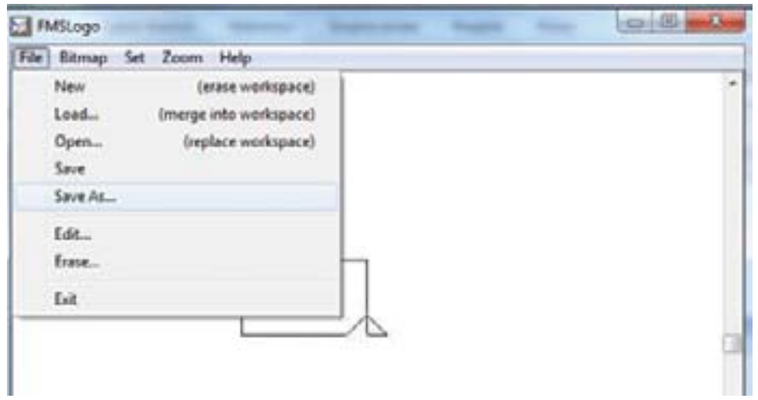
Pritiskom na tipku **Enter** pokrećemo program s imenom **kvadrat**. U prostoru za crtanje nacrtan je kvadrat stranice duljine 200 točkica.

Vježba 1: U istom *editoru* napiši program **pravokutnik** koji crta pravokutnik čije su susjedne stranice duljina 100 i 60 točkica.

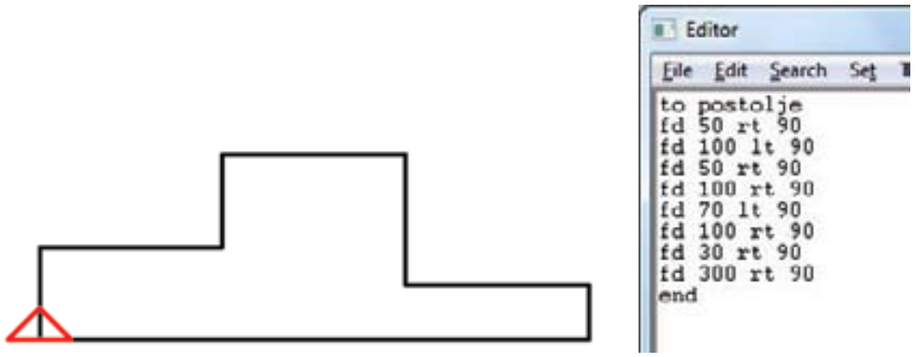
Jedna grupa učenika crta/piše program tako da je trokut na početku crtanja u donjem lijevom vrhu pravokutnika, a druga grupa tako da je trokut u donjem desnom vrhu pravokutnika. Nakon završetka pisanja programa i testiranja učenici uspoređuju programe, uočavaju razlike u programu i crtežu.



Nakon testiranja programa učenici spremaju napisane programe u svoje mape. Programi se spremaju kao dokumenti s nastavkom *.lgo* te otvaraju novi dokument.



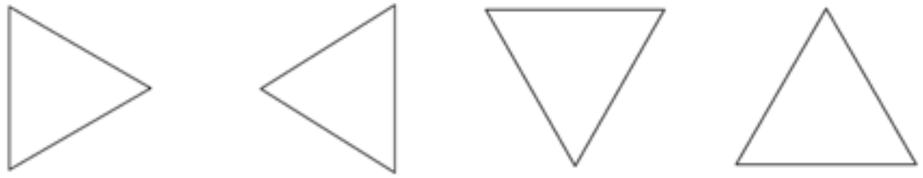
Vježba 2: Napiši program POSTOLJE koji će nacrtati crtež kao na slici. Učenici trebaju nacrtati crtež na papiru s kvadratićima i duljine dužina odrediti sami tako da jedana duljina stranice jednog kvadratića bude 10 (15 ili 20) točkica.



Za domaću zadaću može se zadati da učenici na papiru s kvadratićima nacrtaju crtež za koji će napisati program u programskom jeziku Logo.

1.1.4 Crtanje jednakostraničnog trokuta

Jednakostranični trokut je trokut koji ima sve stranice jednake duljine i kutove jednake veličine. Jednakostranični trokut može se nacrtati u nekoliko položaja.

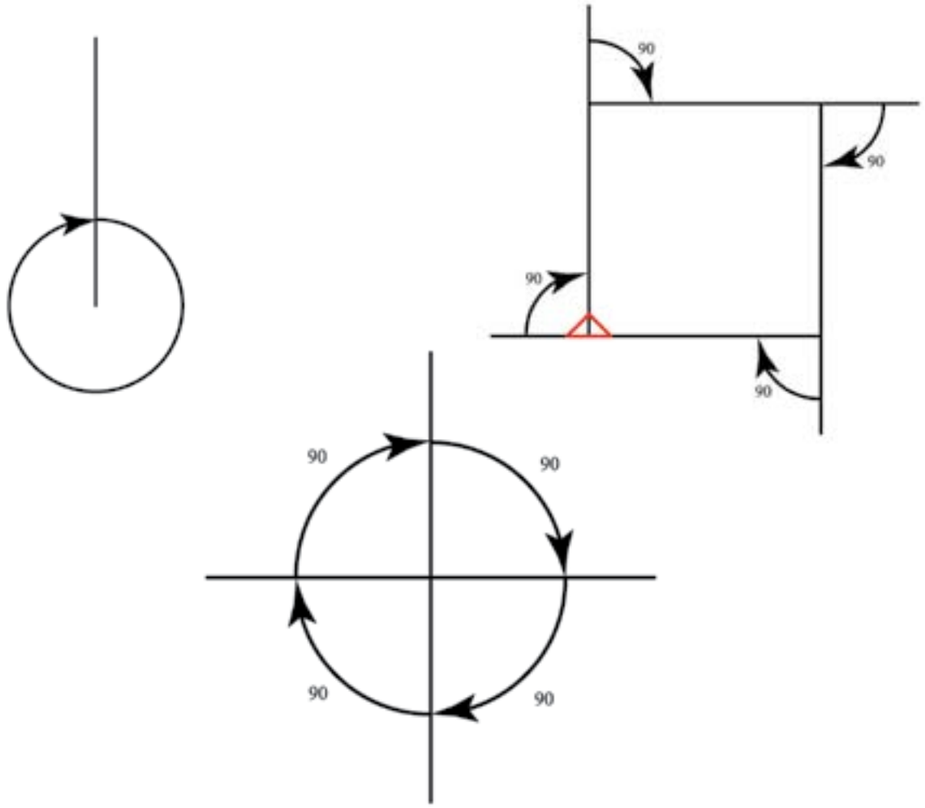


trokut 1 trokut 2 trokut 3 trokut 4

Napišimo program **TROKUT1** koji će nacrtati jednakostranični trokut na slici.

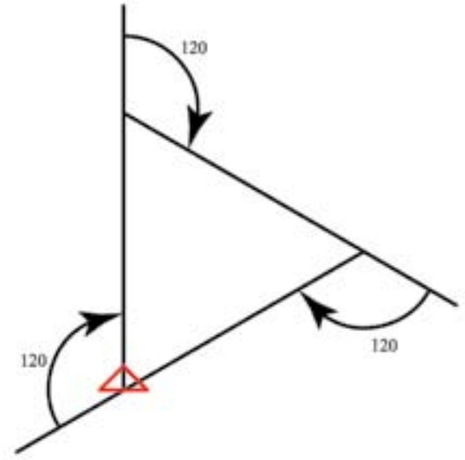
Prije samog pisanja programa u diskusiji s učenicima opisujemo algoritam. Određujemo koje ćemo naredbe rabiti, kako ćemo što nacrtati i koje ćemo okrete upotrijebiti.

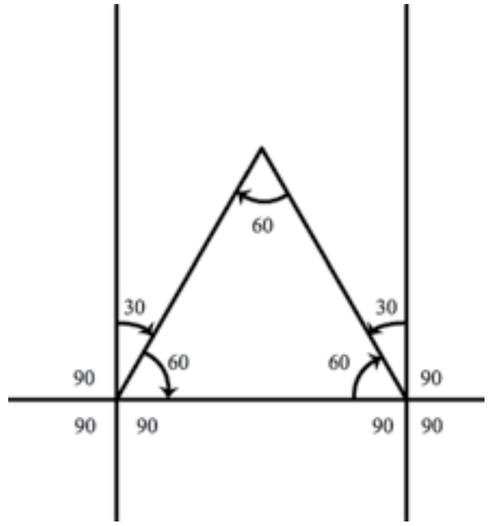
PUNI OKRET: 360



Algoritam za crtanje jednakostraničnog trokuta:

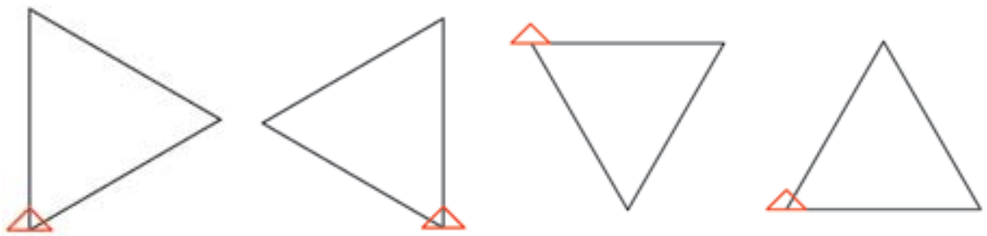
Nacrtaj stranicu zadane duljine, okreni za kut.
Nacrtaj stranicu zadane duljine, okreni za kut.
Nacrtaj stranicu zadane duljine, okreni za kut.





Svi su okreti iste veličine, a okrenuli smo se za puni kut (360). Svaki okret je dakle 360 : 3 = 120

Napišimo programe za trokute na slikama čije su stranice duljine 100 točkica.



to trokut1
fd 100 rt 120
fd 100 rt 120
fd 100 rt 120
end

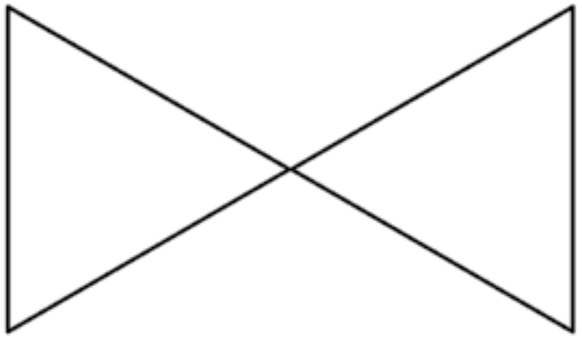
to trokut2
fd 100 lt 120
fd 100 lt 120
fd 100 lt 120
end

to trokut3
rt 90
fd 100 rt 120
fd 100 rt 120
fd 100
rt (120-90)
end

to trokut4
rt 90
fd 100 lt 120
fd 100 lt 120
fd 100 lt 120
lt 30
end

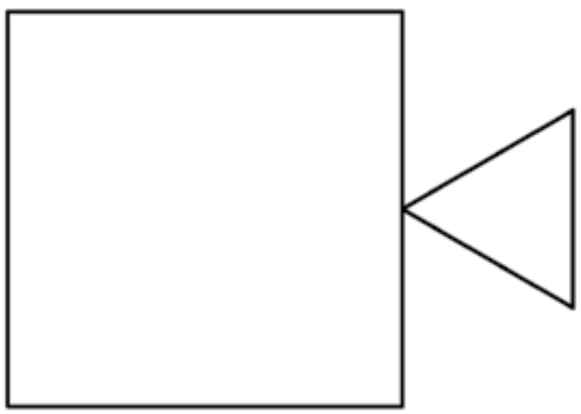
Vježba 1:

Napiši programe koji će nacrtati sljedeće slike:



Ime programa: **LEPTIR**
Duljina stranice trokuta: 200

Ime programa: **ZNAK**
Duljina stranice trokuta: 50
Duljina dužine: 150



Ime programa: **SALICA**
Duljina stranice kvadrata: 140
Duljina stranice trokuta: 70

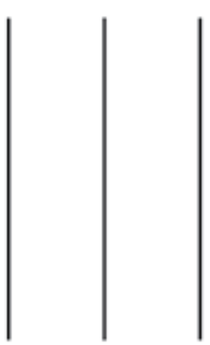
1.1.5 Bez traga

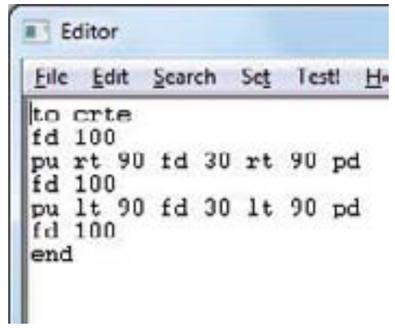
Kako nacrtati isprekidanu crtu?

U programskom jeziku postoji naredba PU (PenUp – digni olovku). Ta naredba naređuje našem trokutu da se podigne i ne ostavlja trag. Kad ponovo želimo da naš trokut ostavlja trag, upotrebljavamo naredbu PD (PenDown – spusti olovku).

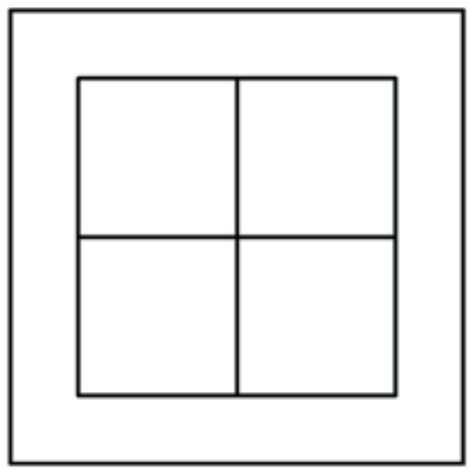
Učenici upisuju u naredbeni redak:
rt 90 fd 50 pu fd 50 pd fd 50

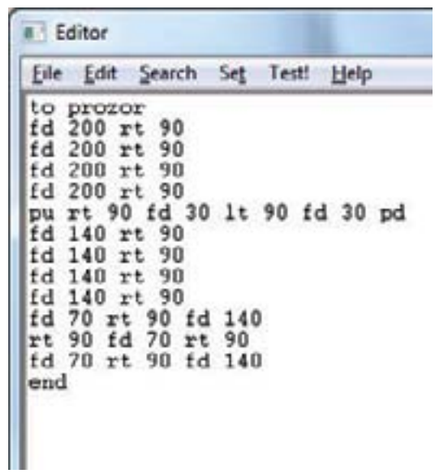
Vježba 1: Napiši program **CRTE** koji će nacrtati tri dužine duljine 100 točkica međusobno razmaknute 30 točkica.





Vježba 2: Napiši program PROZOR koji će nacrtati dva kvadrata jedan unutar drugoga i dužine unutar kvadrata kao na slici. Duljina stranice vanjskog kvadrata je duljine 200 točkica, a duljina stranice unutarnjeg kvadrata je 140 točkica





1.1.6 Potprogrami u programskom jeziku Logo

Zadatak za učenike: U istom editoru napišite programe prav, kv i trok za crtanje sljedećih likova:

prav **kv** **trok**

Važno je napomenuti učenicima da obrate pažnju gdje i u kojem položaju se nalazi trokut kojeg crtamo (neka trokut kojeg crtamo na kraju bude u početnom položaju).

Jednom napisani programi za programski jezik postaju naredbe pa ih možemo rabiti za pisanje novih programa.

Napišimo program PIRAMIDA koji crta tri kvadrata stranice duljine 50 točkica kao na slici:

Algoritam crtanja:

Nacrtaj kvadrat (naredba: **kv**)

Gdje se nalazi trokut za crtanje i kako je okrenut? Gdje treba početi crtanje drugog kvadrata?

Pomakni trokut u novi položaj (naredbe: **rt 90 fd 50 lt 90)**.

Gdje se nalazi trokut za crtanje i kako je okrenut? Gdje treba početi crtanje drugog kvadrata?

Nacrtaj kvadrat (naredbe: **kv**)

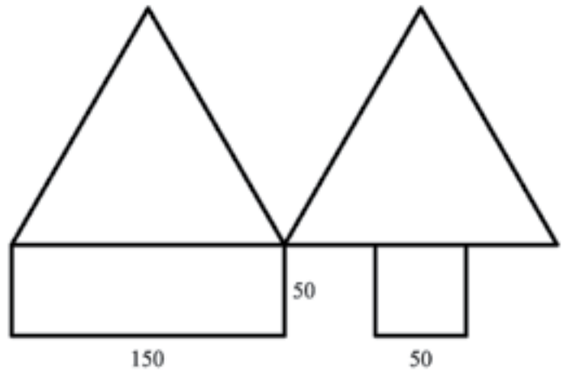
Gdje se nalazi trokut za crtanje i kako je okrenut? Gdje treba početi crtanje trećeg kvadrata?

Pomakni trokut u novi položaj (naredbe: **fd 50 lt 90 fd 25 rt 90**)

Nacrtaj kvadrat (naredba: **kv**)

U programu piramida koristimo se programom **kvadrat** pa kažemo da je program **kvadrat** potprogram programa **piramida**. U pisanju programa možemo upotrijebiti jedan ili više potprograma. Svi potprogrami kojima se koristimo u pisanju programa moraju biti napisani u istom *editoru*.

Napišimo zajedno program **KUCE** koji će nacrtati sliku:

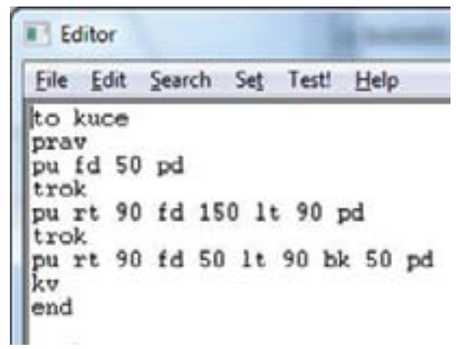



```
to kv
fd 50 rt 90
fd 50 rt 90
fd 50 rt 90
fd 50 rt 90
end

to prav
fd 50 rt 90
fd 150 rt 90
fd 50 rt 90
fd 150 rt 90
end

to trok
rt 90
fd 150 lt 120
fd 150 lt 120
fd 150 lt 120
lt 90
end
```

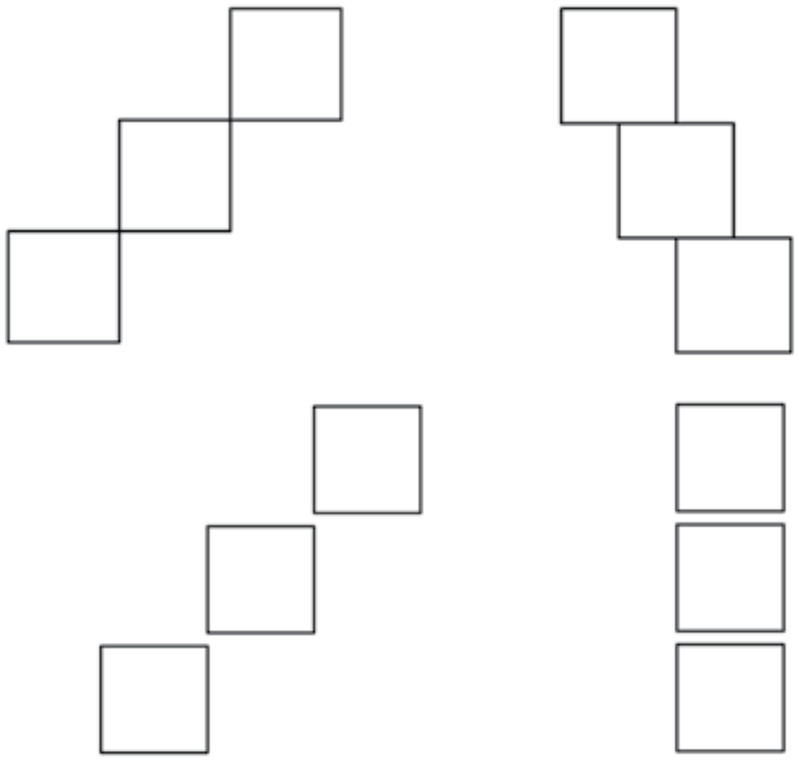
Algoritam crtanja:

(Učenici slažu sliku od postojećih elemenata, ne crta se dužina po dužina.)

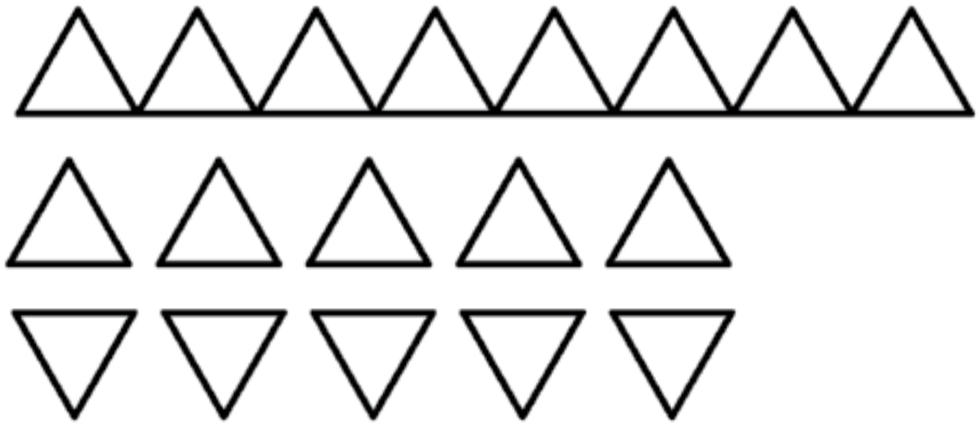
Nacrtaj pravokutnik.
Pomakni trokut za crtanje u novi položaj.
Nacrtaj trokut.
Pomakni trokut za crtanje u novi položaj.
Nacrtaj trokut.



Napiši programe koji će nacrtati jednake kvadrate kao na slici. Duljina stranice kvadrata je 50. Programe napiši rabeći potprogram za crtanje kvadrata. Razmaci između kvadrata su duljine 20. (Učenici sami daju nazive programima.)



Napiši program **ZUBI** koji će nacrtati nizove jednakostraničnih trokuta kao na slici. Duljina stranice trokuta je 40. Programe napiši rabeći potprogram za crtanje jednakostraničnog trokuta. Razmak između trokuta je 15.

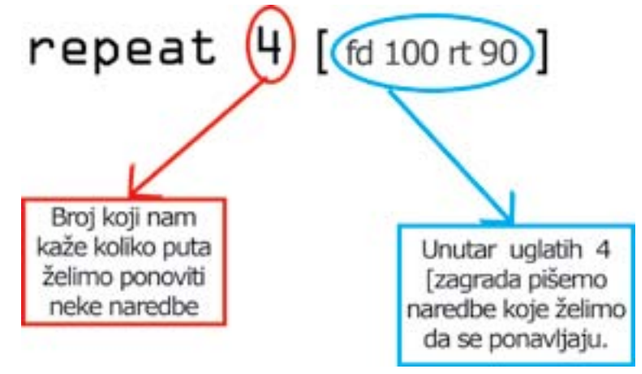


1.1.7 Ah, ta ponavljanja! Upotreba petlji

Napišimo ponovo program **KV** za crtanje kvadrata stranice duljine 100:

```
to kv

fd 100 rt 90
fd 100 rt 90
fd 100 rt 90
fd 100 rt 90

end
```

U programu se odmah uočava da se naredbe **fd 100 rt 90** ponavljaju četiri puta. Da ne bismo ponavljali pisanje istih naredbi, možemo upotrijebiti naredbu za ponavljanje. U programiranju se naredbe za ponavljanje zovu petlje. U svakom programskom jeziku postoji nekoliko vrsta petlja. Jedna od naredbi za ponavljanje u Logu je **REPEAT** petlja. Evo kako se piše:



Lijevu uglatu zagradu ([) dobivamo kombinacijom tipki **Alt gr** i **F**.
Desnu uglatu zagradu (]) dobivamo kombinacijom tipki **Alt gr** i **G**.
Program **kv** za crtanje kvadrata možemo napisati ovako:

```
to kv

   repeat 4 [fd 100 rt 90]

end
```

Evo još jednog primjera korištenja **REPEAT** petlji.

```
to trokut

fd 200 rt 120
fd 200 rt 120
fd 200 rt 120

end
```

ili

```
to trokut

repeat 3 [fd 200 rt 120]

end
```

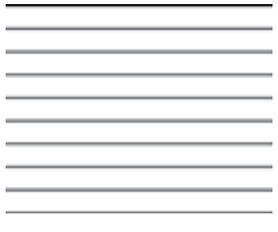
Što će nacrtati program **UPITNIK**? (Učenici na papiru crtaju sliku.)

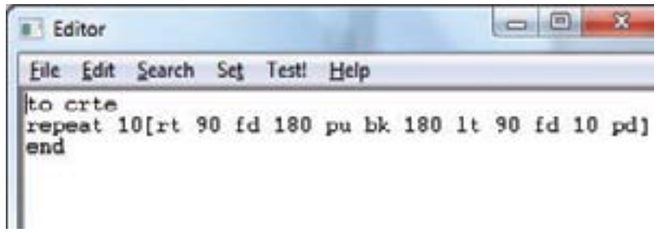
Kako ćemo isti program napisati pomoću naredbe **REPEAT**?

```
to UPITNIK

fd 30 pu fd 30 pd
fd 30 pu fd 30 pd
fd 30 pu fd 30 pd
fd 30 pu fd 30 pd
fd 30 pu fd 30 pd

end
```

```
to upitnik

repeat 5 [fd 30 pu fd 30 pd]

end
```

Napišimo program **CRTE_1** koji će nacrtati 10 crta duljine 180 točkica paralelnih s donjim rubom ekrana međusobno razmaknutih 35 točkica. Zadatak možemo zadati tekstom ili slikom.





Algoritam crtanja:

Uočimo ponavljanje!
Koji dio crteža se ponavlja?

Nacrtaj crtu duljine 180 paralelnu s donjim rubom ekrana (naredbe: **rt 90 fd 180**).
Pomakni trokut u novi položaj za crtanje sljedeće crte (naredbe: **pu bk 180 lt 90 fd 10 pd**) .
Pomoću **REPEAT** petlje možemo nacrtati različite nizove likova.

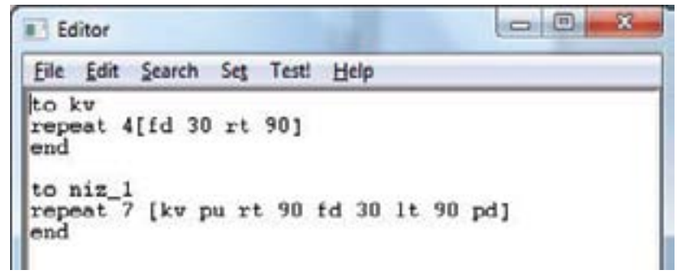
Vježba 1: Napiši program **NIZ_1** koji će nacrtati niz od 7 kvadrata stranice 30 točkica kao na slici:

Što se ponavlja? Kvadrat stranice duljine 30. Napišimo program za crtanje kvadrata.

```
to kv

repeat 4 [fd 30 rt 90]

end
```

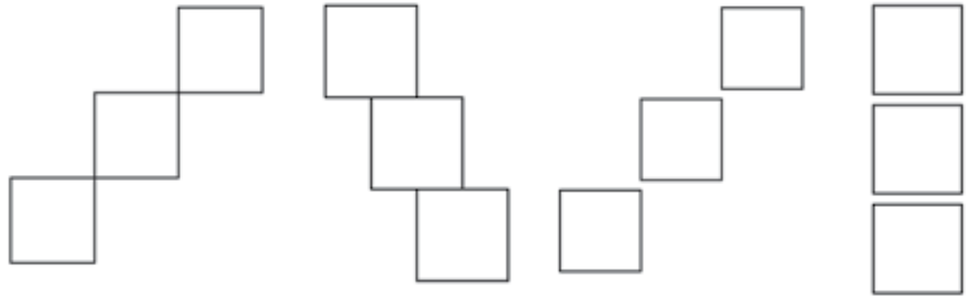

Algoritam crtanja:

Nacrtaj kvadrat (naredba: **kv**)
Pomakni trokut u novi položaj (naredbe: **pu rt 90 fd 30 lt 90 pd**) } Sve ponovi 7 puta

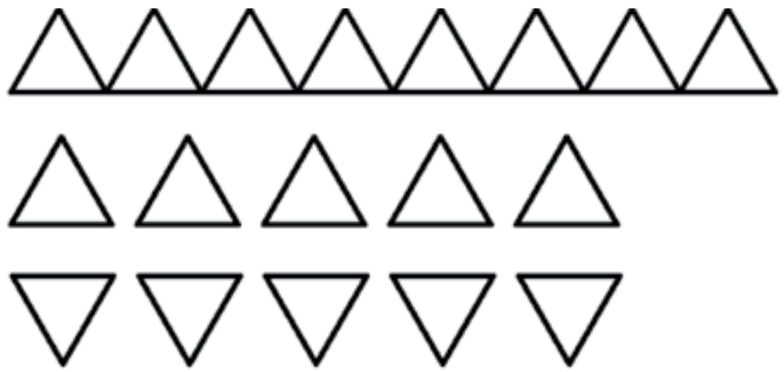


Zadaci za vježbu:

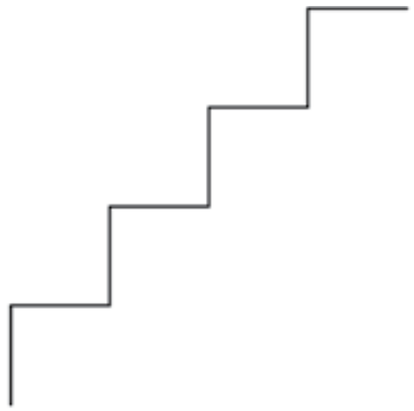
1. Napiši programe koji će nacrtati kvadrate prikazane na slici. Duljina stranice kvadrata je 50. Programe napiši koristeći se potprogramom za crtanje kvadrata. Razmaci između kvadrata su duljine 20. Učenici sami daju nazive programima.



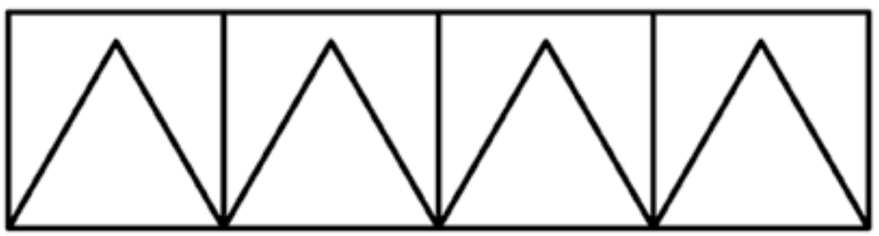
2. Napiši programe koji će nacrtati nizove jednakostraničnih trokuta prikazanih na slici. Duljina stranice trokuta je 40. Programe napiši koristeći se potprogramom za crtanje jednakostraničnog trokuta. Razmak između trokuta je 15.



3. Napiši program **STEPENICE** koji će nacrtati sliku. Duljina svake dužine je 50.



4. Napiši program NIZ koji će nacrtati sliku. Duljina svake dužine je 90.



1.1.8 Crtanje mnogokuta

Do sada smo crtali kvadrate i jednakostranične trokute. Kvadrati i jednakostranični trokuti su vrste pravilnih mnogokuta. Mnogokut je geometrijski lik koji ima tri ili više stranica. Pravilni mnogokut je onaj mnogokut kojemu su sve stranice jednakih duljina i svi unutrašnji kutovi jednake veličine. Geometrijski lik koji ima 5 stranica jednakih duljina zove se pravilni peterokut, geometrijski lik koji ima 6 stranica jednakih duljina je pravilan šesterokut, geometrijski lik koji ima 8 stranica jednakih duljina je pravilan osmerokut, itd...

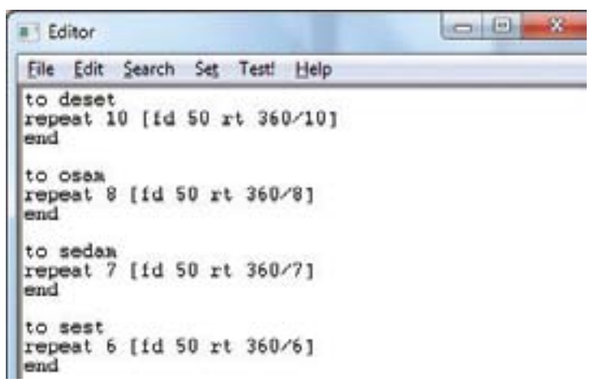
Prisjetimo se kako smo crtali jednakostranični trokut i kvadrat:

to trokut	Zašto je okret 120?
repeat 3 [fd 150 rt 120]	Tri puta smo okretali trokut u istom smjeru. Puni okret je 360, pa je stoga jedan okret 360 : 3 = 120
end	
to kvadrat	Zašto je okret 90?
repeat 4 [fd 150 rt 90]	Četiri puta smo okretali trokut u istom smjeru. Puni okret je 360, pa je stoga jedan okret 360 : 4 = 90
end	

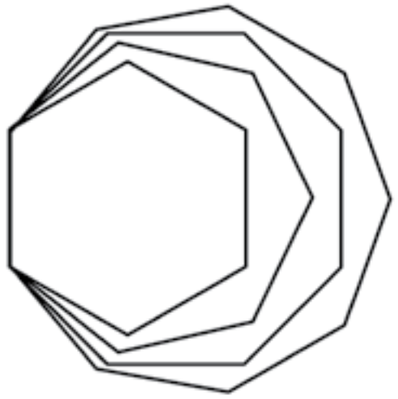
Napišimo program **PETEROKUT** koji će nacrtati pravilan peterokut stranice duljine 100 točkica.

to peterokut	U pisanju programa umjesto da sami izračunamo vrijednost okreta tj. koliko je 360 : 5, možemo to zapisati kao **360/5**. Znak / označava operaciju dijeljenja.
repeat 5 [fd 100 rt 360/5]	
end	

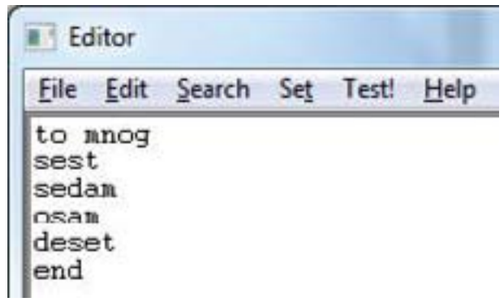
Vježba 1: Napiši programe sest, sedam, osam i deset koji crtaju pravilan šesterokut, sedmerokut, osmerokut i deseterokut stranice duljine 50 točkica.



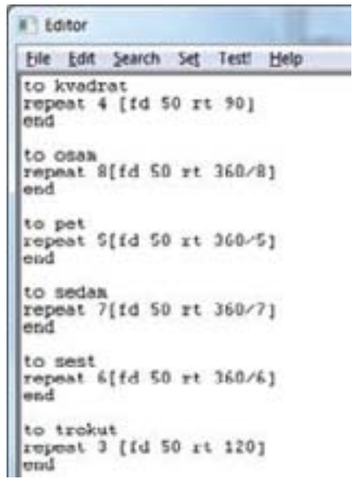
Prvo obriši ekran, a zatim u naredbeni redak upiši redom sest, sedam, osam, deset. Dobivamo sliku:



Vježba 1: Napiši program MNOG koji će nacrtati ovu sliku.

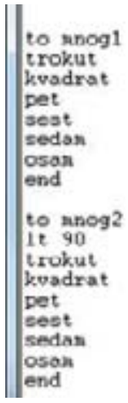


Vježba 2: U istom *editoru* napiši programe **trokut, kvadrat, pet, sest, sedam, osam** koji crta pravilne mnogokute s redom 3, 4, 5 ,6, 7 i 8 stranica duljine 50.



U editoru nakon spremanja programa, programi će biti poredani abecednim redom, a ne po redu kojim smo pisali programe.

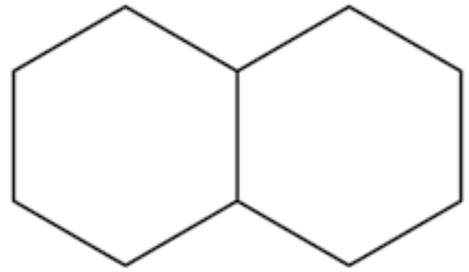
Vježba 3. Koristeći se već napisanim programima napiši programe **MNOG1** i **MNOG2** koji crtaju slike:



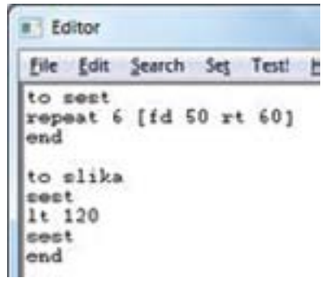
Pravilne mnogokute možemo crtati okrećući geometrijski lik u desnu i u lijevu stranu. Učenicima napišemo program:

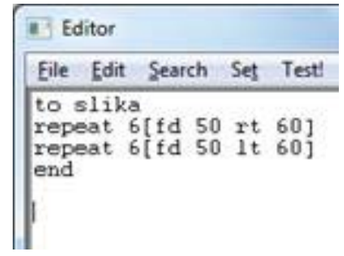
```
to nacrtaj

repeat 3 [fd 40 rt 120]
repeat 3 [fd 40 lt 120]

end
```

Učenici na papiru crtaju što će nacrtati taj program, a zatim provjeravaju svoje crteže upisivanjem programa u editor. Za dodatni sadržaj učenicima se može zadati da nacrtaju dva šesterokuta koja imaju jednu zajedničku stranicu.



Ovaj je zadatak moguće riješiti na više načina. Evo nekih mogućih rješenja





1.1.9 Crtam crtu duljine koje želim

Varijable u programskom jeziku Logo.

Napišimo program crtica koji će nacrtati crtu duljine 50 točkica.

```
to crtica
fd 50
end
```

Učenike tražimo da mijenjaju program kako se mijenja duljina crte. Ako želimo nacrtati crtu druge duljine, moramo ponovo ulaziti u editor te: ili mijenjati program ili pisati novi program. Postoji način da program napišemo tako da možemo samo mijenjati duljinu crte. Učenicima je poznat pojam nepoznanice iz matematike gdje nepoznanica zamjenjuje neki broj i označava se nekim slovom.

U programiranju kad zamjenjujemo broj slovom kažemo da se koristimo varijablom ili ulaznom vrijednošću. U programskom jeziku Logo varijabla (nepoznanica) se piše s dvotočkom ispred slova. Evo kako izgleda program s korištenjem varijable:

```
to crtica :a
fd :a
end
```

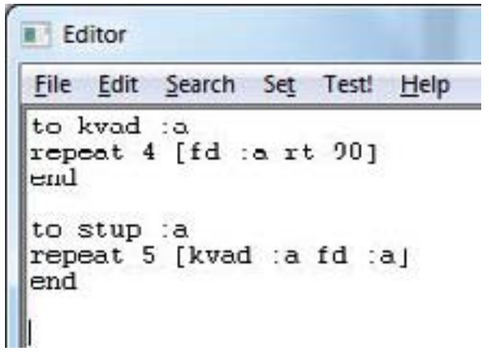
Sad možemo nacrtati crtu bilo koje duljine. U naredbeni redak upisujemo ime programa i određenu zadanu duljinu, npr.: **crtica 50, crtica 100, crtica 30**.
Napišimo program **trok** za crtanje jednakostraničnog trokuta stranice duljine **:a**.

```
to trok :a
repeat 3 [fd :a rt 120]
end
```

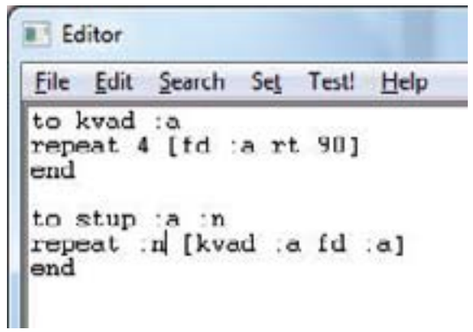
Napiši program **kvad** koji crta kvadrat stranice duljine **:a.**

```
to kvad :a
repeat 4 [fd :a rt 90]
end
```


Napišimo program **STUP** koristeći se programom za crtanje kvadrata koji će nacrtati pet kvadrata jedan iznad drugoga, ako svaki kvadrat ima stranicu duljine **:a**.



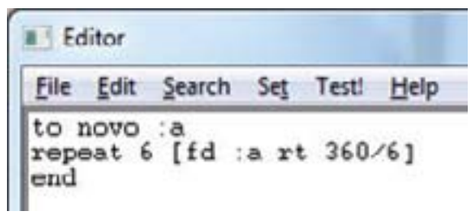
Kako će se mijenjati program ako želimo u stupu nacrtati 3 kvadrata ili 7 kvadrata? U istom programu možemo imati i više varijabli pa za crtanje stupa možemo rabiti drugu varijablu koja će nam određivati broj kvadrata u stupu.



Testirajmo program različitim vrijednostima.

cs stup 20 4
cs stup 30 6
cs stup 50 2

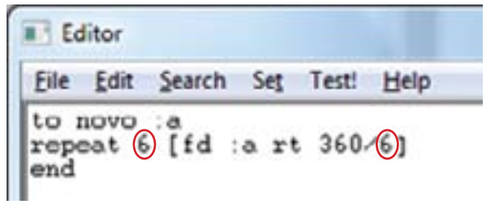
Napišimo program **NOVO** za crtanje pravilnog šesterokuta stranice duljine **:a**.



Testiramo program:

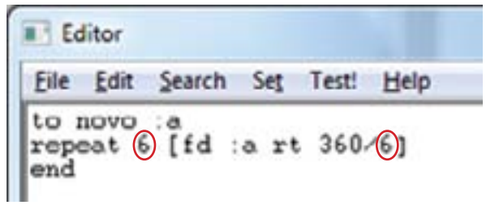
cs novo 50
cs novo 50

Što moramo promijeniti u programu ako želimo da nacrta pravilan peterokut?



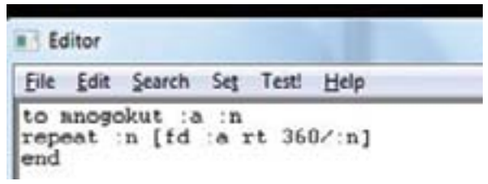
Broj 6 mijenjamo u 5.

Što moramo promijeniti u programu ako želimo da nacrta pravilan osmerokut?



Broj 6 mijenjamo u 8.

Možemo napisati program koji crta bilo koji pravilan mnogokut čija je stranica duljine **:a**. U tome programu upotrebljvamo još jednu varijablu **:n** koja predstavlja broj stranica tog mnogokuta.



Testiramo program:

cs mnogokut 50 5
cs mnogokut 150 6

Zadaci za vježbu:

1. Napiši program **KVADRAT :a** koji će nacrtati kvadrat s duljinom stranice **:a**.
2. Napiši program **PRAVOKUTNIK :a :b** koji će nacrtati pravokutnik s duljinama stranica **:a** i **:b**. Stranica **:a** je paralelna s donjim rubom ekrana.

Test primjeri:

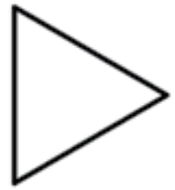
PRAVOKUTNIK 100 50

PRAVOKUTNIK 300 50

3. Napiši program **TROKUT :a** koji će nacrtati trokut s duljinom stranice **:a**.

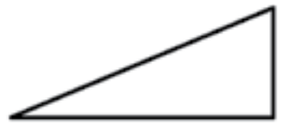
TROKUT 100

TROKUT 200



4. Napiši program **PRAVOKUTNITROKUT :a** :b koji će nacrtati pravokutni trokut sa stranicama **:a** i **:b** kao na slici. Stranica **:a** je paralelna s donjim rubom ekrana.

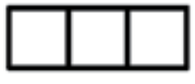
PRAVOKUTNITROKUT 300 100



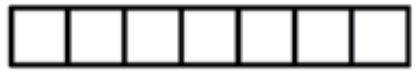
PRAVOKUTNITROKUT 50 200

5. Napiši program **KVADRATI :n** koji će nacrtati niz od **:n** kvadrata s duljinom stranice 40 kao na slici.

KVADRATI 3



KVADRATI 7



6. Napiši program **MAJA :n** :a koji će nacrtati niz od **:n** kvadrata s duljinom stranice **:a** poredanih kao na slici.

MAJA 3 100

MAJA 20 15

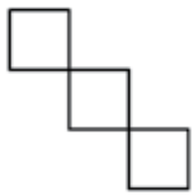
7. Napiši program **NEBODER :n :a** koji će nacrtati niz od **:n** kvadrata s duljinom stranice **:a** poredanih kao na slici.

NEBODER 3 60

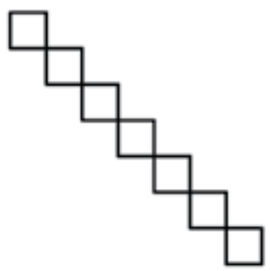
NEBODER 7 30

8. Napiši program **ZAGREB :n :a** koji će nacrtati niz od **:n** kvadrata s duljinom stranice **:a** poredanih kao na slici.

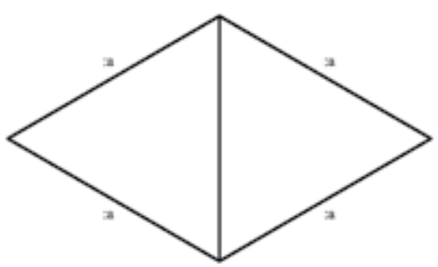
ZAGREB 3 60



ZAGREB 7 30



9. Napiši program **TROKUTI :a** koji će nacrtati dva jednakostranična trokuta kao na slici. Duljina stranica trokuta je **:a**.

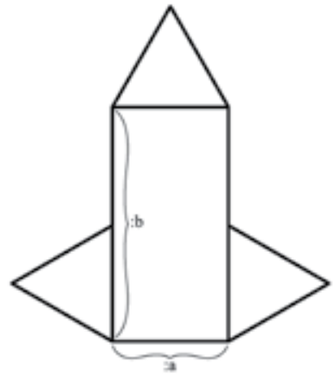


10. Napiši program **PLANINE :n :a** koji će nacrtati niz od **:n** jednakostraničnih trokuta sa stranicama duljine **:a**. Svi trokuti okrenuti su prema gore kao što je prikazano u test primjerima.

cs planine 4 70

cs planine 10 30

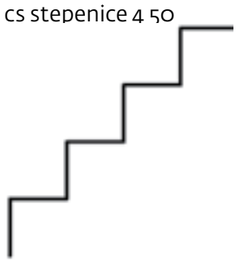
11. Napiši program **RAKETA :a :b** koji će nacrtati jedan pravokutnik i tri jednakostranična trokuta prikazanih na slici. Položaj tih likova mora biti kao na slici. Duljine stranica trokuta su **:a**. Širina rakete je **:a**, a visina rakete je **:b**.



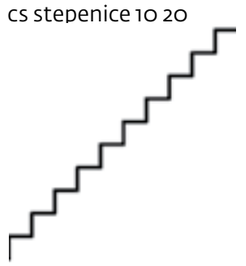
12. Napiši program **STEPENICE :n :a** koji će nacrtati :n stepenica. Visina i širina stepenice je **:a**.

Test primjeri:

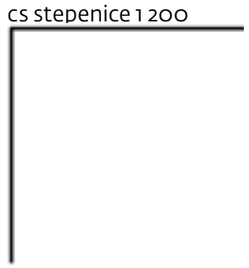
cs stepenice 4 50



cs stepenice 10 20



cs stepenice 1 200



2. poglavlje

INKSCAPE - vektorsko crtanje

6. razred

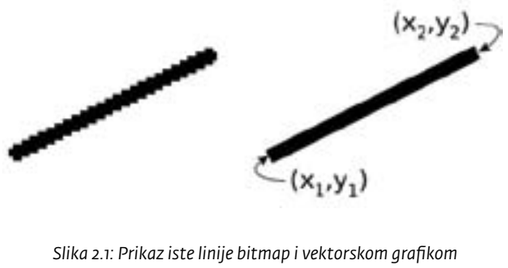
2.0 Inkscape

2.1 Uvod

Začetak programa Inkscape možemo pronaći u programu Gill (*Gnome Illustrator aplication*) autora Rapha Leviana. Taj je program podržan kao projekt od strane programa *SodiPodi*. Cilj autora programa *Inkscape* je bio program koji će u potpunosti moći iskoristiti prednosti SVG standarda. To, naravno, nije lak zadatak, ali ga, uz malo truda, i vi možete uspješno obaviti.

2.2 Vektorska grafika

Postoje dvije osnovne vrste grafičkih datoteka: bitmap (raster) grafičke datoteke i vektorske grafičke datoteke. U slučaju bitmapa slika je definirana recima i stupcima pojedinačnih piksela, gdje svaki piksel ima svoju sliku. Vektorska grafika je definirana ravnim ili zakrivljenim linijama. Razlika između bitmap i vektorske grafike je posebno uočljiva kada se slike znatno povećaju.



Slika 2.1: Prikaz iste linije bitmap i vektorskom grafikom

Kada je razlučivost bitmap grafike jednaka razlučivosti zaslona nacrtani objekt djeluje gladak.

Slika 2.2: Prikaz istog crteža bitmap i vektorskom grafikom (razlučivost slike = razlučivost zaslona)

No kada taj crtež povećate, prikaz bitmap grafike pokazuje zupčaste linije, dok je prikaz vektorske grafike i dalje gladak.

Slika 2.3: Prikaz dijela crteža sa slike 1.bitmap i vektorskom grafikom

SVG (*Scalable Vector Graphics*) koji koristi Inkscape je format zapisa koji se može povećavati/smanjivati bez gubitka detalja do proizvoljne veličine. Skalabilan se odnosi također na mogućnost da u crtež bude ugrađen neograničen broj manjih dijelova koji se mogu koristiti više puta unutar istog crteža.

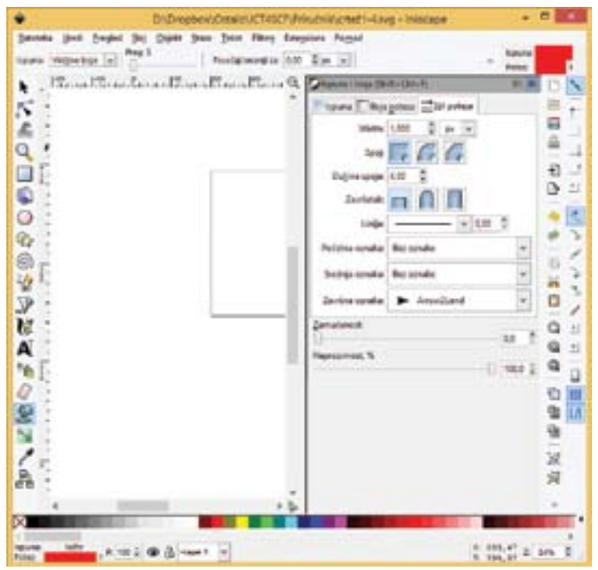
SVG je usmjeren na rad s dvodimenzionalnom grafikom, uključujući i animacije u xml formatu. Datoteke su male, a crteži se mogu prilagoditi različitim načinima prikazivanja. Sve to je dovelo do velikog interesa za format zapisa pa ga danas podržavaju svi važniji internet preglednici kao i pametni telefoni vodećih svjetskih proizvođača.

2.3 Inkscape – programsko sučelje

Pri otvaranju Inkscape programa na zaslonu se prikazuje prozor koji sadrži nekoliko odvojenih područja od kojih mnogi sadrže alate koji se mogu kliknuti ili sadrže padajuće izbornike. Sljedeća slika prikazuje početni prozor i opisuje ključne dijelove.

Slika 2.4: Početni prozor programa Inkscape

Inkscape koristi promjenjive dijaloške okvire koji se smještaju na desnoj strani prozora kao što se vidi na slijedećoj slici.



Slika 2.5: Dockable dijaloški okvir Ispune i linije

2.4 Geometrijski oblici

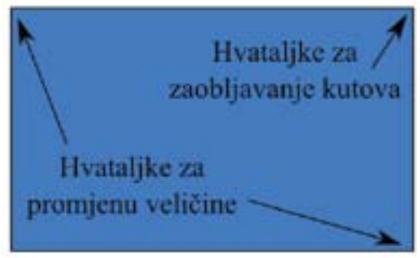
Inkscape osigurava veliki broj alata za crtanje geometrijskih oblika. Alati za crtanje standardnih geometrijskih oblika (pravokutnici, kvadrati, elipse, poligoni, zvijezde i spirale) bit će obrađeni u ovom poglavlju, a alati za rad s krivuljama u slijedećem poglavlju.

2.4.1 Pravokutnici, kvadrati i 3d objekti

Za stvaranje pravokutnika i kvadrata odaberite alat klikom na ikonu na alatnoj traci i:

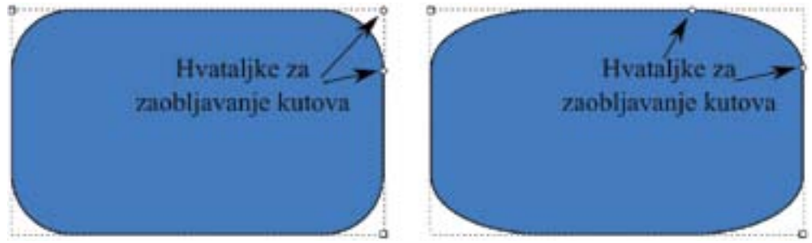
- pritisnite lijevu tipku miša i vucite u suprotni kut,
- želite li nacrtati kvadrat pritisnite tipku Ctrl dok vučete miš,
- želite nacrtati pravokutnik centriran oko točke iz koje smo krenuli crtati držite pritisnutu tipku Shift.

Želite li promijeniti veličinu postojećeg pravokutnika potrebno ga je prvo odabrati lijevim dvoklikom miša. Kada je pravokutnik odabran hvataljke (mali kvadratići) pojavit će se u lijevom gornjem i desnom donjem kutu (slika 1.) pomoću kojih možemo promijeniti veličinu ili možemo unijeti vrijednosti širine i visine pravokutnika u polja **Š:** i **V:** na alatnoj traci (slika 3.)



Slika 2.6: Uređivanje izgleda pravokutnika

Pravokutniku možete napraviti i zaobljene kutove. Dva je načina za to napraviti. Prvi način je pomoću hvataljki (mali kružići) u gornjem desnom kutu pravokutnika (slika 2.), a drugi pomoću polja **Rx** i **Ry** na kontroli alata kada odaberemo alat za pravokutnike (slika 3.).



Slika 2.7: Uređivanje izgleda pravokutnika

Slika 2.8: Alatna traka za uređivanje izgleda pravokutnika

Zaobljene rubove možete ukloniti koristeći alat **Zaoštri kutove** s trake s alatima (slika 8.)

2.4.2 Elipse, kružnice i isječci kružnice

Za stvaranje elipse, krugova i lukova odaberite alat klikom na ikonu i:

- pritisnite lijevu tipku miša i vucite u suprotni kut,
- želite li nacrtati krug pritisnite tipku Ctrl dok vučete miš,
- držeći tipku Shift pritisnutu dok povlačite miš stvarat će se elipsa sa središtem u početnoj točci,
- držeći tipku Alt dok povlačite miš stvarat će se elipsa koja svojim obrubom prolazi kroz početne i završne točke crtanja (gdje ste pritisnuli lijevu tipku miša i gdje ste ju otpustili).

Nakon odabira elipse i ako je alat za crtanje elipse aktivan, elipsa će imati set hvataljki (mali kvadrati i krugovi) koje možete koristiti kako bi se elipsi promijenila veličina ili ju se pretvorilo u luk ili isječak.

Slika 2.9: Uređivanje izgleda elipse

Za pretvaranje elipse u luk, koristite dvije hvataljke za isječak kruga. U početku obje hvataljke su jedna iznad druge. Povucite jednu hvataljku za postavljanje jednog kraj luka, a zatim povucite drugu hvataljku za postavljanje drugog kraja luka. Držite li tipku **Ctrl** dok povlačite hvataljke promjena će se događati u koraku od 15°.

Slika 2.10: Uređivanje izgleda elipse

Isto tako, luk, možete definirati pomoću alata na alatnoj traci kada je alat Elipsa odabran. O postavkama ovisi da li će biti iscrtan isječak elipse ili luk te gdje počinje i gdje završava isti.

Slika 2.11: Alatna traka za uređivanje izgleda elipse

2.4.3 Mnogokuti, zvijezde i spirale

Za crtanje mnogokuta ili zvijezde koristite se alatom , odaberite alat i pritisnite lijevu tipku miša i vucite do vrha mnogokuta ili zvijezde. Objekt će biti nacrtan od središnje točke (mjesto gdje ste pritisnuli miš) prema vrhu. Nakon što odaberete alat crtanje Zvijezde i poligoni pojavljuje se nova alatna traka na kojoj možete odrediti hoće li objekt biti mnogokut ili zvijezda, koliko će vrhova imati, koliki je odnos prečki i kolika je zaobljenost.

Slika 2.12: Alatna traka za uređivanje izgleda mnogokuta/zvijezde

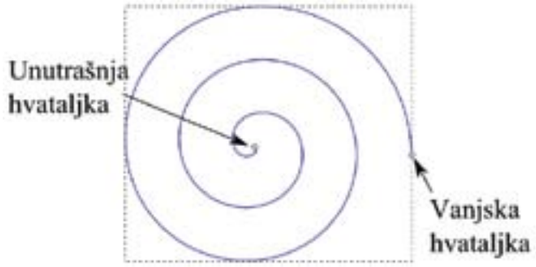
Osim na alatnoj traci izgled zvijezde možemo mijenjati i na samom crtežu koristeći se hvataljkama. Postoji dvije vrste hvataljki (na mnogokutu samo jedna), kojima određujemo položaj vanjskih vrhova zvijezde i položaj unutrašnjih vrhova zvijezde.

Slika 2.13: Uređivanje izgleda mnogokuta/zvijezde

Držite li pritisnutu tipku Ctrl na tipkovnici dok vučete hvataljku za kontrolu unutrašnjih vrhova dobiti ćete pravilnu zvijezdu. Držite li pritisnutu li tipku Shift zaoblit ćete vrhove čiju hvataljku vučete, a držite li pritisnutu tipku Alt pomak vrhova zvijezde ili mnogokuta bit će slučajan.

Slika 2.14: Uređivanje izgleda mnogokuta/zvijezde

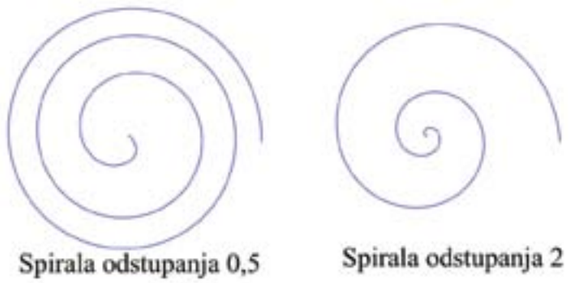
Pomoću alata nacrtajte Arhimedovu spiralu. Spirala će biti nacrtana slično kao zvijezda, od središnje točke (mjesto gdje smo pritisnuli miš) prema kraju spirale.



Slika 2.15: Uređivanje izgleda spirale

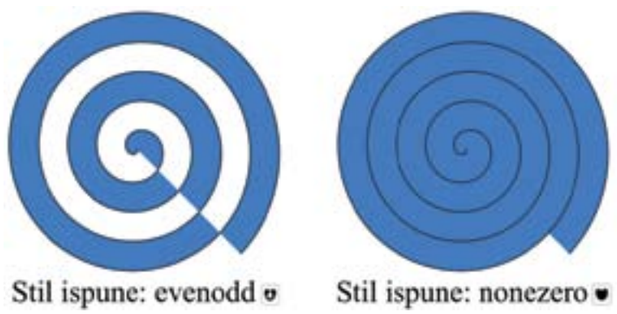
Broj zavoja spirale, odstupanje od standardne podijele i unutrašnji polumjer možete promijeniti na alatnoj traci koja se pojavljuje nakon odabira alata za rad sa spiralama.

Slika 2.16: Alatna traka za uređivanje izgleda spirale



Slika 2.17: Alatna traka za uređivanje izgleda spirale

Unutrašnjost spirala se može ispuniti bojom i za njih vrijede ista pravila kao za „zatvorene" objekte.



Slika 2.18: Ispuna spirale

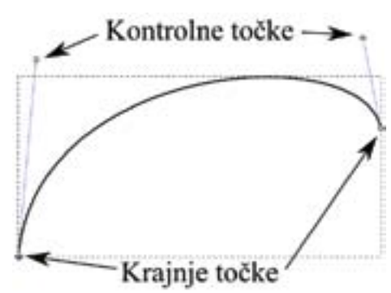
2.5 Krivulje

Krivulje su proizvoljno oblikovani objekti. Mogu biti otvorene (imati dva kraja) ili zatvorene (nemaju kraj). One mogu biti spoj, odnosno sastojati se od odvojenih više otvorenih/zatvorenih krivulja.

Slika 2.19: Vrste krivulja

2.5.1 Bezierova krivulja

Većina krivulja u programu Inkscape-u, kao i u mnogim drugim programima za crtanje, okarakterizirane su kao Bezierove krivulje. Bezierova krivulja određena je sa četiri točke, od kojih su dvije krajnje točke i dvije kontrolne točke. Krajnje točke određuju početak i završetak krivulje, dok kontrolne točke određuju smjer i intenzitet (vektor) krivulje.



Slika 2.20: Bezierova krivulja

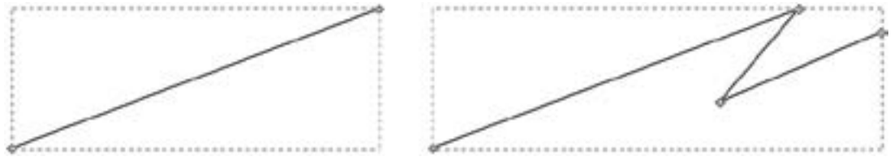
2.5.2 Izrada i uređivanje krivulje

Krivulja može biti nacrtana na tri načina:

- alatom za prostoručno crtanje ,
- za crtanje Bezierove krivulje ,
- za crtanje kistom (kaligrafija) .

Linija koja je nacrtana alatom za prostoručno crtanje ili alatom za crtanje Bezierove krivulje biti će crne boje širine jedne točkice, dok će se kod crtanja kistom koristiti zadnji korišteni oblik.

Alatom za prostoručno crtanje najlakše je nacrtati liniju. Nakon što odaberete alat, kliknite na mjesto početka linije i na mjesto kraja linije. Liniju možete nastaviti crtati tako da kliknete na krajnju točku postojeće linije i potom kliknete na točku gdje će završiti nova linija. Na taj način crtate ravnu liniju. Ako prilikom crtanja linije zadržite lijevu tipku miša iscrtat će se linija koja prati pomicanje miša, odnosno prostoručna linija.



Slika 2.21: Obična (prostoručna) krivulja

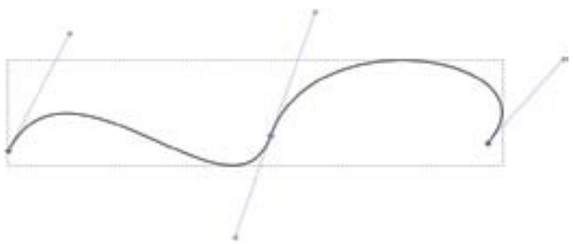
Odabirom alata za crtanje krivulje pojavljuju se različite kontrole kao što je odabir načina rada alata za crtanje , iznos glatkoće linije koju crtamo (Smoothing) ili izgled kista za crtanje (Shape).

Slika 2.22: Dodatne kontrole kod crtanja krivulje

Korištenjem alata za glatkoću (Smoothing) možete postići efekt glatke linije podešavajući njegovu vrijednost. Na slijedećoj slici su primjeri iste linije nacrtane različitom glatkoćom (gornja linija glatkoća 0, srednja 50 i donja 100).

Slika 2.23: Različite glatkoće iste linije

Kao što smo već rekli, većina linija u programu Inkscape su Bezierove linije. Pojasnimo način crtanja. Pritisnite i držite lijevu tipku miša. Time ste odredili početnu točku krivulje i pomicanjem miša određujete kontrolnu točku početka krivulje. Pustite lijevu tipku miša i pomaknite miš na završnu točku krivulje, pa opet pritisnete i držite lijevu tipku miša kako biste odredili kontrolnu točku završetka krivulje. Za završetak crtanja krivulje pritisnite tiku Enter na tipkovnici. Za nastavak crtanja kliknite mišem na završnu ili početnu točku postojeće krivulje i nastavite crtati.



Slika 2.24: Crtanje Bezierove krivulje

Prilikom crtanje kistom i kaligrafiju koristite se alatom koji vam omogućuje da jednim potezom miša stvorite objekte (ne krivulje) koji nalikuju tragu različitih sredstava za pisanje kao što su nalivpero, marker, četka, trag crva i zamrljano.

Slika 2.25: Krivulje nacrtane kaligrafskim alatima

Uređivanje krivulje izvodite koristeći se alatima s dodatne trake alatima za rad s krivuljama. Možete dodavati nove ili brisati postojeće točke, spajati dvije krivulje ili ih razdvajati, vrhove činiti oštrima, zaobljenima ili simetričnima, određivati veličinu krivulje i dr.

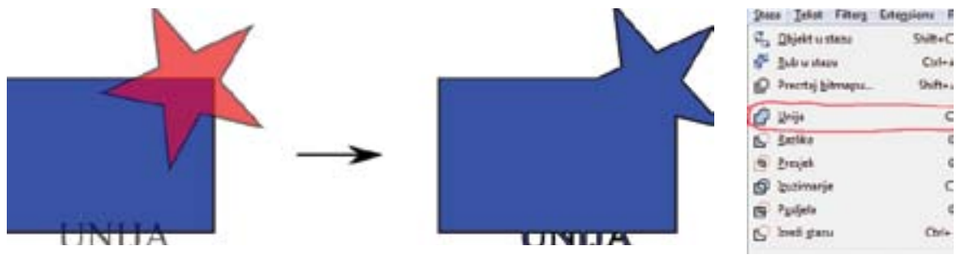
Slika 2.26: Alatna traka za rad s Bezierovom krivuljom

Promjenu položaja krajnjih i kontrolnih točaka izvodimo tako da odaberemo alat i pritiskom i držanjem lijeve tipke miša vršimo njeno premještanje na željeno mjesto.

2.5.3 Operacije s krivuljama (crop, weld, intersect ...)

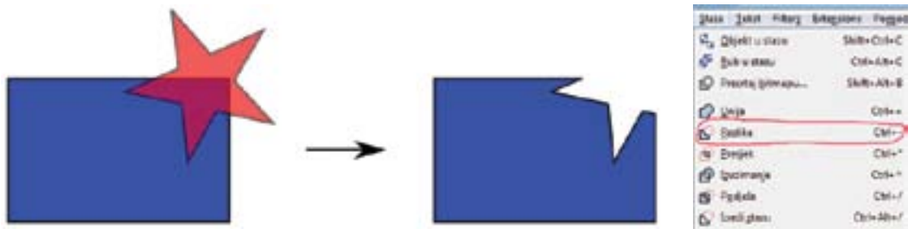
Inkcsape sadrži čitav niz naredbi za stvaranje jedne nove krivulje iz dvije ili više postojećih. Redoslijed vidljivost (Z-order) je bitan kod operacija s krivuljama jer određuje element s kojeg će se preuzeti svojstva (boja, ispuna,...) ili koji element će se spojiti/usjeći u koji. Sve naredbe su nam dostupne putem izbornika Staza. Za potrebe ovih naredbi otvorene krivulje i tekst se automatski pretvaraju u zatvorene krivulje. Prije odabira naredbe potrebno je odabrati krivulje nad kojim će se provesti naredba.

Unija jedne ili više krivulja će biti nova krivulja koja sadrži sve dijelove izvornih krivulja.



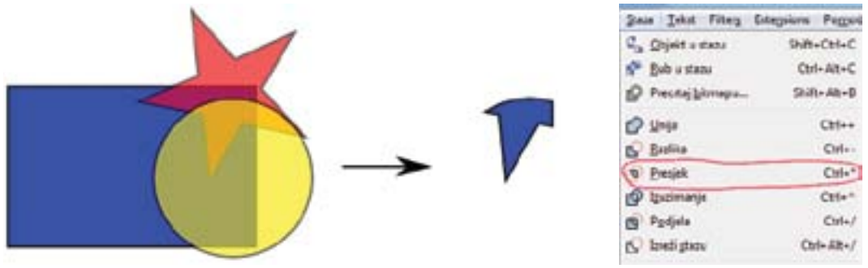
Slika 2.27: Unija dvije krivulje i teksta

Kod razlike dvije krivulje područje gornje krivulje se uklanja iz područja donje krivulje.



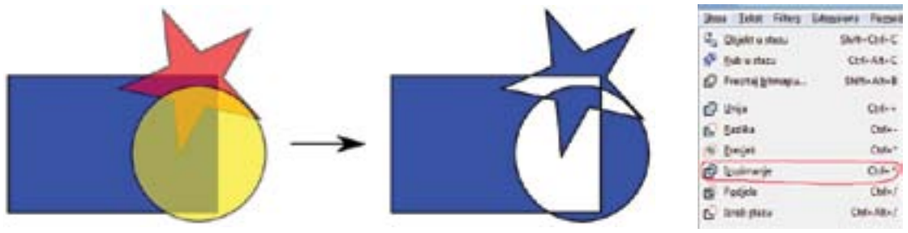
Slika 2.28: Razlika dvije krivulje

Presjek dviju ili više krivulja čini njihov zajednički prostor prije izvođenja naredbe.



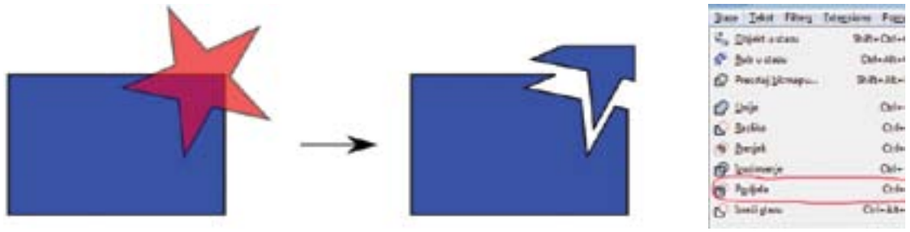
Slika 2.29: Presjek tri krivulje

Izuzimanje je naredba koja se temelji na Even-Odd ispuni (ako se područje nalazi na presjeku neparnog broja krivulja bit će dio nove krivulje, a ako je na presjeku parnog broja krivulja onda neće biti dio nove krivulje).



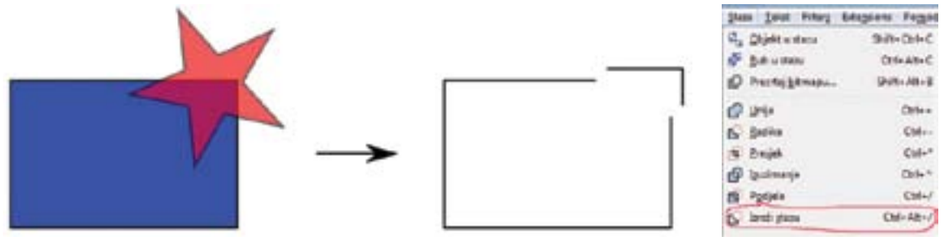
Slika 2.30: Izuzimanje tri krivulje

Podjela je naredba za rad s dvije krivulje kod kojeg gornja krivulja svojim obrubom dijeli donju krivulju na dva ili više dijelova.



Slika 2.31: Podjela dvije krivulje

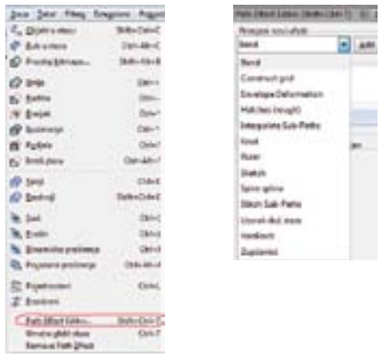
Izreži stazu naredba je koja dijeli donju krivulju u dvije ili više novih krivulja na sjecištima donje i gornje krivulje. Nove krivulje su otvorene i nemaju ispunu.



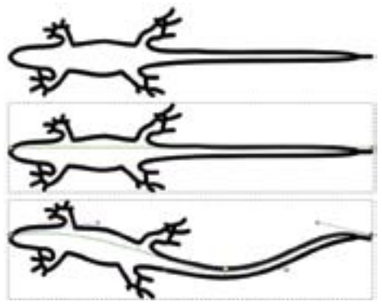
Slika 2.32: Izrezivanje dvije krivulje

2.5.4 Efekti na krivuljama (lpes) - primjeri

LPES (Live Path Effects) su efekti na krivuljama kojima možete napraviti promjenu izgleda krivulje prema nekom uzorku. Originalna krivulja se pohranjuje unutar crteža i možete ju naknadno pozvati. Za korištenje efekata na krivuljama morate prvo odabrati krivulju, potom iz izbornik Staza izabrati Path Efect editor, odabrati željeni efekt i dodati ga na odabranu krivulju koristeći tipku Add. Nakon toga možete mijenjati izgled koristeći dostupne alate koji su različiti za svaki efekt.



Efekt Bend izvodi promjenu izgleda postojeće krivulje u odnosu na središnju os. Odnosno kako mijenjate izgled središnje osi, mijenjat će se izgled i originalne krivulje.

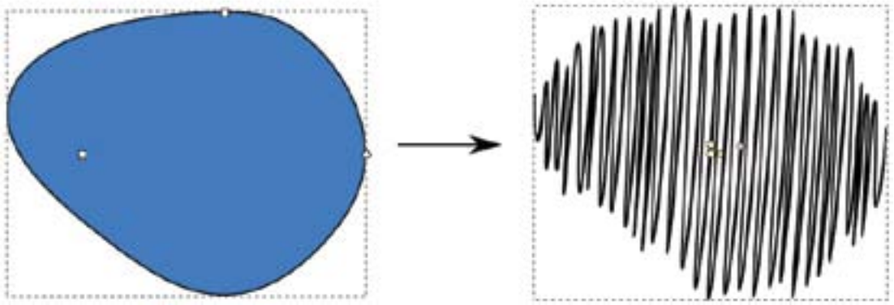


Slika 2.33: Promjena krivulje primjenom efekta Bend

Efekt Envelope Deformation omogućuje promjenu izgleda krivulje koristeći se graničnim okvirom. Svaku kontrolnu krivulju možete zasebno uređivati klikom na odgovarajuću ikonu . Na kontrolne krivulje moguće je dodavati nove čvorove i brisati ih.

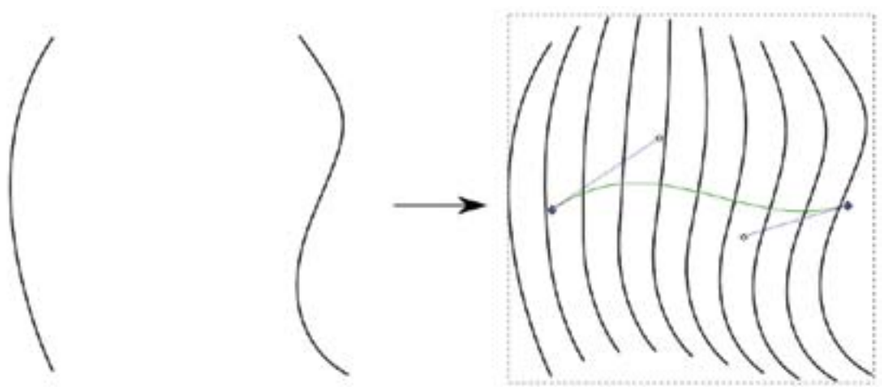
Slika 2.34: Promjena krivulje primjenom efekta Envelope Deformation

Hatches (Rough) je efekt koji ispunjava odabrani objekt krivudavom linijom koja simulira brzo, prostoručno sjenčanje olovkom.



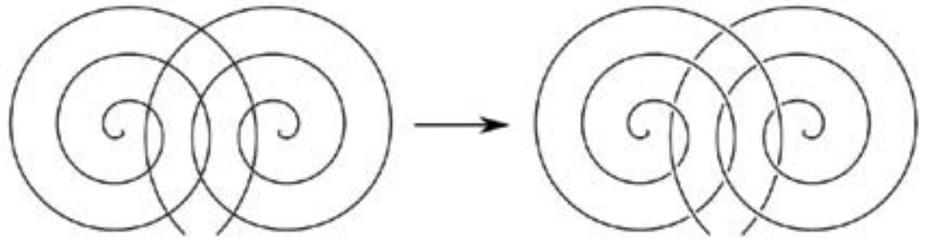
Slika 2.35: Promjena krivulje primjenom efekta Hatches (Rough)

Efekt Interpolacije stvara dodatne krivulje između dvije postojeće krivulje. Kontrolna krivulja određuje položaj novih linija koje efekt stvara.



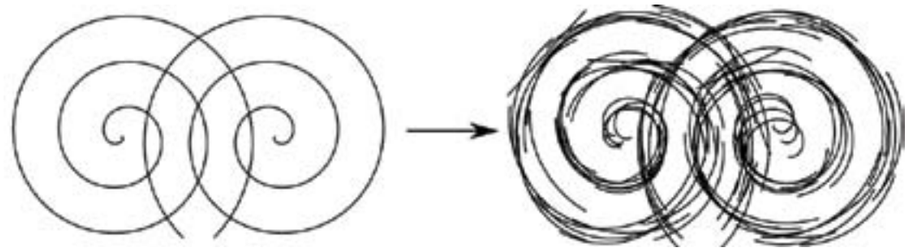
Slika 2.36: Promjena krivulje primjenom efekta Interpolacije

Efekt Knot pretvara krivulju u čvor, odnosno, na svakom sjecištu gdje krivulja siječe sama sebe, dio krivulje je sakriven kako bi se dobio efekt čvora.



Slika 2.37: Promjena krivulje primjenom efekta Knot

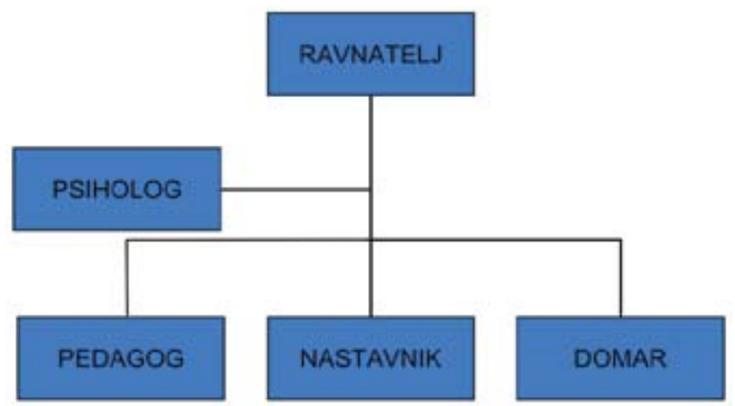
Efekt Sketch simulira crtkanje olovkom.



Slika 2.38: Promjena krivulje primjenom efekta Sketch

2.5.5 Poveznice (konektori)

Poveznice su linije koje povezuju objekte. Vrlo su korisne kod crtanja organizacijskih grafikona ili dijagrama toka. Objekti ostaju povezani poveznicama i u slučajevima kada su isti pomaknuti na drugo mjesto.



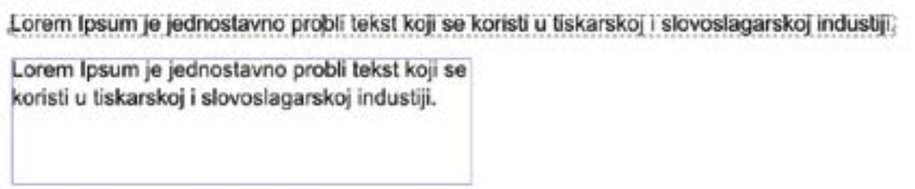
Slika 2.39: Organizacijski grafikon nacrtan koristeći poveznice

2.6 Tekst

Inkscape ima moćan sustav za pisanje i rukovanje s tekstom. Osim promjene stila teksta, vodoravnog i okomitog poravnavanja, tekst možete postaviti na krivulju ili možete ga ispisati unutar nekog nacrtanog objekta.

2.6.1 Unos, promjena, uređivanje i oblikovanje teksta

Dva su načina unosa teksta. Prvi obični tekst, kod kojeg se odabirom alata za rad s tekstom i klikom na radnu površinu programa pojavljuje treptajući pokazivač, te možete slobodno unositi tekst. Drugi način je tekući tekst kod kojeg je nakon odabira alata za rad s tekstom potrebno nacrtati oblik u koji ćete pisati tekst. Napisani tekst neće izlaziti izvan nacrtanog područja.



Slika 2.40: Primjer običnog teksta (gore) i tekućeg teksta (dolje)

Označavanje, promjena i uređivanje teksta je isto kao u programima za obradu teksta, pa ih stoga nećemo dalje objašnjavati.

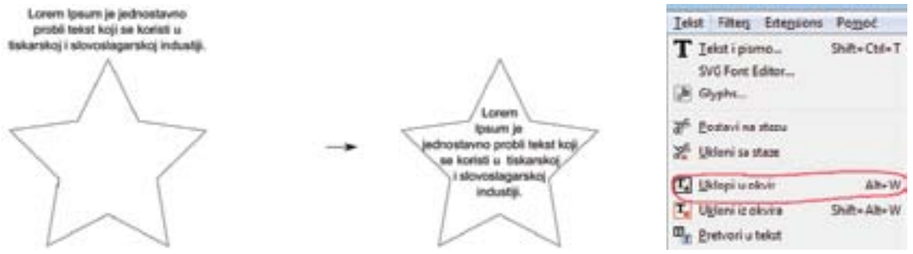
2.6.2 Tekst na krivulji i u obliku

Tekst u Inscape-u možemo postaviti na željenu krivulju ili unutar željenog oblika.

Za postavljanje teksta na krivulju potrebno je unaprijed napisati tekst i nacrtati krivulju (krivulja mora biti otvorena), potom označiti zajedno krivulju i tekst koji želimo i odabrati naredbu *Postavi na stazu* iz izbornika *Tekst*.

Slika 2.41: Postavljanje teksta na krivulju

Osim postavljanja teksta na krivulju, moguće je napisani tekst postaviti unutar postojećeg objekta. Potrebno je zajedno označiti tekst i objekt te odabrati naredbu Uklopi u okvir iz izbornika Tekst.



Slika 2.42: Postavljanje teksta u objekt

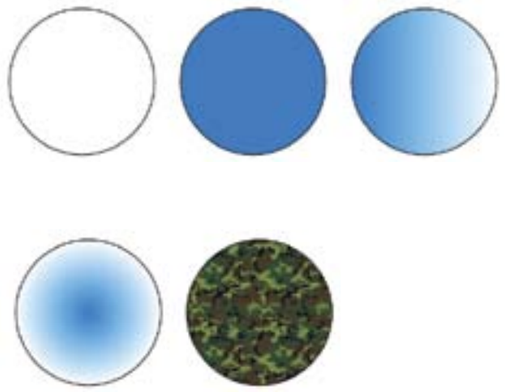
2.7 Atributi i alati

Svaki objekt ima atribute (svojstva) kao što su boja ispune i stil poteza (okvira). Ispuna se odnosi na unutrašnjost objekta, dok se potez odnosi na samu krivulju. Ispuna i potez (boja poteza) mogu biti jednolična boja, pretapanje boje, uzorak (pattern) ili ništa. Ispuna i boja poteza imaju ista svojstva, a potez još ima svojstva kao širina, vrsta crtkanja ili stil crte.

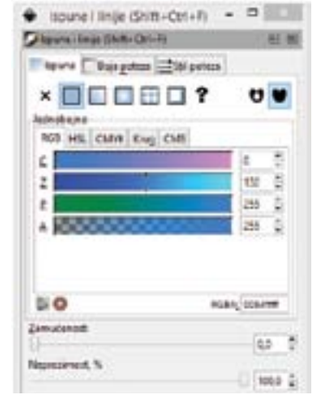
Tekstu mogu biti postavljena ista svojstva kao i bilo kojem nacrtanom objektu. Pojedinim slovima moguće je postaviti različite jednolične boje, ali pretapanje boja i uzorak je moguće postaviti samo na cijeli tekstualni objekt.

2.7.1 Ispune objekata

Kod ispune objekata postoji veliki broj različitih mogućnosti. Primjeri različitih mogućnosti prikazani su ispod. Ispunu objekta postavljate klikom na boju na paleti s bojama na dnu programa ili u prozoru za ispunu i potez. Klik desnom tipkom miša na objekt pa klik na stavku Ispune i potezi otvara prozor Ispune i linije u kojemu u prozoru Ispuna odabirete vrstu ispune, boju (koristeći različite načine miješanja boje, razinu zamućenosti i neprozirnosti odabranog objekta).



Slika 2.43: Primjeri ispune objekta



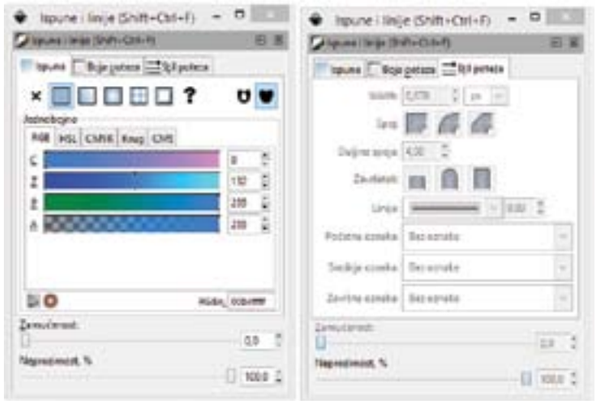
Slika 2.44: Prozor za odabir ispune objekta

2.7.2 Konture objekata

Boju poteza objekta (konturu, obrub) odabiremo kao i boju ispune, ali u prozoru Boja poteza.

Slika 2.45: Primjeri poteza objekta

Ostale atribute poteza odabirete u prozoru *Stil poteza*. Ovdje možete odabrati širinu, način završetka i spoja linije, vrstu crtkanja, oblik strelice te naravno, zamućenosti i neprozirnost.



Slika 2.46: Primjeri poteza objekta

2.7.3 Alat tweak, spray, eraset i paint bucket

Alat Tweak koristi se za male izmjene na objektima, krivuljama i boji. Alat Tweak radi kao kist koji pokriva kružni dio crteža i označen je narančastim krugom.

Slika 2.47: Izgled alata Tweak

Dodatni izbor veličine, izgleda i funkcije alata vršimo na dodatno traci s alatima koja se pojavljuje nakon što odaberemo alat.

Slika 2.48: Dodatne mogućnosti alata Tweak

2.8 Efekti i filteri

Filteri omogućuju korisniku promjenu izgleda objekta prema unaprijed određenom načinu. Objekti se korištenjem filtera zamagljuju, zamućuju, miješaju boje i slično. Inkscape podržava gotovo sve standardne filtere, ali podržava i neke korisničke (napredne filtere).

2.8.1 Standardni filteri

Inkscape sadržava nekoliko stotina standardnih filtera. Ti filteri se mogu pronaći u izborniku *Filters*.

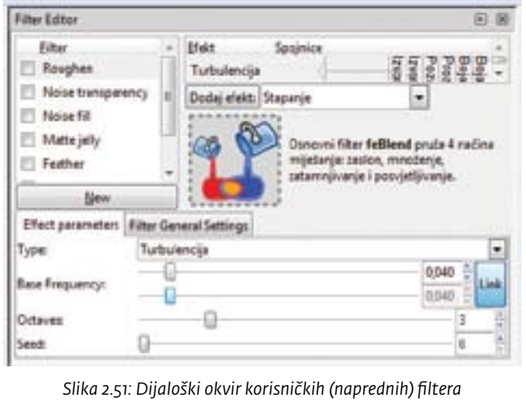
Za uporabu pojedinog filtera potrebno je prvo označiti objekt ili grupu objekata i potom odabrati željeni filter.

Slika 2.50: Primjena standardnog efekt ABC. Slijeva originalni crtež, Black Outline,Diffuse light, Feather, Matte jelly, Noise fill, Nise transparency i Roughen

Budući da Inkscape podržava nekoliko stotina standardnih filtera u ovom priručniku nećemo pojasniti uporabu svakog pojedinačno zbog ograničenja prostora.

2.8.2 Korisnički (napredni filteri) filteri

Korisnički (napredni) filteri su filteri čije sastavnice određuje sam korisnik prema svojim potrebama. Pristup dijaloškom okviru za definiranje korisničkog filtera nalazi se u izborniku *Filters-Filter Editor...* nakon čega se otvara dijaloški okvir prikazan na slici ispod.



Slika 2.51: Dijaloški okvir korisničkih (naprednih) filtera

Odabirom i dodavanjem novih elemenata korisničkog filtera stvarate filter prema svojim potrebama koji poslije možete jednostavno i brzo upotrebljavati u daljnjem radu.

2.9 Skiciranje (trace) slike

Inkscape može skiciranjem pretvoriti bitmap sliku u krivulju. Skiciranje slike nije jednostavan postupak i najbolji rezultati se dobiju na crno-bijelim slikama i slikama s jasno izraženom granicom između boja, a slike kod kojih je jako izraženo pretapanje boja skiciranje možda neće dati zadovoljavajući rezultat. Alat pokrećemo preko izbornika *Staza – Precrtaj bitmapu*. Dobivena krivulja često zna imati tisuće točaka i za rad s njom je potrebno pojednostaviti dobivenu krivulju (izbornik *Staza – Pojednostavi*).

2.9.1 Osnovno skiciranje slike

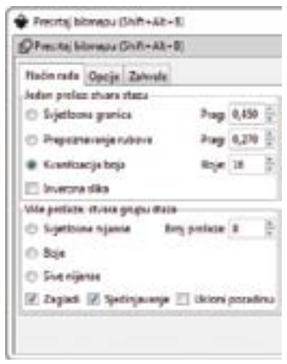
Osnovno skiciranje slike stvara jednu liniju iz odabrane slike. Jednostavno skiciranje ćete napraviti pomoću jednostavne slike ruže.

Svjetlosna granica je dobar izbor na crno bijelim slikama, ali na slikama u boji postoje određeni problemi kod skiciranja tako da kod ove slike ruže i uz veliki trud u podešavanju iznosa svjetlosne granice nije moguće dobiti odgovarajući rezultat. Upotrebom mogućnosti *Prepoznavanje rubova* s vrijednošću 0,27 dobiven je slijedeći crtež.

Slika 2.52: Slika ruže

Slika 2.53: Slika ruže precrtana korištenjem mogućnosti Precrtavanje rubova

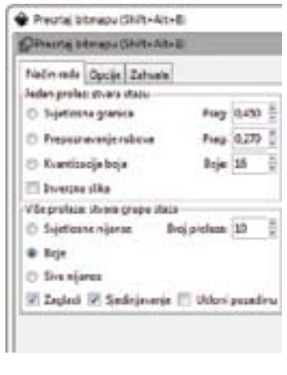
Kod mogućnosti Kvantizacija boja povećavanjem broja boja dobiva se veća razina detalja. Sami morate odabrati broj boja koji vam daje najbolji rezultat.



Slika 2.54: Slika ruže precrtana korištenjem mogućnosti Kvantizacija boja (lijevo sa 4 boje, desno sa 16 boja)

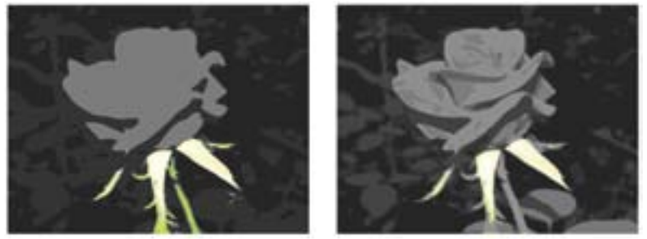
2.9.2 Napredno skiciranje slike

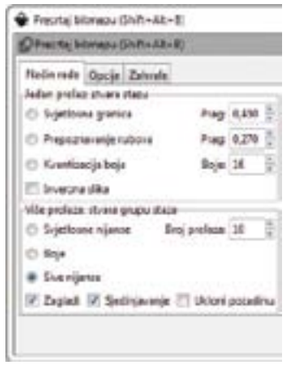
Kod naprednog skiciranja slike slika se skicira nekoliko puta te se kod svakog skiciranja koriste različite postavke. Kod svakog skiciranja nastaje jedna krivulja, a krivulje od svih skiciranja su grupirane po završetku skiciranja. Kod naprednog skiciranja u polju *Broj prolaza* određujete koliko želite puta ponovo skicirati sliku, a kod svakog skiciranja ćete dobiti jednu boju. Ostale zajedničke opcije su: zagladi, sjedinjavanja i ukloni pozadinu.
Koristeći mogućnost *Boja* rezultat skiciranja će biti crtež koji se sastoji od višebojnih krivulja.



Slika 2.55: Slika ruže precrtana korištenjem mogućnosti Boja (lijevo sa 4 prolaza, desno sa 10 prolaza)

Koristeći mogućnost *Sive nijanse* rezultat skiciranja će biti crtež koji se sastoji od više sivih krivulja.





Slika 2.56: Slika ruže precrtana korištenjem mogućnosti Sive nijanse (lijevo sa 4 prolaza, desno sa 10 prolaza)

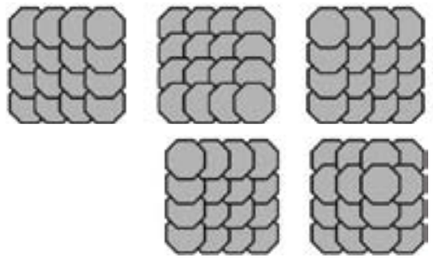
2.10 Dodatne mogućnosti programa

Inkscape može biti nadograđen pomoću dodataka, odnosno skripti i programa koji mogu biti pokrenuti iz njega. Dodaci mogu biti i dobar način nadogradnje programa Inkscape za napredne korisnike kako biste si olakšali radu u njemu jer dodaci, osim što mogu biti pokrenuti ručno od strane korisnika, mogu biti pokrenuti i automatski u pozadini kada se ispune određeni uvjeti. Ove dodatne mogućnosti možemo pronaći u izborniku *Extensions*.

2.10.1 Raspoređivanje objekata

Ova kategorija preslaguje objekte u crtežu mijenjajući njihovu vrijednost na z-osi, odnosno mijenja redoslijed prikazivanja objekata. Pristupamo joj preko izbornika *Extensions – Rasporedi*. U ovoj kategoriji imamo samo jedan alat Restack.

Slika 2.57: Dijaloški okvir alata Restack



Slika 2.58: Prikaz Mogućnosti alata Restack

Na slici iznad prikazane su mogućnosti alata Restack. S lijeva na desno prikazane su: originalni crtež, Top to Bottom, Right to Left, Arbitrary Angle (135°) i Radial Inward.

2.10.2 Rad s bojama (rgb, grayscale...)

Ova grupa dodataka manipulira bojama objekta ili grupe objekata. Ako nije označen niti jedan objekt, onda se promjena primjenjuje na sve objekte u trenutnom crtežu. Pristupamo joj preko izbornika *Extensions – Boja*. Black and White pretvara crteže u crno bijelu boju.

Slika 2.59: Lijevo: originalne boje, desno: nakon primjene dodatka Black and White

Jače osvjetljeno je dodatak koji tamnije boje pretvara u svjetlije, ali je potreban oprez jer svjetlije boje mogu nestati. Dodatak radi malu promjenu pa je ponekad potrebna višestruka primjena.

Slika 2.60: Lijevo: originalne boje, desno: nakon dvostruke primjene dodatka Jače osvjetljeno

Manje zasićeno je dodatak koji smanjuje zasićenje boje u crtežu u koraku od 5%.

Slika 2.61: Lijevo: originalne boje, desno: nakon dvostruke primjene dodatka Manje zasićeno

Nasumično je dodatak koji mijenja boje odabranih objekata. Mijenjaju se karakteristike iz područja nijanse, zasićenosti i osvijetljenosti boje.

Slika 2.62: Lijevo: originalne boje, desno: nakon primjene dodatka Nasumično

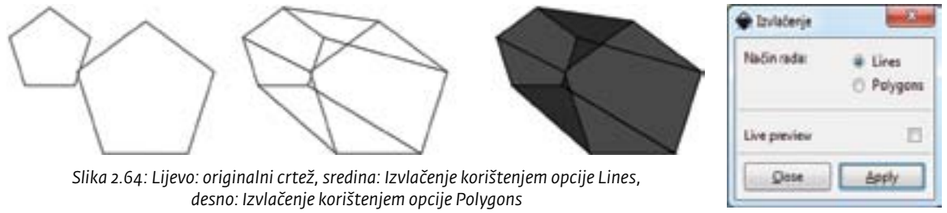
Inkscape sadrži još mnogo dodataka iz skupine Boja koje možete upotrijebiti prema potrebi.

2.10.3 Interpolacija i ekstrudiranje objekata, dodavanje efekata na objekte

U ovoj grupi dodataka nalaze se dodaci koji stvaraju nove objekte iz jednog ili više postojećih. Pristupamo im preko izbornika *Extensions – Generiraj iz staze*. *Interpoliraj* je dodatak koji stvara željeni broj novih objekata između dva postojeća objekta postupno mijenjajući oblik i boju jednog objekta prema drugom objektu.

Slika 2.63: Lijevo: originalni crtež, desno: nakon primjene dodatka Interpoliraj s 5 koraka

Izvlačenje (Extrude) je dodatak koji spaja krajnje točke iz dvije krivulje ili dva poligona. Ako jedan od objekata ima više krajnjih točaka, višak točaka na jednom poligonu se ne upotrebljava. Kod ovoga dodatka možemo birati hoćemo li koristiti opciju Lines ili Polygons.



Slika 2.64: Lijevo: originalni crtež, sredina: Izvlačenje korištenjem opcije Lines, desno: Izvlačenje korištenjem opcije Polygons

Inset/Outset Halo... je dodatak koji proizvodi zamućene slike originalnog objekta. Dodatak stvara više kopija polaznog objekta koje su povećane/smanjene i na kojima je povećana prozirnost objekta.

Slika 2.65: Lijevo: originalni crtež, desno: nakon primjene dodatka Inset/Outset Halo...

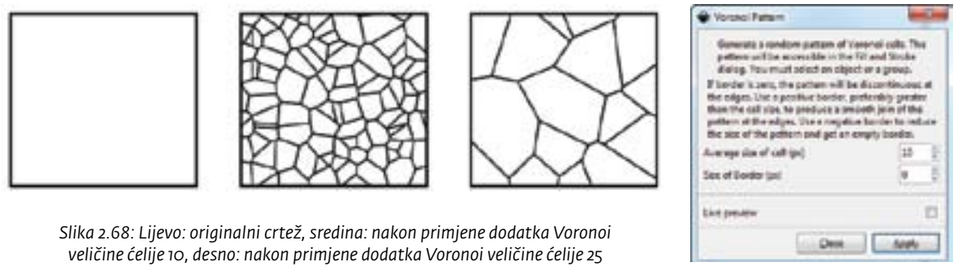
Motion je dodatak koji simulira pokret objekta, odnosno iscrtava kopije istog objekta iza originala i spaja odgovarajuće točke na objektu u liniju. Smjer i pomak objekta mogu biti uređivani, a novonastali objekti nasljeđuju svojstva originala.

Slika 2.66: Lijevo: originalni crtež, desno: nakon primjene dodatka Motion

Scatter je dodatak koji iscrtava (kopira) odabrani objekt duž određene krivulje. U dodatku se mogu postaviti razmaci između kopiranih objekata, njihovo zakretanje, praćenje krivulje, kloniranje, kopiranje ili pomicanje objekta i još dosta različitih mogućnosti.

Slika 2.67: Lijevo: originalni crtež, desno: nakon primjene dodatka Scatter

Voronoi je dodatak koji popunjava objekt ili grupu objekata Voronoievim uzorkom koji se temelji na distribuciji početnih i završnih točaka linije metodom slučajnog odabira. U dodatku možete postaviti prosječnu veličinu ćelije u točkicama i debljinu linije.

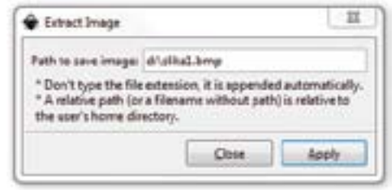


Slika 2.68: Lijevo: originalni crtež, sredina: nakon primjene dodatka Voronoi veličine ćelije 10, desno: nakon primjene dodatka Voronoi veličine ćelije 25

2.10.4 Uvoz i spremanje slika u različitim formatima

Slike u program Inkscape možete ugrađivati koristeći izbornik *Extension-Slike-Embed Images*. Na taj način slika se ugrađuje unutar dokumenta i više je nije potrebno imati izvan crteža te je moguće kopiranje i premještanje crteža bez kopiranja vanjske datoteke iste slike.

Program Inkscape također ima mogućnost izdvajanja slike koja je ugrađena u crtež. Sliku prethodno označite i nakon toga koristeći izbornik *Extensions-Slike-Extract Images* izvršite izdvajanje slike u .bmp format u željenu mapu i pod željenim nazivom.



Slika 2.69: Dijaloški okvir za izdvajanje slike iz crteža

2.10.5 Rasterizacija vektorskih objekata

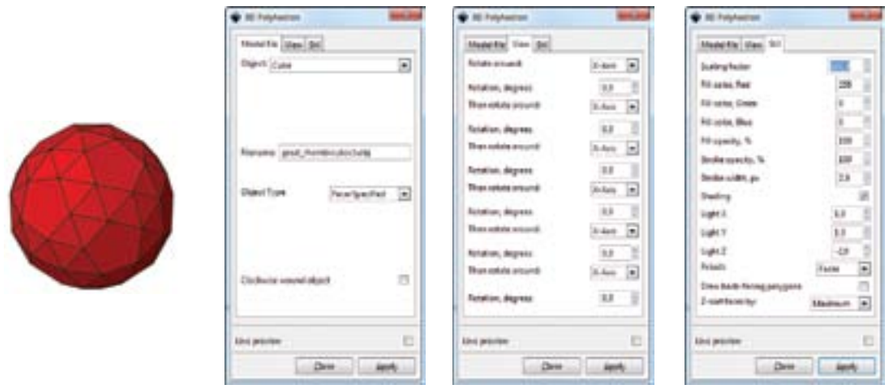
Ova skupina dodataka služi za manje korekcije i izmjene ugrađene slike unutar crteža, a pristupa im se iz izbornika Extensions-Raster. Nije namijenjena za veće zahvate na slikama, ali može pomoći da se bez upotrebe vanjskog programa za obradu slike naprave neke jednostavnije obrade slike.

Slika 2.70: Lijevo: originalni slika, sredina: nakon primjene dodatka Gaussovo zamućivanje, desno: nakon primjene dodatka Wave

U skupini Raster se nalaze slijedeći dodaci: Adaptive Threshold, Add Noise, Blur, Channel, Charcoal, Colorize, Contrast, Cycle Colormap, Despeckle, Dither, Edge, Emboss, Enhance, Equalize, Gaussian Blur, HSB Adjust, Implode, Level, Level (with Channel), Median, Modulate, Negate, Normalize, Oil Paint, Opacity, Raise, Reduce Noise, Resample, Shade, Sharpen, Solarize, Spread, Swirl, Unsharp Mask i Wave.

2.10.6 Renderiranje

Ova skupina dodataka stvara nove objekte. Pristupamo joj preko izbornika Extensions – Iscrtaj. 3D Polyhedrons je dodatak koji stvara 3D polihedrone.



Slika 2.71: 3D polihedron stvoren programom Inkscape

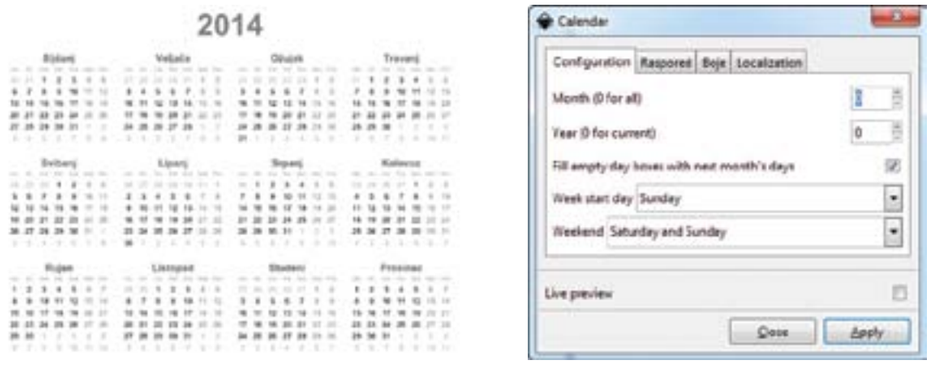
Alphabet Soup je dodatak koji stvara tekst neobičnog izgleda promjenom izgleda dijelova unesenog teksta.

Slika 2.72: Tekst „Inkscape" izmjenjen dodatkom Alphabet Soup

Barkod je dodatak kojim se stvaraju barkodovi. Mogu se stvarati različite vrste barkodova, bilo linijski ili datamatrix barkodovi.

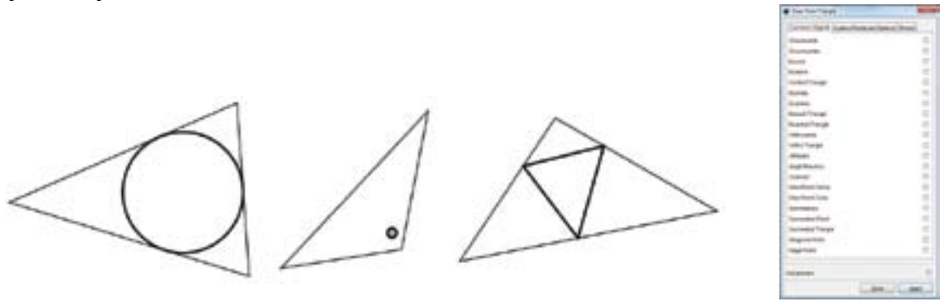
Slika 2.73: Lijevo: linijski barkod, desno dotmatrix barkod

Calendar je dodatak za stvaranje različitih vrsta kalendara: za cijelu godinu ili za jedan mjesec, s početkom tjedna u nedjelju ili ponedjeljak, koji su dani vikend, a koji nisu...



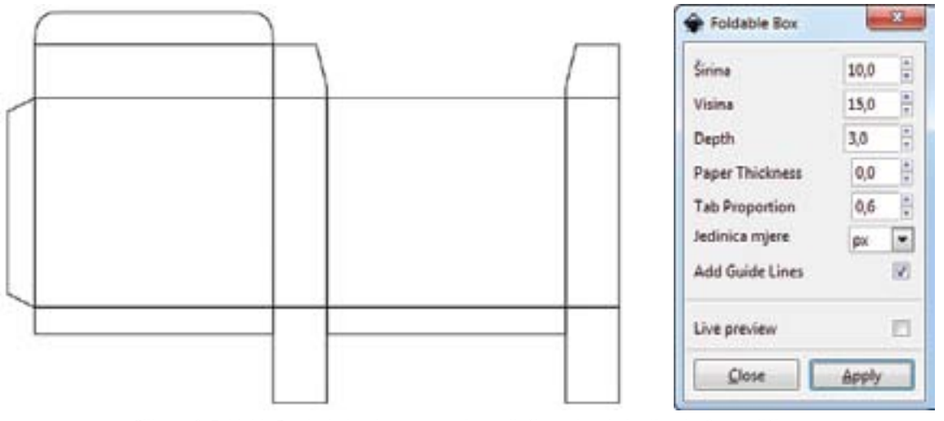
Slika 2.74: Kalendar za 2014. godinu stvoren pomoću dodatka Calendar

Draw From Triangle je omiljeni dodatak među ljubiteljima geometrije. On omogućava stvaranje brojnih konstrukcija temeljenih na trokutu. Trokut može biti nacrtan koristeći alate za crtanje linija, ali linija mora biti zatvorena.



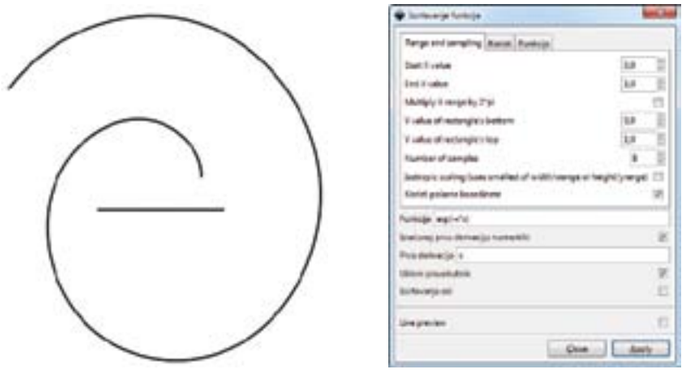
Slika 2.75: Konstrukcije temeljene na trokutu izrađene dodatkom Draw From Triangle

Foldable Box je dodatak za stvaranje predloška za izradu kutije određene veličine.



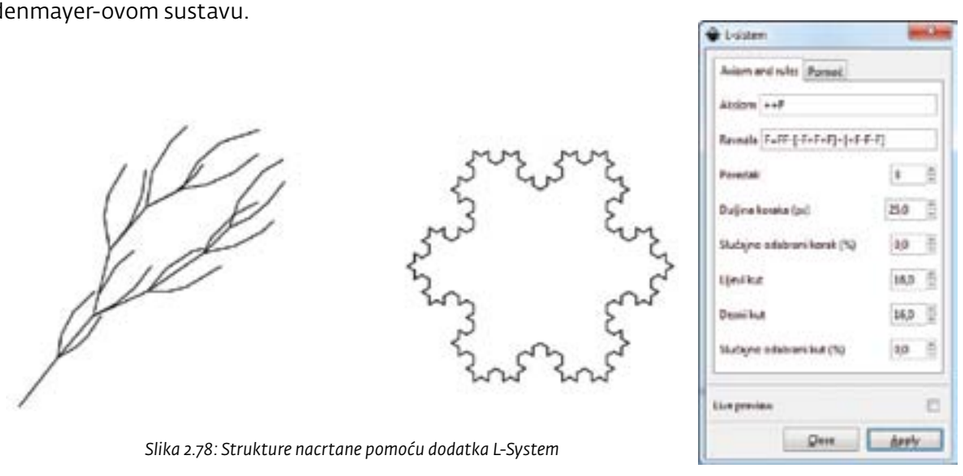
Slika 2.76: Foldable Box je dodatak za stvaranje predloška za izradu kutije određene veličine.

Function Plotter je dodatak pomoću kojega možemo crtati grafove matematičkih funkcija.



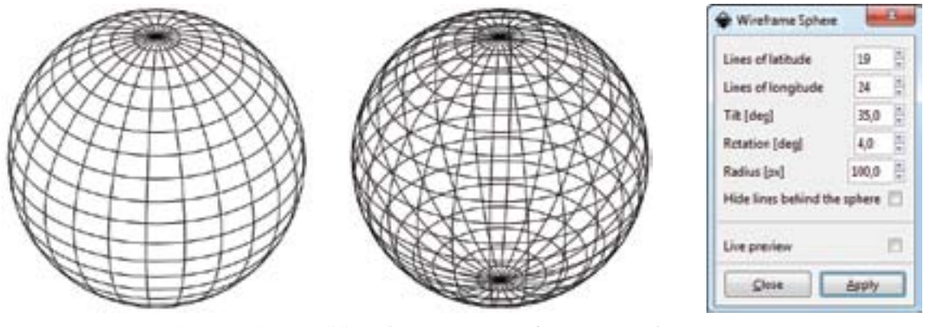
Slika 2.77: Graf funkcije y=sqrt(x) izrađen dodatkom Function Plotter

L-System (Fractal-Lindenmayer) je dodatak pomoću kojeg možemo crtati strukture prema Lindenmayer-ovom sustavu.



Slika 2.78: Strukture nacrtane pomoću dodatka L-System

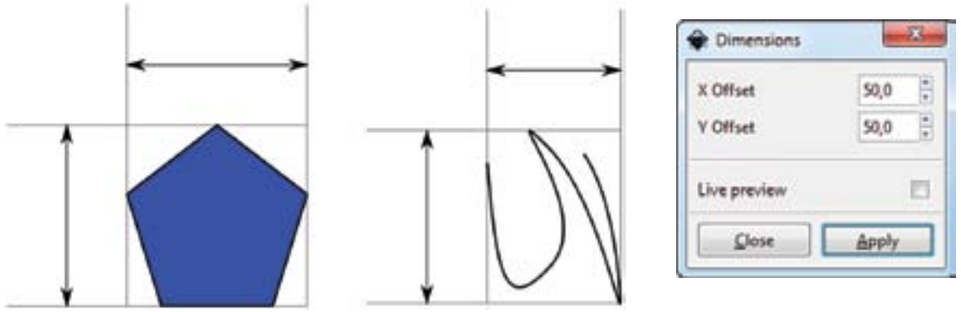
Wireframe Sphere je dodatak koji nam omogućuje crtanje sfere sačinjene od žičane mreže koristeći elipse koje predstavljaju geografsku dužinu i širinu.



Slika 2.79: Strukture žičane sfere nacrtane pomoću dodatka Wireframe Sphere

2.10.7 Vizualizacija krivulje

Dodatak Visualize Path vam omogućuje ispis informacija o krivulji na samoj krivulji. Dimensions je dodatak koji omogućava ispis veličine objekta ili grupe objekata. Dodatak se koristi hvataljkama na objektima za određivanje pozicija strelica.



Slika 2.80: Dimenzioniranje objekata koristeći dodatak Dimensions

Draw Handles iscrtava kontrolne linije koje se vide prilikom uređivanja krivulje.

Slika 2.81: Kontrolne linije nacrtane dodatkom Draw Handles

Measure Path je dodatak koji mjeri dužinu krivulje i ispisuje istu uz krivulju.

Slika 2.82:Dužina krivulje ispisana koristeći dodatkom Measure Path

Number Nodes je zanimljiv dodatak koji izdvaja početne (krajnje) točke krivulje, označava ih brojevima i stvara starinski „spoji točke“ zadatak.

Slika 2.83: Krivulja pretvorena u „spoji točke" zadatak pomoću dodatka Number Nodes

2.11 Svg na internetu

SVG tek na internetu pokazuje sve svoje prednosti. Male i kompaktne strukture datoteke brzo se preuzimaju s interneta na korisničko računalo, a njegova vektorska struktura rezultira u odličnoj kvaliteti iscrtavanja pri bilo kojem povećanju. SVG datoteke mogu biti ugrađene u internet stranice, mogu sadržavati veze prema drugim internet stranicama, mogu sadržavati skripte i mogu biti animirane.

2.11.1 Jednostavan svg prikaz

Mnogo je različitih načina za prikaz SVG datoteke na internetu, ali je najjednostavniji način povezivanje pomoću oznake(tag) <a>. Internetski preglednici koji podržavaju SVG prikazati će ih sami, dok za ostale će biti potrebno ugraditi SVG sadržaj jednom od slijedećih mogućnosti: <object>, <embed>, <iframe>, <img>, Inline SVG i CSS pozadina.

Ugradnju SVG datoteke prikazat ćemo na primjeru oznake <object>.

Slika 2.84: Jednostavan svg objekt ugrađen pomoću oznake <objekt>

2.11.2 Položaj svg-a

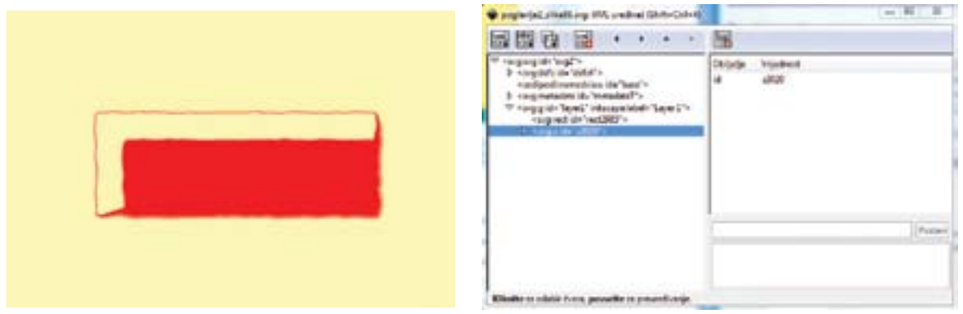
Budući da sada znate kako prikazati SVG crtež u html datoteci, pokazat ćemo kako ga postaviti na željeno mjesto. Taj postupak je malo zamršen i rezultati ne moraju biti isti u različitim internetskim preglednicima. Postupak se sastoji od dva koraka: prvi je odrediti područje na internet stranici u kojemu će se prikazati SVG i drugi korak je odrediti kako će SVG ispuniti taj rezervirani prostor.

Slika 2.85: Jednostavan svg objekt ugrađen pomoću oznake <objekt> s određenom veličinom i položajem

2.11.3 Dodavanje veze

Veze na internetskim stranicama povezuju resurse. U tradicionalnom, HTML stilu, sredstva su internetske stranice, slike i sl. U SVG stilu primjenjuje se moćniji i složeniji XLinks. U ovom priručniku bit će prikazana najjednostavniju upotreba XLinks povezivanja.

Za dodavanje veze na neki objekt potrebno je kliknuti na njega desnom tipkom miša i odaberite stavku Stvori vezu iz dobivenog izbornika. Iako se ne na prvi pogled čini da se ništa nije dogodilo, ovaj potez je stvorio <svg:a> omot oko objekta ili grupe objekata na koji ste kliknuli desnom tipkom miša. Klikom na izbornik Uredi –XML uređivač vidljivo je da je Inkscape pripremio objekt za postavljanje veze.



Slika 2.86: Prikaz objekta (lijevo) i prikaz XML Uređivača (desno)

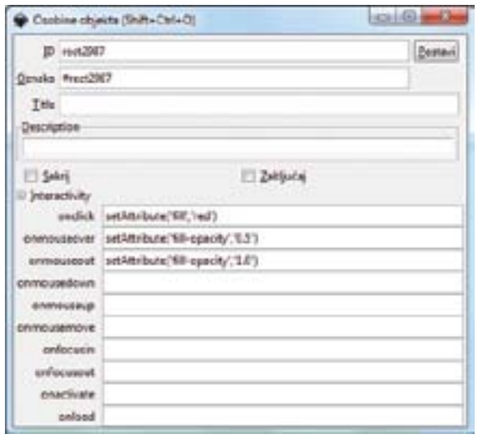
Ponovo kliknite desnom tipkom miša na objekt i odaberite stavku Svojstva veze. Otvara se prozor u kojem ih možemo urediti.

Slika 2.87: Dijaloški okvir Svojstva veze

2.11.4 Dodavanje javascript-a

SVG crteži mogu koristiti JavaScript (ECMAScript) kako bi omogućili složeno rukovanje objektom. U primjeru koji slijedi pokazat ćemo jednostavnu primjenu JavaScript-a na SVG objektu. Prvo je potrebno ukloniti postavke stila na željenom objektu koristeći XML uređivač.

Potom kliknite na objekt desnom tipkom miša i odaberite stavku *Svojstva objekta*, kliknite na dnu prozora na stavku *Interactivity* i popunite polja *onmouseover*, *onmouseout* i *onclick*.



Slika 2.88: Dijaloški okvir Osobine (svojstva) objekta

Ako u internet pregledniku potom otvorite ovaj crtež dobit ćete slijedeće izglede objekta.

Slika 2.89: Lijev: izgled objekta pri otvaranju, sredina: izgled objekta kada je miš iznad njega, desno: izgled objekta kada je pritisnuta lijeva tipka miša

2.11.5 Jednostavne animacije

SVG standard osigurava podršku za animiranje dijelova crteža. Animacije je moguće izvesti interno kroz animacije elemenata i izvana kroz skripte.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<svg
   xmlns="http://www.w3.org/2000/svg"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   version="1.1"
   onload="Start(evt)"
   width="150"
   height="150">
  <script type="text/ecmascript">
   <![CDATA[

    var time = 0;
    var delta_time = 10;
    var max_time = 1000;
    var dir = 1;

    var the_rect;

    function Start(evt) {

      the_rect = evt.target.ownerDocument.getElementById("kvadrat");
      Oscillate();
    }

    function Oscillate() {

      time = time + dir * delta_time;
      if (time >  max_time)  dir = -1;
      if (time < -max_time)  dir =  1;

      // Calculate x position
      x_pos = (time * 25) / max_time;
          y_pos = (time * 25) / max_time;
      the_rect.setAttribute("transform", "translate(" +x_pos+ "," +y_pos+ ")");

      // Repeat
      setTimeout("Oscillate()", delta_time)
    }

    window.Oscillate = Oscillate

   ]]>
  </script>
  <a xlink:href="http://www.w3.org/"
    style="fill-opacity:0.75">
    <rect
     id="kvadrat"
     width="90"
     height="90"
     x="30"
     y="30"
     style="fill:#0000ff;stroke:#000000"
     onmouseover="setAttribute('fill-opacity','1.0')"
     onmouseout="setAttribute('fill-opacity','0.75')">
      <desc
       id="desc3364">A clickable square to test simple JavaScript.</desc>
      <title
       id="title3362">Click square to go to http://www.w3.org/</title>
    </rect>
  </a>
</svg>
```

Slika 2.89: Java skripta koja izvodi animaciju kvadrata

Slika 2.90: Prikaz animacije kvadarata

3. poglavlje

ROBOTIKA

7. razred

3.0 Robotika

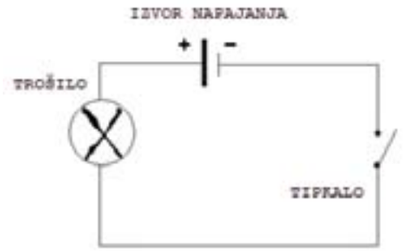
3.1 Uvod

U robotici se sve svodi na upravljanje jednostavnim električnim uređajima, žaruljicama, elektromotorima... Pomoću računala mogu se izvesti sljedeće radnje: paljenje i gašenje žaruljica, pokretanje i zaustavljanje elektromotora te ostale radnje koje nam omogućuju upravljanje uređajima. Školski robotski sustavi služe za učenje o obavljanju funkcija robota, vježbanje u rukovanju i programiranju robota. Školski model robota i industrijski robot ne razlikuju se u bitnim funkcijama. Najčešći oblik robotskih sustava su robotska kolica. Robotskim kolicima mogu se obavljati zanimljivi pokusi i istraživanja te se mogu programirati tako da sama izbjegavaju zapreke koje im stoje na putu. U labirintu primjerice mogu imati zadaću sami pronaći izlaz, pomoću senzora prate crtu na podu te mogu spašavati zamišljene „žrtve" itd. Robote pokreću različiti motori koji su i osnova svakog rada robota. Gibanje se prenosi s motora na pokretne dijelove robota različitim prijenosnicima. Prijenosnici služe za prijenos gibanja te za promjenu brzine gibanja i zakretnog momenta. Kako bismo mogli komunicirati s našim robotima, moramo na računalo instalirati programski jezik u kojemu izrađujemo programe za ostvarivanje komunikacije. Postoji mnogo programa koji su razvijeni za pokretanje i kontrolu robota. Takvi programi za kontrolu i upravljanje robotima mogu se sastaviti i pomoću programskih jezika opće namjene kao što je npr. Qbasic. Jedan od programa za programiranje robota je i RoboPro, a ikone pomoću kojih slažemo program povezujemo jednu za drugom u oblik dijagrama tijeka. Na takav način dobivamo složeni program kojeg povezujemo sa sučeljem preko kojeg dajemo naredbe robotu. Kako bismo pomoću računala mogli upravljati robotom, potreban je poseban sklop tzv. robotičko USB sučelje ili interface, kojim se povezuju računalo i robot. Robotičko sučelje na računalo spajamo pomoću USB kabela i služi za prebacivanje programa s računala, tj. zapisivanje programa direktno u „mozak" robota. Osim RoboPro programa, na računalo moramo instalirati i upravljačke programe za sučelje koje koristimo. Spojimo sučelje na računalo i na napajanje (baterija 9V) te s instalacijskog CD-a *RoboPro Software* instaliramo potrebne upravljačke programe za robotičko sučelje.

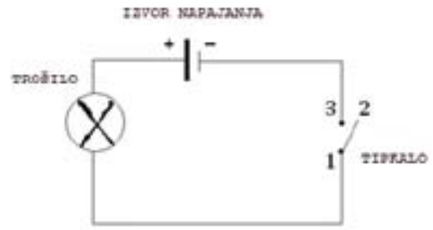
Slika 3.1: Programski paket „RoboPro"

3.2 Strujni krug

Upravljanje radom električnih i elektroničkih uređaja temelji se na uspostavljanju i prekidanju strujnih krugova. Jednostavan strujni krug sadrži izvor napajanja (bateriju), trošilo (žaruljicu) i električne vodove (spojne žice, vodiče).



Slika 3.2: Shema s tipkalom



Slika 3.3: Shema s Fischertechnik tipkalom

Povežemo li dvjema spojnim žicama polove na bateriji sa žaruljicom dobit ćemo jednostavni strujni krug. Struja prolazi od izvora napajanja preko vodiča koji služe kao prijenosnici energije do trošila (žaruljica). Prikazanim strujnim krugom na shemi 2 i 3 upravlja se pritiskom na tipkalo koje služi za paljenje i gašenje žaruljice. Ovisno o položaju tipkala otvaramo i zatvaramo strujni krug bez potrebe fizičkog iskopčavanja/ukopčavanja spojnih žica (vodiča). Žaruljica svijetli ukoliko je strujni krug zatvoren. Slike 2 i 3 prikazuju otvoreni strujni krug jer tipkala u ovim slučajevima kao da prekidaju vodič te ovim strujnim krugovima ne teče električna struja. Pritisnemo li tipkalo zatvaramo strujni krug, žaruljica počinje svijetliti jer je zatvorenim strujnim krugom potekla je struja. Žaruljica iz baterije dobiva električnu energiju koju pretvara u svjetlost. Budući da žaruljica troši električnu energiju pretvarajući je u toplinsku i u svjetlosnu energiju nazivamo je trošilo. Što će se dogoditi ako na jednom mjestu odvojimo spojnu žicu od baterije i žaruljice? Žaruljica će se ugasiti. Strujni krug bit će otvoren pa kroz njega neće prolaziti struja.

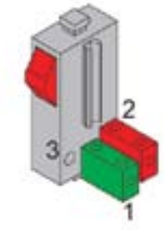
Osnovni elementi strujnog kruga su:

1. Izvor napajanja (baterija)
2. Vodiči (spojne žice) - materijal koji dobro provodi struju, preko njega se struja prenosi strujnim krugom
3. Tipkalo (prekidač) - sklopka za uključenje/isključenje strujnog toka
4. Trošilo (žaruljica).

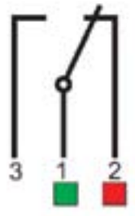
3.3 Izmjenično tipkalo

Na jednoj strani Fischertechnik tipkala prikazana je shema kao na slici 4. Sa sheme možemo vidjeti da su u početnom položaju (u kojem tipkalo nije pritisnuto), unutar tipkala spojeni kontakti 1 i 2. Pritisnemo li tipkalo, sklopka koja je spojena na kontakt 1 prebaci se iz položaja 2 u položaj 3. Iz priloženog možemo primijetiti da tipkalo ima dva stanja preko kojih se u određenom trenutku zatvara strujni krug. Zbog takvog načina rada u kojem se stanja mogu izmjenjivati tipkalo nazivamo izmjenično. Način spajanja tipkala ovisi o našim potrebama i željama.

Slika 3.4:Shema spoja kontakata unutar tipkala

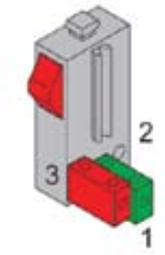


Slika 3.5: Tipkalo s priključnicama (1,2 i 3)

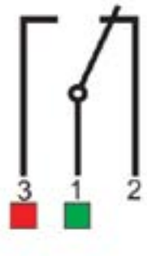


Slika 3.6: Shema spoja

Spojimo li vodiče na tipkalo kao na slici 5, unutar tipkala strujni krug bit će zatvoren ako tipkalo nije pritisnuto (slika 6). Ako u strujni krug uz tipkalo i izvor napajanja stavimo i žaruljicu (trošilo) u početnom stanju ona će svijetliti. Pritiskom na tipkalo žaruljica se gasi.



Slika 3.7: Tipkalo s priključnicama (1,2 i 3)

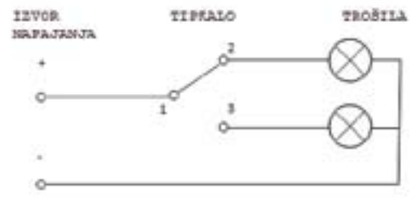


Slika 3.8: Shema spoja

Spojimo li vodiče na tipkalo kao na slici 7 unutar tipkala strujni krug bit će otvoren ako tipkalo nije pritisnuto (slika 8). Ako u strujni krug uz tipkalo i izvor napajanja stavimo i žaruljicu (trošilo) u početnom stanju ona neće svijetliti. Pritiskom na tipkalo žaruljica se pali.

3.3.1 Strujni krug s izmjeničnim tipkalom

Pomoću strujnog kruga s izmjeničnim tipkalom dobivamo izmjenično paljenje i gašenje žaruljica. Kada tipkalo nije pritisnuto, spojeni su kontakti 1 i 2 i svijetli gornja žaruljica. Pritisnemo li tipkalo spoje se kontakti 1 i 3, a gornja žaruljica se ugasi te upali donja žaruljica.



Slika 3.9: Strujni krug s izmjeničnim tipkalom

U informatici se rad računala temelji na dva definirana fizikalna stanja.
Ima impulsa (napona) – simbolička oznaka „1".
Nema impulsa (napona) – simbolička oznaka „0".

Elektronički sklopovi koji u računalu obavljaju razne operacije ponašaju se slično kao prekidači (tipkala). Preko tablice stanja (prijelaza) provjeravamo neke uvjete (npr. u kojem stanju žaruljica mora biti upaljena, a u kojem ugašena). U tablici su oznake za trenutno stanje tipkala, a druge mogućnosti uključuju sljedeće stanje. Podaci se u tablici čitaju red po red.

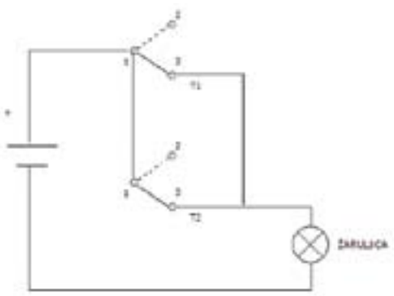
TABLICA STANJA:

Oznake u tablici stanja za tipkala su „0" i „1". Simboličkom oznakom „0" označavamo stanje kada tipkalo nije pritisnuto, a oznakom „1" označavamo stanje kada je tipkalo pritisnuto. U tablici ispod možemo vidjeti čitajući red po red da ukoliko tipkalo nije pritisnuto (stanje „0") svijetli prva (gornja) žaruljica, a ako pritisnemo tipkalo (stanje „1") svijetli druga (donja) žaruljica.

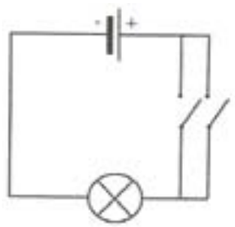
TIPKALO (T)	TROŠILO (ŽARILJICA)
0	Svijetli prva (gornja) žaruljica
1	Svijetli druga (donja) žaruljica

3.3.2 Paralelni spoj tipkala (logički sklop ili – „or")

Odabir načina spajanja tipkala (serijski ili paralelno) ovisi o tome što želimo postići kao krajnji rezultat na izlazu. Tipkala spajamo paralelno tako da povežemo po dvije iste priključnice tipkala (priključnicu 1 prvog tipkala na priključnicu 1 drugog tipkala te priključnicu 3 prvog tipkala na priključnicu 3 drugog tipkala). Ako imamo obična tipkala koja imaju dva pola, spojimo međusobno iste polove kao na slici 10.b. Logički sklop ILI/OR u informatici podrazumijeva da impulse ne dobivamo samo ukoliko su oba stanja u nuli. U ovom slučaju ako tipkala ne pritisnemo ona su u stanju nula te je strujni krug otvoren. Kod paralelnog spoja tipkala bez obzira koliko je tipkala pritisnuto (minimalno jedno), strujni krug se zatvara i žaruljica svijetli. Početno stanje tipkala ako nije pritisnuto jest da su (unutar tipkala) spojeni kontakti 1 i 2. U tom stanju strujni krug preko njega nije zatvoren. Ako pogledamo shemu, možemo vidjeti da ukoliko pritisnemo tipkalo „T1", spoje se kontakti 1 i 3 te se strujni krug zatvara preko toga tipkala i žaruljica svijetli. Isti princip koji vrijedi za tipkalo „T1" vrijedi i za tipkalo „T2". Žaruljica ne svijetli jedino u slučaju da nijedno tipkalo nije pritisnuto jer strujni krug nije zatvoren preko nijednog tipkala.



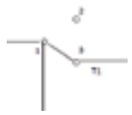
Slika 3.10 - a: Paralelni spoj Fischertechnik tipkala



Slika 3.10 - b: Shema paralelnog spoja tipkala

Legenda:

Spojeni kontakti 1 i 2 tipkala T1.



Spojeni kontakti 1 i 3 tipkala T1.

TABLICA STANJA:

Oznaka „0"označava stanje u kojem tipkalo nije pritisnuto, a oznaka „1" stanje u kojem je tipkalo pritisnuto. Ako pritisnemo bilo koje tipkalo (ako su „T1" ili „T2" u stanju „1"), palimo žaruljicu, a u početnom stanju dok tipkala nisu pritisnuta žaruljica je ugašena.

TIPKALO (T1)	TIPKALO (T2)	TROŠILO (ŽARULJICA)
0	0	Ne svijetli
1	0	Svijetli
0	1	Svijetli
1	1	Svijetli

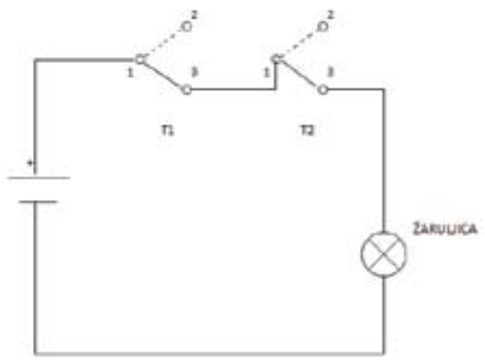
3.3.3 Serijski spoj tipkala (logički sklop i – „and")

Tipkala spajamo serijski tako da povežemo dvije priključnice tipkala unakrsno (priključnicu 1 prvog tipkala na priključnicu 3 drugog tipkala te priključnicu 3 prvog tipkala na priključnicu 1 drugog tipkala). Ako imamo obična tipkala koja imaju 2 pola, spojimo međusobno suprotne polove kao na slici 11.b. Logički sklop *I* u informatici podrazumijeva da impulse dobivamo ukoliko su oba stanja u jedinici. U ovom slučaju ako oba tipkala pritisnemo, ona su u stanju jedinica te je strujni krug zatvoren. U serijskom spoju električne komponente se spajaju redom, jedna za drugom tako da svim komponentama teče zajednička (ista) struja. Kod serijskog spoja tipkala strujni krug je zatvoren i žaruljica svijetli SAMO ukoliko su oba tipkala u stanju „1", tj. ukoliko su pritisnuta. Ako samo jedno tipkalo nije pritisnuto, žaruljica neće svijetliti.

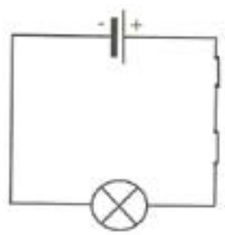
TABLICA STANJA:

Oznaka „0" označava stanje kada tipkalo nije pritisnuto, a oznaka „1" označava stanje kada je tipkalo pritisnuto. Žaruljica svijetli jedino onda kada su oba tipkala u stanju „1", tj. pritisnuta. U svim ostalim slučajevima žaruljica je ugašena.

TIPKALO (T1)	TIPKALO (T2)	TROŠILO (ŽARULJICA)
0	0	Ne svijetli
0	1	Ne svijetli
1	0	Ne svijetli
1	1	Svijetli



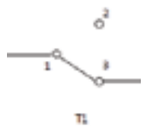
Slika 3.11 - a:Serijski spoj Fischertechnik tipkala



Slika 3.11 - b: Prikaz zatvorenog strujnog kruga serijskog spoja tipkala

Legenda:

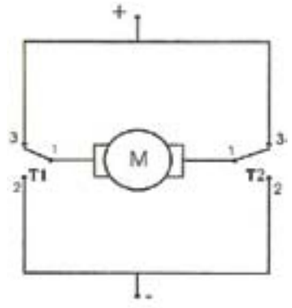
Spojeni kontakti 1 i 2 tipkala T1.



Spojeni kontakti 1 i 3 tipkala T1.

3.3.4 Upravljanje elektromotorom pomoću dva tipkala

Elektromotor je bitan dio robotskih kolica te robota općenito jer pomoću njega pokrećemo dijelove robota (robotsku ruku, kotače vozila itd.). Mijenjajući polaritet izvora (minus i plus), mijenja se i smjer vrtnje rotora motora. Smjerom vrtnje elektromotora možemo upravljati i s dva izmjenična tipkala kao na slici 3.12. Elektromotor će se vrtjeti u jednom smjeru kad je pritisnuto prvo tipkalo, a u drugom kad je pritisnuto drugo tipkalo. Ako su oba tipkala istovremeno pritisnuta ili otpuštena, elektromotor se neće vrtjeti.



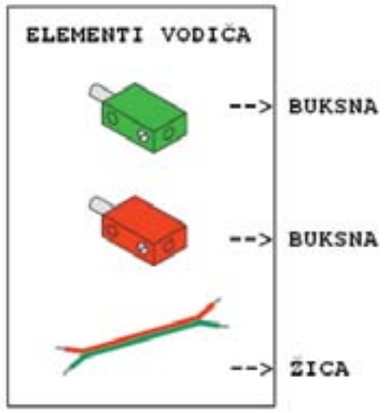
Slika 3.12: Shema spoja elektromotora i tipkala

ZADATAK ZA VJEŽBU

Nacrtati strujni krug s 2 tipkala i 2 trošila (žaruljice). Ako pritisnemo prvo tipkalo svijetlit će jedna žaruljica, a ako pritisnemo drugo tipkalo, svijetlit će druga žaruljica. Napraviti tablicu stanja.

3.4 Postupak izrade vodiča

Preko vodiča dobivamo napon na ostale dijelove robota. Buksne postavljamo na krajeve žica, kako bismo dobili vodič. Buksne su crvene i zelene boje, kako bismo se lakše snalazili (poželjno je da kod slaganja vodiča uvijek stavljamo iste boje buksni na oba kraja istoga vodiča). Kako bismo dobili vodič koji ćemo kasnije koristiti u robotici moramo nabaviti žicu i dvije buksne.



Slika 3.13: Elementi pomoću kojih sastavljamo vodič

Buksna je element Fischertechnik kompleta koji služi povezivanju ostalih elemenata npr. trošila, tipkala... Žica se sastoji od središnjeg dijela koji je vodič (npr. bakar) oko kojega se nalazi zaštitna izolacija. Potrebno je s krajeva žice ukloniti barem 5 mm izolacije kako bismo lakše mogli umetnuti dio bez izolacije u buksne (slika 3.14). Savinemo žicu preko izolacije da se bolje drži u buksni, umetnemo žicu u buksnu (slika 3.15) i zategnemo odvijačem vijak koji se nalazi na buksni: unutar nje pritegnemo žicu radi bolje vodljivosti.

Slika 3.14: Skinuta izolacija sa žica

Slika 3.15: Savijanje žice i umetanje u buksnu

Gotovi vodič možemo vidjeti na slici 3.16.

Slika 3.16: Gotovi vodič

3.4.1 Provjera ispravnosti vodiča univerzalnim instrumentom

Svaki dio strujnog kruga (svaki element u strujnom krugu) pruža neki otpor. Univerzalnim instrumentom možemo mjeriti jakost struje, otpor elemenata, napon (ovisno o tome na koje mjerno područje okrenemo sklopku), a u ovom slučaju mjerit ćemo otpor. Univerzalnim instrumentom provjeravamo ispravnost izrade vodiča da utvrdimo je li vodič ispravno napravljen, jer postoji mogućnost da vodič neće provoditi struju te nam je kao takav neupotrebljiv. Drugi problem bio bi ako nismo dovoljno dobro učvrstili vijak na buksni pa nam žica lako može ispasti iz buksne, a u tom slučaju vodič ne bi provodio struju na elemente.

Način mjerenja otpora univerzalnim instrumentom:

1. Na mjernom instrumentu odaberemo omsko mjerno područje – oznaka „200 Ω".
2. Mjernim instrumentom kratko spojimo krajeve vodiča te ispitamo vodič.

   a) Mjerni instrument trebao bi pokazati mali otpor (cca. 0,03).

   b) Kod mjerenja otpora žaruljice, nakon što spojimo instrumentom polove žaruljice, mjerni instrument trebao bi pokazati veći otpor (cca. 9 – 10 Ω).

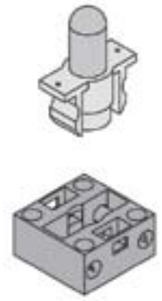
   c) Ako mjerni instrument pokazuje „1", to je pokazatelj da je došlo do kratkog spoja, tj. nije dobro spojeno!

Slika 3.17: Univerzalni instrument

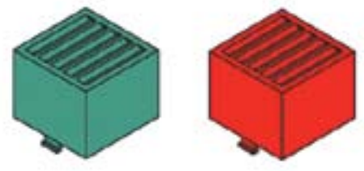
3.5 Sastavljanje trošila (žaruljica)

Zanimljiva vježba za učenike je sastavljanje trošila budući da se u strujne krugove često stavljaju žaruljice. Kako bismo sastavili žaruljicu, potrebno je uzeti žaruljicu i podnožje za žaruljicu u Fischertechnik elementima (slika 3.18).

Trebamo pronaći i pokrove žaruljica (slika 19).

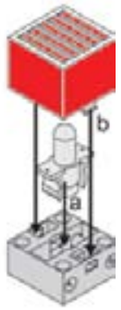


Slika 3.18: Žaruljica i podnožje za žaruljicu



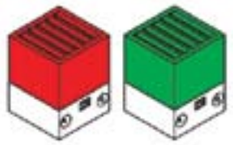
Slika 3.19: Pokrov žaruljice

Pokrovi su napravljeni u više boja – crvena, zelena, žuta... Boju pokrova treba prilagoditi zadatku ili odabrati proizvoljno. Kada smo našli sve potrebne elemente, žaruljicu trebamo umetnuti u predviđeno podnožje za žaruljicu te na tako spojeni dio umetnuti pokrov (slika 3.20).



Slika 3.20: Umetanje žaruljice u podnožje (a) te nakon toga stavljanje pokrova (b)

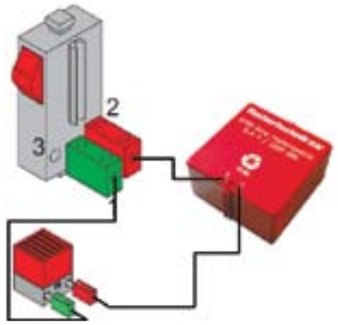
Žaruljicu sada prekriva pokrov, a gotova žaruljica izgleda kao na slici 3.21. Ukoliko spojimo vodiče preko izvora napajanja na u to predviđene priključnice koje se nalaze na podnožju žaruljice, mogli bismo vidjeti kako žaruljica svijetli u određenoj boji, ovisno o boji pokrova žaruljice.



Slika 3.21: Gotova žaruljica

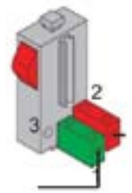
3.5.1 Strujni krug s izvorom napajanja (baterija), tipkalom i trošilom (žaruljica)

U slučaju da je u strujni krug spojeno tipkalo s baterijom i trošilom (žaruljicom) preko kontakta 1 i 2, žaruljica bi svijetlila – strujni krug bio bi zatvoren. Pritiskom na tipkalo, spojili bismo kontakte 1 i 3, žaruljica ne bi svijetlila – strujni krug bio bi otvoren.



Slika 3.22: Primjer 1 - spoj tipkala, trošila i izvora napajanja

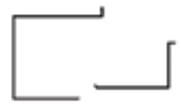
Objašnjenje oznaka sa slike:



Tipkalo

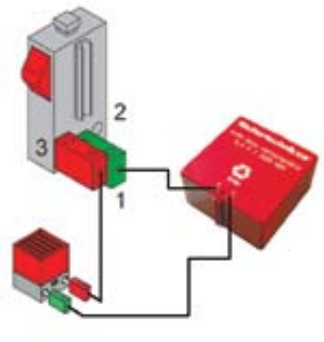
Izvor napajanja (baterija)

Trošilo (žaruljica) i vodič koji se spaja u trošilo



Vodiči

U primjeru 1 spojili smo kontakte tipkala 1 i 2, što znači da u stanju kada tipkalo nije pritisnuto žaruljica svijetli, a kada pritisnemo tipkalo, žaruljica se ugasi. Ukoliko u strujni krug spojimo tipkalo preko kontakata 1 i 3, žaruljica neće svijetliti – strujni krug bit će otvoren, međutim, pritiskom na tipkalo, spojili bismo kontakte 1 i 2 te zatvorili strujni krug, a žaruljica bi zasvijetlila.

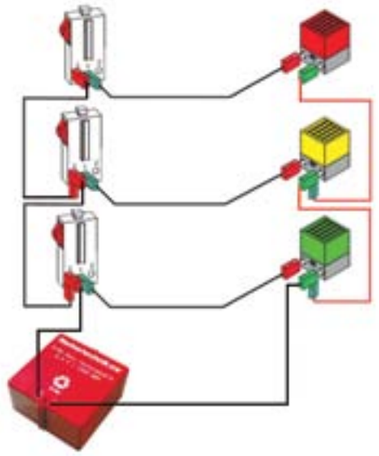


Slika 3.23: Primjer 2 - spoj tipkala, trošila i izvora napajanja

U primjeru 2 spojili smo kontakte tipkala 1 i 3 što znači da u stanju kada tipkalo nije pritisnuto žaruljica ne svijetli, a kada pritisnemo tipkalo, žaruljica se pali.

3.5.2 Spajanje više žaruljica s tipkalima

Prilikom spajanja više žaruljica s tipkalima, kontakte 3 na tipkalu moramo međusobno spojiti vodičima, a pritom jedan vodič s kontakta spojiti i s izvorom napajanja (baterijom). Kontakte 1 na tipkalima vodimo na jedan pol trošila (žaruljice), a drugi pol trošila (žaruljice) međusobno povežemo vodičima te s istog pola na koji smo spojili vodiče, odvedemo jedan vodič i na drugi pol izvora napajanja (baterija). Budući da su spojeni kontakti 1 i 3 na tipkalima, znači da u stanju kada tipkala nisu pritisnuta žaruljice ne svijetle, a ako pritisnemo tipkala žaruljice se pale.



Slika 3.24: Primjer spoja više tipkala s trošilima (žaruljicama)

ZADATAK ZA VJEŽBU

Potrebno je pomoću tipkala upravljati radom žaruljica i to na način da jedna žaruljica svijetli kad je tipkalo pritisnuto, a druga kad nije pritisnuto. Nacrtajmo shemu spajanja.

TABLICA STANJA:

Oznaka „0" označava da tipkalo nije pritisnuto, a oznaka „1" označava pritisnuto tipkalo.

TIPKALO	ŽARULJICA 1	ŽARULJICA 2
0	Svijetli	Ne svijetli
1	Ne svijetli	Svijetli

ZADATAK ZA VJEŽBU

Zadatak je pomoću dva tipkala upravljati radom žaruljica. U početnom stanju dok tipkala nisu pritisnuta svijetle dvije žaruljice. U bilo kojem drugom slučaju, one se gase. Nacrtajmo shemu spajanja uz pomoć donje tablice stanja.

TABLICA STANJA:

Oznaka „0"označava da tipkalo nije pritisnuto, a oznaka „1" označava pritisnuto tipkalo.

TIPKALO 1	TIPKALO 2	ŽARULJICA 1	ŽARULJICA 2
0	0	Svijetli	Svijetli
0	1	Ne svijetli	Ne svijetli
1	0	Ne svijetli	Ne svijetli
1	1	Ne svijetli	Ne svijetli

3.6 Semafor

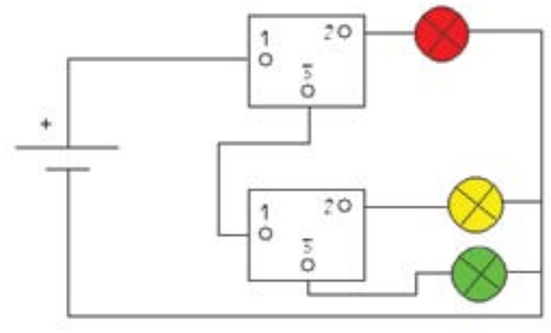
Način rada semafora imamo prilike upoznati kada izađemo na ulicu. Semafor je uređaj za signalizaciju i regulaciju prometa. Najčešće se postavlja na raskrižju, pješačkom prijelazu, željezničkoj pruzi ili na bilo kojem drugom mjestu gdje će služiti za reguliranje prometa. Prema međunarodnoj normi, temeljne boje su mu crvena, žuta i zelena. Rad semafora zasniva se na tome da je u jednom trenutku upaljeno crveno svijetlo, nakon toga se uz crveno upali i žuto svijetlo, te na kraju zeleno, dok se paralelno crveno i žuto gase. Mi ćemo simulirati rad semafora pritiskom na određena tipkala. Potrebni elementi su: dva tipkala, tri žaruljice (crveni, žuti i zeleni pokrov za žaruljice) te izvor napajanja (baterija 9V).

TABLICA STANJA

Oznaka „0" označava da tipkalo nije pritisnuto, a oznaka „1" označava pritisnuto tipkalo.

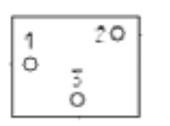
TIPKALO 1	TIPKALO 2	TROŠILO (ŽARULJICA)
0	0	Crvena žaruljica svijetli
1	0	Crvena i žuta žaruljica svijetle
0	1	Zelena žaruljica svijetli

U tablici stanja vidimo da na početku mora svijetliti crvena žaruljica (ukoliko tipkala nisu pritisnuta). Ako pritisnemo prvo tipkalo, svijetle crvena i žuta žaruljica, a ako pritisnemo drugo tipkalo, svijetli zelena žaruljica.

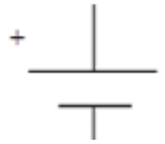


Slika 3.25: Shema strujnog kruga s tri trošila (žaruljice) i dva prekidača (tipkala)

Objašnjenje oznaka sa slike:



Tipkalo s priključnicama(1,2 i 3)

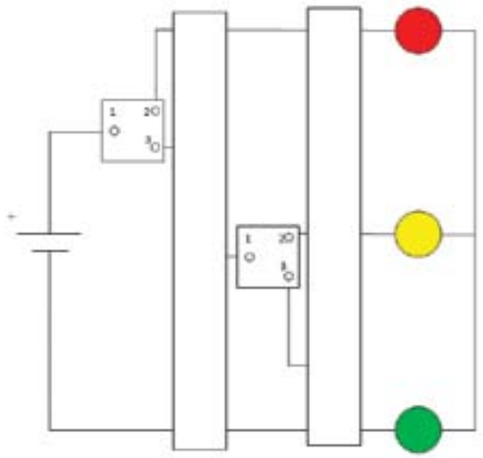


Izvor napajanja

Trošila (žaruljice)

ZADATAK ZA VJEŽBU

Ispod je prikazana nedovršena shema strujnog kruga s tri žaruljice i dva tipkala (prekidača). Dopuni shemu kako bismo dobili pravilno spojen strujni krug s tri trošila i dva tipkala, kojim bi simulirali rad semafora.

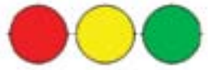


Slika 3.26: Primjer sheme kojom se simulira rad semafora (nedopunjena shema)

Objašnjenje oznaka sa slike:

Izvor napajanja

Tipkalo (priključnice 1, 2 i 3)



Trošila (žaruljice) – dovršiti izgled simbola

Mjesto preko kojeg treba spojiti vodiče

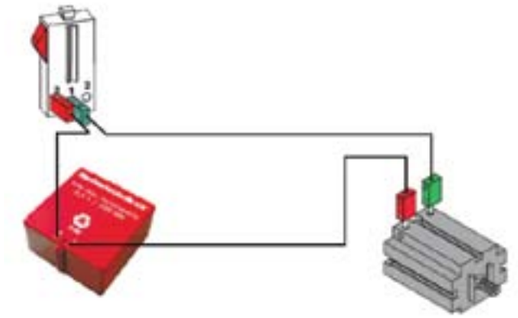
TABLICA STANJA

Oznaka „0" označava da tipkalo nije pritisnuto, a oznaka „1" označava pritisnuto tipkalo. Crvena žaruljica svijetli kada tipkala nisu pritisnuta. Pritisnemo li prvo tipkalo uključit će se žuta žaruljica. Ako je pritisnuto prvo i drugo tipkalo, uključuje se zelena žaruljica, a žuta se gasi. Ni u jednom trenutku ne svijetle dvije žaruljice zajedno.

1. TIPKALO	2. TIPKALO	STANJE ŽARULJICA
0	0	Crvena žaruljica svijetli
1	0	Žuta žaruljica svijetli
1	1	Zelena žaruljica svijetli

3.7 Strujni krug s izvorom napajanja (baterija), tipkalom i motorom (pokretanje motora)

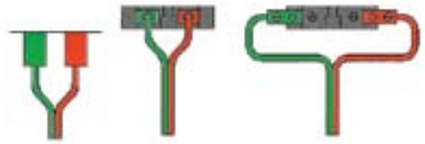
Želimo li dobiti upravljač kojim upravljamo radom robotskih kolica preko tipkala potrebno je u strujni krug staviti tipkalo i motor. Pokretanje rotora elektromotora dobivamo ako ga vodičima priključimo na izvor napajanja (bateriju). Rotor se počinje okretati u jednu stranu. Želimo li da se rotor elektromotora okreće u suprotnom smjeru od prethodnoga, trebamo promijeniti polaritet napajanja na elektromotoru. To dobivamo tako da zamijenimo vodiče koji su spojeni na lijevoj i desnoj strani motora. Motor preko zupčanika pokreće kotače te se robotska kolica mogu pokretati (pogonski zupčanik gura zupce drugog zupčanika koji se nalazi na mjenjačkoj kutiji). Razmak i veličina zubaca moraju biti potpuno jednaki jer trebaju sjedati jedan u drugog. Zaustavljanje rada motora moguće je ako u strujni krug spojimo i tipkalo, pomoću kojeg možemo isključiti strujni krug.



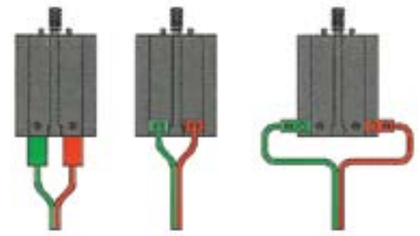
Slika 3.27: Primjer spajanja tipkala, izvora napajanja i motora

Objašnjenje oznaka sa slike:

Motor



Vodiči označeni crvenom i zelenom bojom



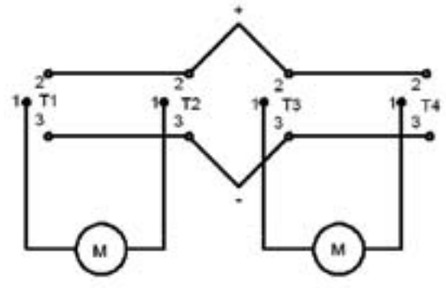
Slika 3.28: Primjer spajanja vodiča na motore

Slika 3.29: Prikaz rada zupčanika

Robotska kolica moraju imati dva elektromotora. Svaki elektromotor pokreće svoj kotač (jedan lijevi kotač, a drugi desni kotač). Želimo li robotska kolica pokrenuti prema naprijed ili unatrag, potrebno je priključiti napon na oba elektromotora (pritom moramo paziti na polaritete napajanja na elektromotorima) da dobijemo okretanje oba kotača u istom smjeru. Pogriješimo li pri spajanju napona na elektromotore, kolica će se početi okretati oko svoje osi ulijevo ili udesno. Taj način kretanja koristio bi nam ako želimo kolica okrenuti na manjem prostoru.

3.8 Shema spajanja tipkala i motora robotskih kolica (upravljanje vozilom pomoću tipkala)

Da bi vozilo bilo potpuno funkcionalno mora se moći kretati naprijed – natrag, lijevo – desno. Želimo li kontrolirati dva elektromotora, moramo upotrijebiti četiri tipkala.
Potrebni elementi: 2 elektromotora (oznake „M"), 4 tipkala (oznake „T1", „T2", „T3" i „T4") i vodiči.



Slika 3.30: Shema spajanja motora, tipkala i izvora napajanja za vožnju robotskih kolica

Objašnjenje oznaka sa slike:

Tipkalo (1, 2 i 3 su priključnice tipkala)

Izvor napajanja

Motor

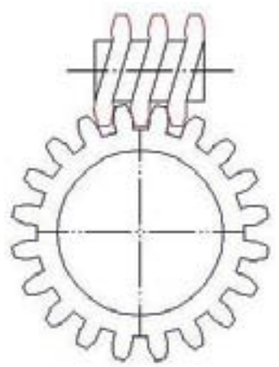
Vodiči

Oznaka spoja (mjesto gdje se spajaju vodiči na elemente)

Jedna strana svih tipkala spaja se na minus pol baterije, a druga strana na plus pol. Srednji izvodi tipkala spajaju se na elektromotore (npr. dva lijeva tipkala spajaju se na lijevi elektromotor, a dva desna tipkala spajaju se na desni elektromotor). Vozilo bi išlo naprijed ako su pritisnuta dva krajnja tipkala. Pritisnu li se dva unutarnja tipkala, vozilo bi išlo unatrag. Ukoliko se elektromotor ne vrti u željenom smjeru, na njemu se zamjene vodiči na priključnicama.

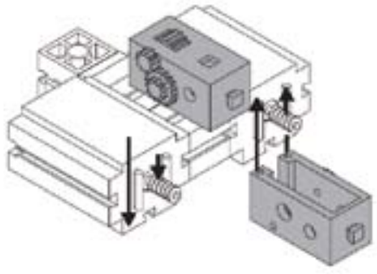
3.8.1 Primjer spajanja motora, mjenjačke kutije i kotača na osovini

Mjenjačka kutija je uređaj koji prenosi snagu motora na kotače vozila. Prijenos snage se najčešće vrši pužnim prijenosom. To je najlakše izvesti spajanjem motora i mjenjačke kutije.

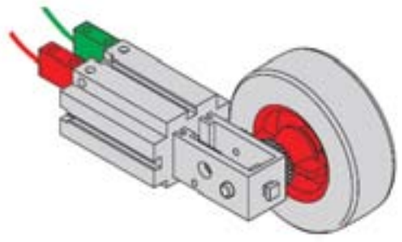


Slika 3.31: Pužni prijenos

Zupčanici koje možemo vidjeti ako pogledamo Fischertechnik motor izrađeni su u obliku vijka kako bi sjedali u drugi zupčanik kojeg treba pogoniti. Spajanjem motora na izvor napajanja zupčanik koji se nalazi na motoru počne se okretati u jednom smjeru (naziva se pogonski zupčanik), a prilikom spajanja na mjenjačku kutiju navoji toga vijka ulaze među zupce zupčanika koji se nalazi na mjenjačkoj kutiji i tako ga pokreću (naziva se gonjeni zupčanik jer ga pokreću zupčanici motora). Zupci zupčanika na taj način prenose okretaje i snagu. Osovinu koja se nalazi u Fischertechnik elementima spajamo na mjenjačku kutiju te se preko te osovine gibanje dalje prenosi na kotače vozila.



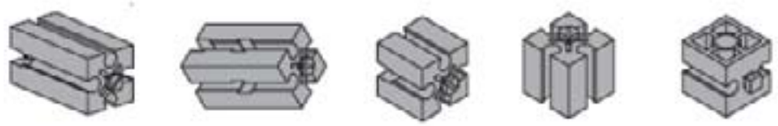
Slika 3.32: Spajanje mjenjačke kutije na motor



Slika 3.33: Spajanje kotača preko osovine na mjenjačku kutiju i motor

3.9 Elementi za slaganje vozila - robotskih kolica

Pomoću Fischertechnik elemenata možemo slagati robote, robotska kolica te, ukratko, sve što poželimo.



kockice za spajanje elemenata

mjenjačka kutija

motor

zupčanici, u kombinaciji s osovinom dobivamo prijenos okretaja na kotače

osovina i držač za stabilnost osovine

kotač

oplata (postavljamo ih unutar kotača)

matica (maticom pričvrstimo oplatu na kotačima)

primjer elemenata kojima dobivamo dodatni kotač za stabilnost kolica

spojnice (pomoću njih možemo povezati elemente)

Slika 3.34: Primjeri elemenata za slaganje robota

3.9.1 Primjeri slaganja robotskih kolica (korak po korak)

Za slaganje robotskih kolica potrebni su nam Fischertechnik elementi. Robotska kolica možemo složiti na više načina, a način slaganja i završni izgled vozila ovise o nama i našoj mašti.

Slika 3.35: Elementi za izradu vodiča i izrada vodiča za spajanje na izvor napajanja

Slika 3.36: Spajanje mjenjačke kutije na motor (2 komada za 2 kotača)

Slika 3.37: Slaganje izgleda robotskih kolica pomoću kockica za spajanje elemenata

Slika 3.38: Slaganje dodatnih kotača za stabilnost kolica

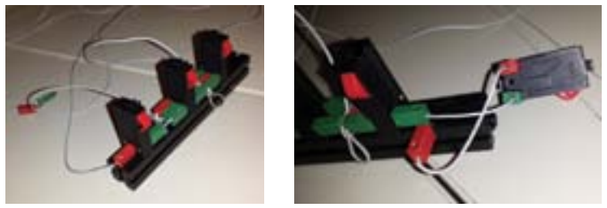
Slika 3.39: Montiranje dodatnih kotača

Slika 3.40: Elementi za slaganje kotača

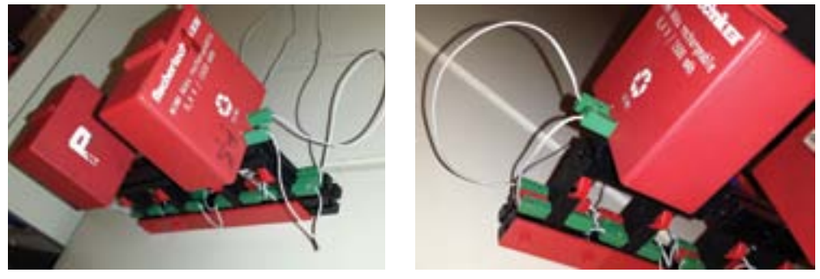
Slika 3.41: Spajanje kotača i osovine u kombinaciji s oplatom i maticama

Slika 3.42: Umetanje osovine kotača na mjenjačku kutiju koja je povezana na motor

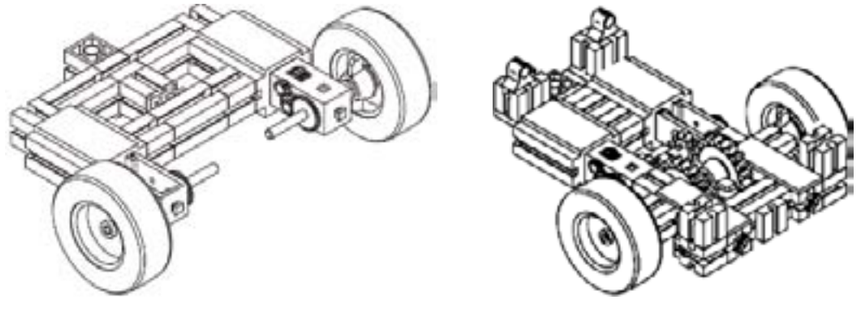
Slika 3.43: Spajanje izvora napajanja na robotska kolica pomoću vodiča



Slika 3.44: Slaganje tipkala za upravljanje robotskim kolicima



Slika 3.45: Spajanje izvora napajanja na tipkala (za ručno upravljanje vozilom)



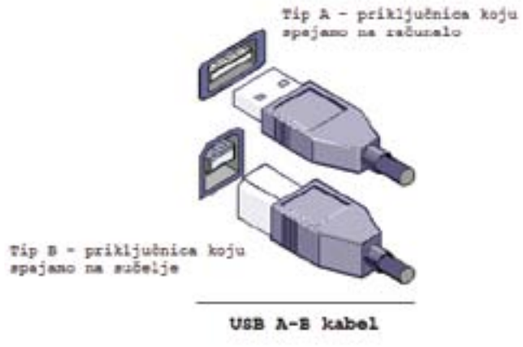
Slika 3.46: Primjeri gotovih robotskih kolica

3.10 Sučelje za spajanje robotičkih sustava

Sučelja za spajanje robotičkih sustava mogu biti u raznim izvedbama, ali na svima nalazimo utore za spajanje žaruljica, motora, izvor napajanja i USB kabela. Na slici 47. vidimo razne izvedbe sučelja (interface) na koja se mogu spojiti roboti. Svrhovito obavljanje zadaća robotskog sklopa postiže se tek spajanjem robota s računalom kako bismo ih pokrenuli preko sučelja naredbama iz računala.

Slika 3.47: Primjeri sučelja

Slika 3.48: Kabel za spajanje sučelja i računala (USB kabel TIP A-B)



Slika 3.49: Priključnice USB kabela TIP A-B

Na sučeljima postoje „ulazi" i „izlazi". „I" je oznaka za ulaz (engl. input). „O" označava izlaze (engl. output). Na ulaze spajamo tipkala i senzore. Na izlazne izvode označene kao četiri para izvoda od „M1" do „M4" ili kao izvodi od „O1" do „O8" spajamo trošila (elektromotore, žaruljice...). Oznaka „M" označava motor, a kako imamo četiri para izvoda na sučelja možemo spojiti četiri elektromotora ili osam drugih trošila (npr. žaruljice). Prvi motor spajamo na izvode označene oznakom „M1", drugi na „M2", treći na „M3", a četvrti na „M4". Žaruljice spajamo na sučelje na način da jedan pol žaruljice spajamo na izlaze od „O1" do „O8", a drugi pol spajamo na konektor mase koji se nalazi na sučelju (oznaka ⊥). Na taj način možemo spojiti do 8 žaruljica, svaku na jedan izlaz i zajedno na masu.

Slika 3.50: Spajanje robotskih kolica na sučelje (za automatsko upravljanje robotskim kolicima pomoću računala)

Slika 3.51: Spajanje robotskih kolica, sučelja i računala

3.10.1 Primjeri robota i robotskih kolica sa sučeljem

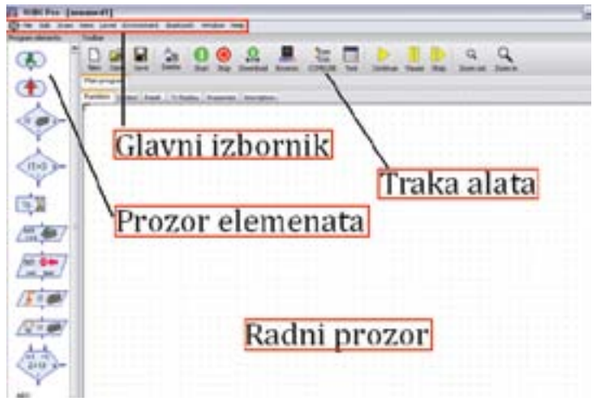
Na slikama 52. i 53. mogu se vidjeti sučelja povezana s robotima koje smo dobili slaganjem Fischertechnik elemenata.

Slika 3.52: Primjer spajanja tipkala, žaruljica i sučelja

Slika 3.53: Primjer spajanja robotskih kolica i sučelja

3.11 Sučelje programa RoboPro

Robota treba „naučiti", tj. programirati što mora raditi. Svi potrebni podaci za rad robota upisuju se u program već pri njegovoj izradi. Kada se program pokrene robot „zna" što mu je činiti. To su uglavnom jednostavni programi u kojima je određeno koliko će se dugo koji od motora robota okretati naprijed ili natrag, koliko će vremena koja od žaruljica svijetliti itd. RoboPro je zanimljiv program jer nam za rad u njemu nije potrebno posebno poznavanje programiranja, već programi nastaju uz pomoć ikona koje su slikovito napravljene (u obliku sličica).



Slika 3.54: Prikaz sučelja programa RoboPro

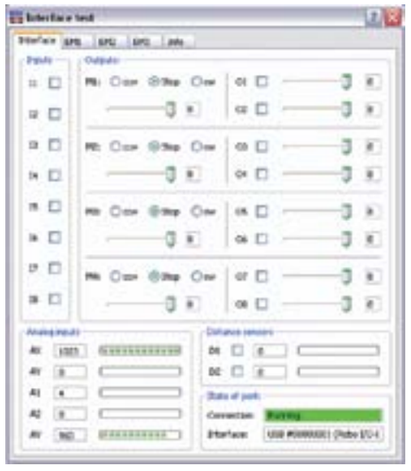
Za potrebne elemente u programu, u izborniku „Level" odaberemo „Level 4: Objects". Odabirom te opcije s lijeve strane prikaže se izbornik „Program elements" te u njemu podizbornik „Basic elements" u kojem su prikazani svi elementi koji su nam potrebni za programiranje rada robota (elementi žaruljica, motora itd.).

Slika 3.55: Izbornik „Level" unutar programa RoboPro

Prvi korak nakon spajanja robotičkog sučelja i računala je podešavanje parametara rada USB sučelja RoboPro programa. U traci s alatima odaberemo ikonu „COM USB" za mogućnosti ulaznog sučelja (Set interface port options) te u novom prozoru odaberemo postavke izlaznog sučelja.

Slika 3.56: Skočni prozor u programu RoboPro koji prikazuje postavke izlaznog sučelja

Drugi korak je provjera ispravnosti spoja između sučelja i računala, tj. provjera rada sučelja koju provodimo klikom na ikonu „Test" za provjeru rada sučelja (Test interface) te se otvara prozor za testiranje sučelja. U dnu prozora vidimo zelenu traku s tekstom Running, što znači da je sučelje ispravno spojeno i da radi.



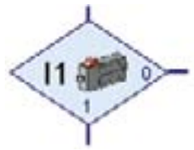
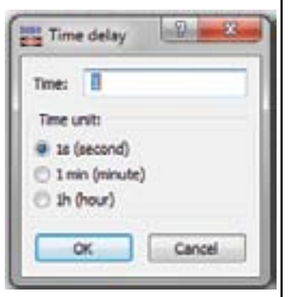
Slika 3.57: Prikaz prozora za provjeru ispravnosti spoja sučelja i računala

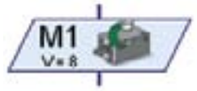
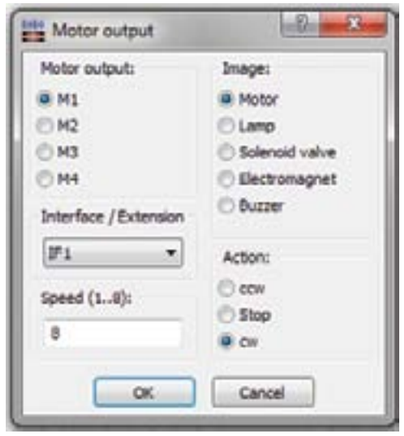
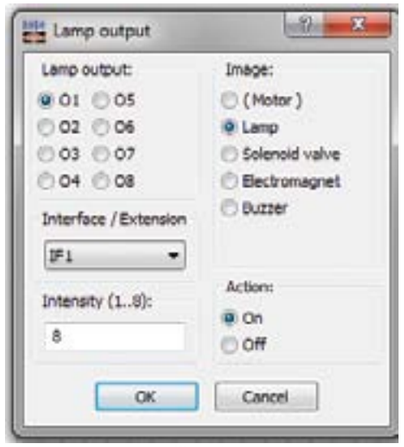
Ukoliko nam sučelje nije spojeno s računalom, a u tom trenutku pritisnemo ikonu „Test", prikazat će nam se poruka kao na slici 58., koja nam govori da sučelje nije spojeno s računalom.

Slika 3.58: Poruke koje govore da sučelje nije spojeno s računalom

3.11.1 RoboPro - Program elements – basic elements

Ikone se iz prozora elemenata u radni prozor prebacuju opcijom primi i ispusti (drag and drop). U prozoru elemenata označimo ikonu koju želimo staviti u program (na ikonu kliknemo lijevom tipkom miša i držimo pritisnutu tipku miša) te ikonu odvučemo u radni prozor i ispustimo. Na taj način slažemo programe i međusobno spajamo elemente.

	Ikona za početak programa („Start"). Početak pisanja programa. Ikonu stavljamo na početak pisanja programa
	Ikona za kraj programa („End"). Završetak pisanja programa. Ikonu stavljamo na kraj napisanoga programa.
	Ikona za tipkalo („Digital branch"). Na ikoni osim oznake broja tipkala („I1") imamo i brojeve „0" i „1" koji označavaju stanja pritisnutog ili otpuštenog tipkala. Oznaka „0" označava stanje kada tipkalo nije pritisnuto, a oznaka „1" stanje kada je tipkalo pritisnuto. Desnim klikom miša na ikonu otvaraju nam se dodatna svojstva (slika 59) kojima možemo promijeniti oznake položaja 0 / 1 u 1 / 0 i odabrati kojim tipkalom radimo (oznake od „I1" do „I8"). Ako želimo provesti neku radnju onda kad je tipkalo pritisnuto, tu radnju trebamo spojiti s brojkom „1". Ako se neka radnja provodi dok tipkalo nije pritisnuto, ikonu spojimo s brojkom „0".  *Slika 3.59*
	Ikona za vremenski odmak („Time delay"). Ikonu stavljamo u program na mjesto gdje čekamo određeno vrijeme na provođenje neke radnje (npr. upali nam se žaruljica i želimo da gori jednu sekundu i da se nakon te sekunde ugasi ili želimo da nam se motor okreće tri sekunde te nakon toga stane). Desnim klikom miša na ikonu prikazuju se svojstva (slika 3.60) u kojima možemo promijeniti vrijeme (sekunde/ minute/sati) i napisati koliko vremena treba čekati do iduće radnje koja se izvodi nakon ove ikone.  *Slika 3.60*

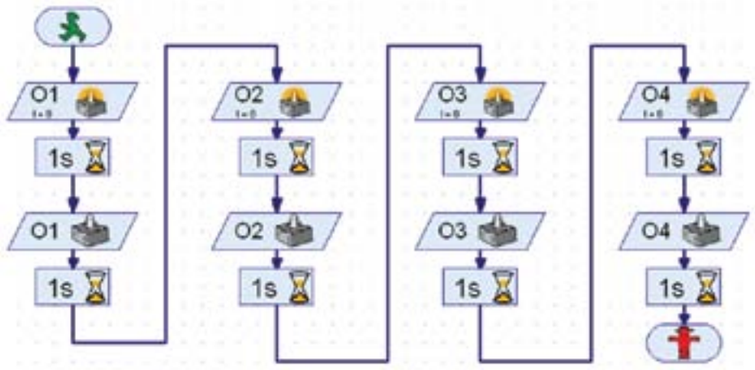
	Ikona kojom reguliramo rad motora („Motor output"). Desnim klikom miša na ikonu dobivamo svojstva (slika 3.61) unutar kojih odabiremo vrti li se motor ili stoji, odabiremo i smjer vrtnje rotora motora: ccw – okretanje rotora motora u smjeru obrnutom od kazaljke na satu, cw (clock wise) – okretanje rotora motora u smjeru kazaljke na satu. Također možemo odabrati kojom se brzinom rotor motora okreće.  *Slika 3.61:*
	Ikona kojom reguliramo rad trošila („Lamp output"). Simbol trošila (žaruljica). Desnim klikom miša na ikonu otvaramo svojstva (slika 3.62) unutar kojih mijenjamo stanja trošila – on/off, ukoliko žaruljica nakon neke radnje mora biti upaljena ili ugašena. Također biramo ime trošila kojim radimo (oznake od „O1" do „O8").  *Slika 3.62:*

3.12 Primjeri programa koji se mogu raditi s učenicima

Program 1 – trčeće svjetlo

Zadatak je dobiti trčeće svjetlo. Trčeće svjetlo znači da prva žaruljica svijetli određeni period, tada se ugasi i ugašena je neko vrijeme, nakon toga se upali druga žaruljica, gori određeno vrijeme i tako u krug. U našem primjeru period u kojemu je određena žaruljica upaljena tj. ugašena je jedna sekunda.

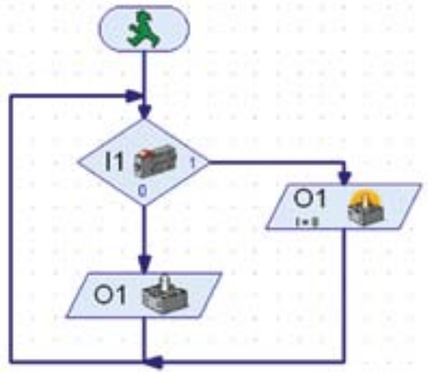
Potrebni elementi: četiri žaruljice, vodiči, baterija, robotičko sučelje koje spajamo na računalo preko USB kabla, USB kabel, kabel za napajanje.



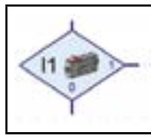
Slika 3.63: Program u programu RoboPro (simulacija uzastopnog paljenja i gašenja žaruljica)

Program 2 – tipkalo (prekidač, sklopka)

Potrebno je postaviti tipkalo i žaruljicu na elemente za slaganje kako bismo osigurali stabilnost svih elemenata koje koristimo u ovoj vježbi te ih pravilno povezati sa sučeljem (tipkalo i žaruljicu). Zadatak: kada pritisnemo tipkalo, žaruljica svijetli; kada tipkalo nije pritisnuto, žaruljica ne svijetli. Potrebni elementi: tipkalo, žaruljica, baterija, robotičko sučelje koje spajamo na računalo preko USB kabla, USB kabel, kabel za napajanje.

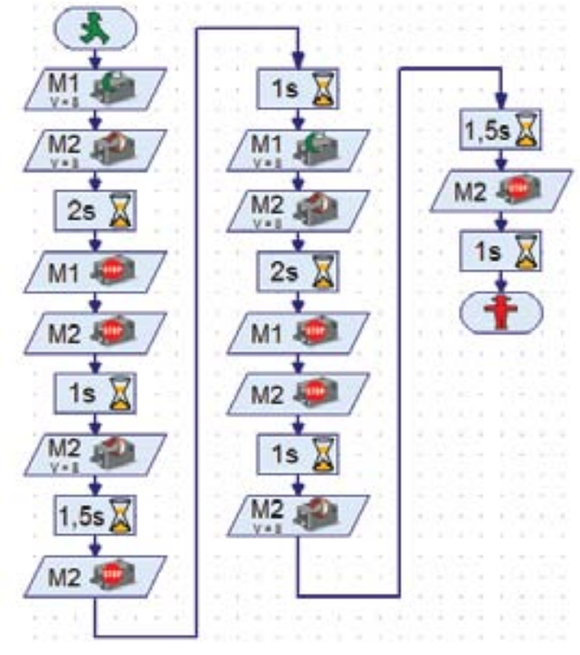


Slika 3.64: Program u programu RoboPro (simulacija paljenja i gašenja žaruljica pomoću tipkala)

	- oznaka koja nam govori da na ovom mjestu gledamo je li tipkalo pritisnuto ili ne. Ukoliko je tipkalo pritisnuto, provodi se iduća radnja koja se nastavlja od brojke „1" (pali se žaruljica). Ukoliko tipkalo nije pritisnuto, provodi se radnja koja se nalazi ispod broja „0" (žaruljica se gasi).

Program 3 - vožnja robotskih kolica

Od Fischertechnik elemenata potrebno je konstruirati robotska kolica kojima možemo upravljati putem računala. Zatim motore robotskih kolica moramo pravilno povezati na sučelje. Zadatak vježbe je da se oba motora („M1" i „M2") pokrenu u isto vrijeme (vozilo ide prema naprijed), vozi dvije sekunde. Nakon dvije sekunde vozilo se zaustavlja (oba rotora motora prestanu s radom), čekaju jednu sekundu do idućeg koraka. Nakon jedne sekunde samo motor „M2" počne s radom te se rotor motora „M2" pokreće jednu i pol sekundu, dok u istom trenutku motor „M1" stoji, a oko osi tog motora vozilo se zakreće. Poslije zakretanja motor se zaustavlja na jednu sekundu. Nakon što se vozilo zakreće dvije sekunde oko osi motora „M1", oba se rotora motora pokrenu te nakon toga stanu. Vozilo je u stanju mirovanja 1 sekundu nakon čega opet počne s radom samo motor „M2", koji se opet pokrene na jednu i pol sekundu, što znači da se vozilo opet zakreće u jednom smjeru oko osi motora „M1" te nakon toga staje.
Potrebni elementi: gotova robotska kolica s dva motora, vodiči, baterija, robotičko sučelje koje spajamo na računalo preko USB kabela, USB kabel, kabel za napajanje.



Slika 3.65: Program u programu RoboPro za automatsko upravljanje robotskim kolicima

	- oznaka da se motor broj 1 okreće u smjeru kazaljke na satu
	- oznaka da se motor broj 2 okreće u smjeru obrnutom od kazaljke na satu
	- oznaka da se motor broj 1 zaustavlja u ovom trenutku

Program 4 – vozilo koje prati crtu

Osim što ih možemo programirati da odvoze krug, robotska kolica možemo programirati i na način da prate npr. crnu crtu na bijeloj podlozi te na taj način prijeđu neki put od točke A do točke B.

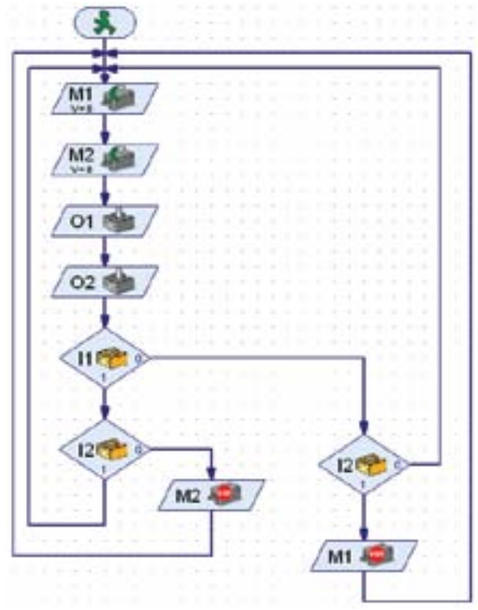
Slika 3.66: Fischertechnik element fototranzistora

Jedno od svojstava tipkala kojeg možemo vidjeti tako da na ikonu tipkala u programu RoboPro kliknemo desnom tipkom miša (slika 67) jest i fototranzistor (elektronički dio koji zatvori strujni krug kad na njega padne svjetlo, a otvori ga kad svjetla nema). Sličan princip kao i kod tipkala (kad tipkalo pritisnemo, uključimo, tj. isključimo strujni krug).

Slika 3.67: Promjena ikone tipkala u fototranzistor u programu RoboPro

Način slaganja fototranzistora na robotska kolica: dva fototranzistora spojimo na prednji kraj ispod vozila s čije lijeve strane su elektromotor 1 i fototranzistor 1, a s desne strane elektromotor 2 i fototranzistor 2. Ako ispred fototranzistora stavimo žaruljice, one bi osvjetljavale bijelu podlogu, a svjetlost bi se od bijele podloge odbijala i padala na otvore fototranzistora. Računalni program provjeravao bi stanje tih fototranzistora. Ako se svjetlost odbija od podloge, zatvara se strujni krug te se vozilo pokreće. Crna podloga koja je u stvari crna crta koju vozilo prati ne odbija svjetlost na fototranzistore. Crnu crtu najlakše simuliramo tako da na podlogu nalijepimo crnu izolir-traku.

Fototranzistori koji se nalaze na vozilu ne smiju biti iznad trake zbog odbijanja svjetlosti od podloge jer želimo da nam se vozilo kreće po crti. Kad pokrenemo vozilo oba su fototranzistora postavljena iznad bijele podloge te primaju odbijeno svijetlo. U trenutku kad vozilo dođe do zavoja, jedan fotosenzor našao bi se iznad crne podloge što bi značilo kao da je strujni krug prekinut, tj. otvoren te vozilo mora skrenuti tako da su oba fototranzistora opet iznad bijele podloge). Žaruljice spajamo na izlaze sučelja npr. „O1" i „O2". Fototranzistore spajamo na ulaze „I1" i „I2" (ako ne reagiraju, potrebno im je zamijeniti polaritet). Elektromotore spajamo na priključnice sučelja „M1" i „M2".



Slika 3.68: Program u programu RoboPro za upravljanje vozilom koje prati crtu

	- oznaka fototranzistora

PRIJEDLOG ZADATKA ZA VJEŽBU

VOŽNJA KOLICA PO ZADANOM PUTU

Programiraj robotska kolica da se vrte u krug.

PRINCIP RADA SEMAFORA

Napravi robotski sustav s 3 žaruljice (crveni, žuti i zeleni pokrov). Spoji žaruljice na postolje jednu ispod druge (crvena, ispod nje žuta, i ispod žute zelena). Programiraj žaruljice da se pale i gase po principu rada semafora. Rad semafora zasniva se na tome da je u jednom trenutku upaljeno crveno svijetlo, nakon toga se uz crveno upali i žuto svijetlo te na kraju zeleno, dok se paralelno crveno i žuto gase.

4. poglavlje

WORDPRESS - sustav za upravljanje sadržajem

8. razred

4.1 Wordpress - uvod

4.1.1 Dinamička mrežna stranica - uvod

Krajem 20. i početkom 21. stoljeća povećana potreba korisnika za raznolikim internetskim sadržajima ubrzala je razvoj internetskih tehnologija. Internetske tehnologije su spoj mrežne infrastrukture i programskih rješenja koja omogućavaju komunikaciju putem internetske mreže. Potreba za uvođenjem dinamičkog sadržaja u mrežne aplikacije dovela je do razvoja složenih internetskih tehnologija. Mrežnim aplikacijama smatraju se aplikacije koje omogućavaju prikaz dinamičkog sadržaja. Dinamički sadržaj podrazumijeva podatke na mrežnoj stranici koji se mijenjaju za vrijeme korištenja te mrežne stranice.

Postupak izgradnje dinamičkih mrežnih stranica složen je i dugotrajan posao, a ovdje ću predstaviti jednostavan način stvaranja dinamičkog *weba*.

Mrežni (*web*) dokumenti se dijele u sljedeća tri tipa:

1. Statički mrežni dokumenti
2. Dinamički mrežni dokumenti
3. Aktivni mrežni dokumenti

Dinamičke stranice su ugrađeni programi koji omogućavaju automatske promjene nekih podataka na stranicama bez intervencije autora stranice, npr. ugrađene skripte ili povezanost s nekom pozadinskom aplikacijom ili bazom podataka.

4.1.2 *WordPress* - predstavljanje platforme

WordPress je vrhunska platforma za osobno izdavaštvo s naglaskom na estetiku, mrežne standarde i lakoću uporabe. *WordPress* je istovremeno i besplatan i neprocjenjiv. Jednostavnije rečeno, *WordPress* upotrebljavate kada se želite baviti svojim *blogom*, a istovremeno želite izbjeći teškoće koje se mogu pojaviti sa softverom za izradu.
(izvor: http://hr.wordpress.org/)

Unatoč brojnim gotovim predlošcima, koje nudi *WordPress*, dobro je upoznati jednostavnu dinamičku mrežnu stranicu da biste je mogli prilagoditi i izraditi vlastitu temu *WordPressa*. Boljim poznavanjem PHP-a, HTML-a i CSS-a možete jednostavno stvoriti i prilagoditi bilo koju temu.

PHP (*Hypertext Preprocessor*), prije Personal *Home Page Tools*, je programski jezik koji se orjentira po sintaksi C i Perl. Namijenjen je, prije svega, programiranju dinamičnih mrežnih stranica. PHP se ističe širokom podrškom raznih baza podataka i internetskih protokola kao i raspoloživošću brojnih programerskih knjižnica.

HTML (*HyperText Markup Language*) je programski jezik koji upotrebljavamo za kreiranje dokumenata na *World Wide Webu* (www). HTML se rabi za stvaranje hipertekstualnih datoteka (datoteka koje sadržavaju *linkove*).

CSS (*Cascading Style Sheets*) je jezik za oblikovanje stila koji određuje izgled HTML dokumenta, npr. CSS može određivati vrstu slova, boje, margine, crte, visinu, širinu, pozadinsku sliku, napredno pozicioniranje i štošta drugo. HTML također može biti upotrijebljen za određivanje izgleda mrežne stranice. No, CSS nudi više mogućnosti, točniji je i profinjeniji. CSS podržavaju svi današnji preglednici.

HTML rabimo za određivanje strukture sadržaja, a CSS upotrebljavamo za njegovo oblikovanje. Odvajanje stila mrežne stranice od sadržaja dokumenta, učinilo je održavanje mnogo lakšim.

U početku trebamo znati nekoliko osnovnih stvari da bismo kreirali jednostavnu temu *WordPressa*. Trebamo znati koje su datoteke potrebne za stvaranje uobičajene teme *WordPressa*. Zatim, trebamo upoznati način izrade predloška za svaku datoteku te, na kraju, kako sve zajedno ukomponirati unutar *WordPressa*. To je sve što vam je potrebno za stvaranje osnovne teme.

Dakle, krenimo sa stvaranjem vlastite teme *WordPressa* u nekoliko jednostavnih koraka. Prvo moramo odlučiti želimo li kreirati mrežnu stranicu izravno na poslužitelj, ili lokalno, na svojemu računalu. Svakako bih preporučio lokalni rad jer izravno kreiranje na mrežnom poslužitelju pokazuje nedostatke. Ukoliko želite raditi izravno na mrežnom poslužitelju, postupak je isti. Upotrijebite WinSCP ili neki drugi FTP program. O tome kako se rabe FTP programi možete pročitati u posljednjem poglavlju.

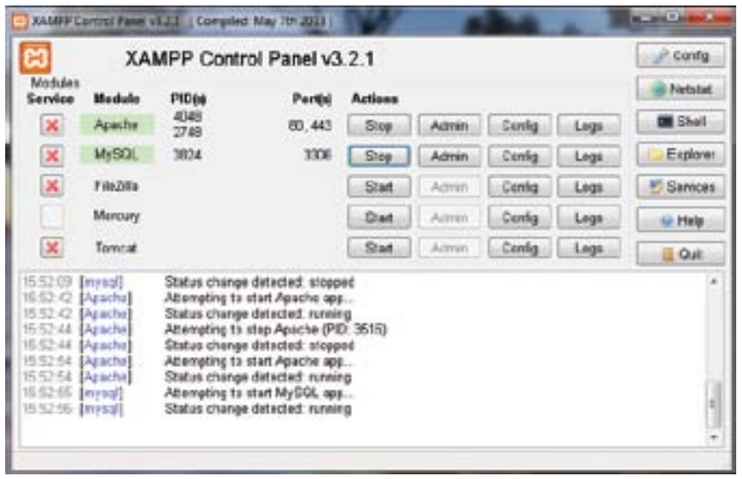
4.2. Instalacija i konfiguriranje Apache mrežnog poslužitelja i MySQL-a

4.2.1. Postavljanje Apache mrežnog poslužitelja, priprema MySQL-a i PHP-a

Ako instalirate *WordPress* lokalno, na svoj kompjuter, trebate preuzeti neki od programa koji će to omogućiti. Vjerojatno najpopularniji, a k tomu i besplatan i jednostavan je *Xampp* koji omogućava instalaciju *Apache* mrežnog poslužitelja na lokalno računalo i ujedno sadrži MySQL, PHP i *Perl*. Preuzmite *Xampp* i instalirajte ga.

Slika 4.1: Prikaz računalnog programa XAMPP

Nakon instalacije, pokrenite program. Aktivirajte module ***Apache*** i ***MySQL***.



Slika 4.2: XAMPP Control Panel

Omogućite *firewallu* pristup modulima za aktivaciju.

Trebamo preuzeti i novu verziju *WordPressa* da bismo je instalirali na računalo, a ujedno, kada napravimo mrežno adresište (*site*), da bismo je prenijeli na mrežni poslužitelj. Nakon preuzimanja, raspakirajte. Pronađite mapu u koju je instaliran *Xampp* i kopirajte mapu *WordPressa* na: **c:\xampp\htdocs**.

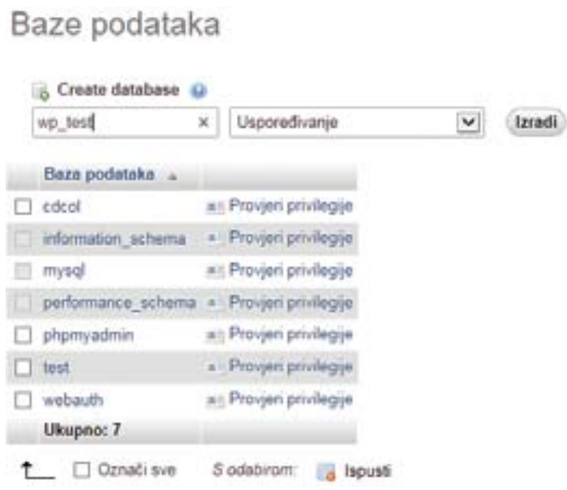
4.2.2. Kreiranje MySQL *Database WordPressa*

Vrijeme je za stvaranje naše prazne baze podataka MySQL-a kojom će se koristiti *WordPress*. U instaliranom XAMPP-u dolazi i *phpMyAdmin*.

Što je *phpMyAdmin*? *PhpMyAdmin* je besplatni softver napisan u PHP-u s ciljem podrške administriranju MySQL-a preko *World Wide Weba*. *PhpMyAdmin* podržava širok raspon operacija s MySQL-om. Najčešće operacije su podržane od strane korisničkog sučelja: upravljanje bazama podataka, tablice, polja, odnosa, indeksi, korisnici, dozvole, itd. pa imate mogućnost izravno izvršiti bilo koji SQL. Vratimo se stvaranju našeg localhosta baze podataka MySQL.
Otvorite svoj mrežni preglednik i idite na: **http://localhost/phpmyadmin/**

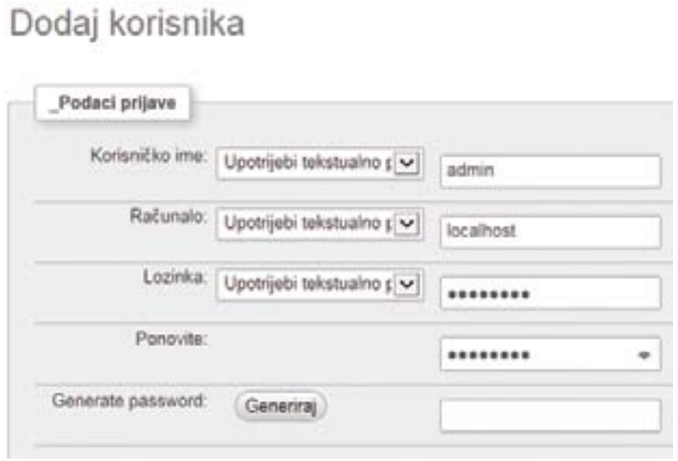
Slika 4.3: Opće postavke

U izborniku odaberite **Baze podataka** (*Databases*) i kreirajte bazu podataka, npr: **wp_test** i kliknite **Izradi**:



Slika 4.4: Baze podataka

Tako smo kreirali bazu podataka koja je trenutno, naravno, prazna. Kliknite **Provjeri privilegije** na **Dodaj korisnika**:



Slika 4.5: Dodavanje korisnika

Pod **Opće privilegije** kliknite na **Označi sve**:

Slika 4.6: Opće privilegije

Pri dnu prozora kliknite **Kreni**:

Slika 4.7: Pregled korisnika

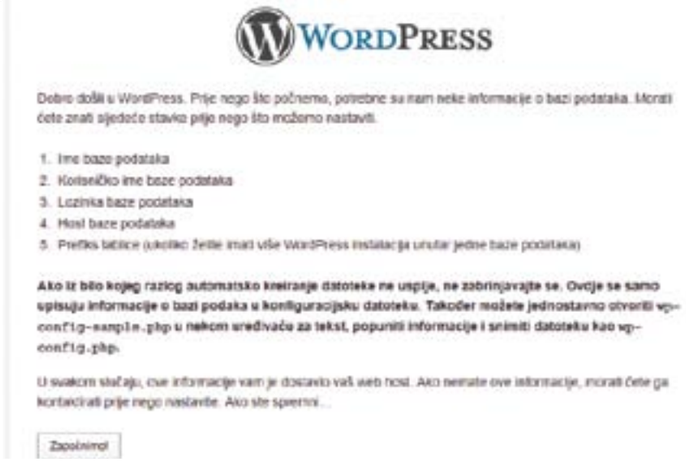
4.2.3. Instalacija i konfiguriranje postavki *WordPressa*

Time smo pripremili sve za lokalnu instalaciju *WordPressa*. U mrežni preglednik unesimo: **http://localhost/wordpress.**

Kliknite **Kreiraj Konfiguracijsku Datoteku**:

Slika 4.8: Kreiranje konfiguracijske datoteke

Kliknite **Započnimo!**



Slika 4.9: Ulaz u WordPress

Popunite tražene podatke (koje smo maloprije naveli pri dodavanju korisnika) i kliknite **Pošalji**:

Slika 4.10: Podaci o povezivanju s bazom podataka

Kliknite **Pokreni instalaciju**:

Slika 4.11: Završetak instalacije

I popunite informacijama:

Slika 4.12: Informacije o korisniku

Kliknite na **Install WordPress** i logirajte se u *WordPress* svojim korisničkim imenom i lozinkom.

4.3. Izrada teme WordPressa i rad na localhostu

4.3.1. Kreiranje datoteka

Svaka tema *WordPressa* se sastoji od nekoliko datoteka koje su temelj za izgradnju složenijeg okvira. Za bolje upravljanje, programeri kreiraju dodatne mape za *hosting* multimedijskih sadržaja i skripti kojima se koristimo za temu. Najčešća mapa je *images* koja se upotrebljava za pohranu zajedničkih ili često korištenih slika u temi. No, gdje stvoriti te datoteke unutar *WordPressa*?
Sve teme se nalaze u mapi **wp-content/themes**.
Dakle, prvo napravimo mapu na tomu mjestu. Možete dati smisleni naziv za ovu mapu koja je bliska nazivu same teme. Izradimo mapu s nazivom **moja_tema**.

Pogledajmo osnovne i bitne datoteke koje moraju biti kreirane unutar ove mape:

- **style.css** je datoteka koja sadrži kôd potreban za podešavanje izgleda teme
- **index.php** , datoteka koja sadrži kôd za pokretanje početne stranice mrežnog adresišta
- **header.php**, kao što ime upućuje, datoteka postavlja sadržaj u zaglavlju (*header* je prostor namijenjen za dodavanje slike i naziva vašeg adresišta)
- **footer.php** je datoteka uređuje izgled podnožja teme
- **sidebar.php**, *sidebar* je prostor namijenjen za dodavanje pomoćnih elemenata kao što su popis najnovijih postova, pregled kategorija postova, pregled arhive postova, reklamnih sadržaja, polja za pretraživanje i slično
- **ingle.php** je datoteka koja sadrži kôd za upravljanje sadržajem jedne poruke tijekom komunikacije
- **page.php** je još jedna bitna datoteka koja omogućava upravljanje sadržajem statične stranice kreirane od strane korisnika
- **archive.php** je datoteka koja omogućava upravljanje sadržajem arhive
- **functions.php** omogućava vama ili korisnicima dodavanje prilagođenih funkcija u temi.

Da bismo kreirali datoteke i radili s HTML kôdom, potreban nam je i kvalitetan HTML editor. Moj izbor je **Brackets**, odličan HTML editor otvorenog kôda. Možete ga preuzeti na **http://brackets.io/**.

Pokrenite *Brackets* i kreirajte sljedeće datoteke (**File** zatim **New** i dodajte nazive i ekstenziju):
style.css
index.php
header.php
footer.php
sidebar.php
single.php
page.php
archive.php
functions.php

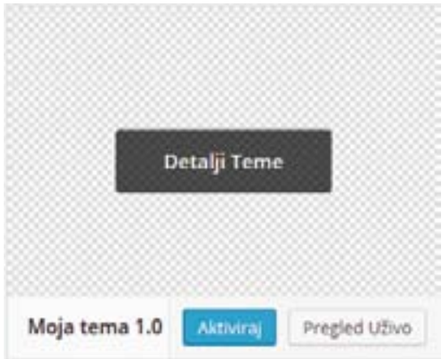
Spremite sve datoteke u mapu koju smo prethodno kreirali:
c:\xampp\htdocs\wordpress\wp-content\themes\moja_tema

4.3.2. Izrada teme Wordpressa

Nakon kreiranja potrebnih datoteka otvorite **style.css** i u editoru dodajte sljedeći komentar. To su informacije o nazivu teme, verziji, imenu autora, URI autora (*Uniform Resource Identifier*) i URI teme.

```
/*
Theme Name: Moja tema 1.0
Theme URI: http://www.inf.hol.es
Description: WordPress tema
Author: Ž.
Author URI: http://www.inf.hol.es
Version: 1.0
*/
```

Naravno, vi možete promijeniti informacije o temi. Spremite i zatvorite datoteku i idite na **Izgled** zatim **Teme** *WordPress* nadzorne ploče. Pregled sadrži *screenshot* teme, naziv teme i aktivacijski gumb. Osvježite prozor svoga mrežnog preglednika:



Slika 4.13: Informacije o temi WordPress-a

Ukoliko želite vidjeti *screenshot* svoje teme, napravite sliku veličine 600 × 450 piksela. Možete smanjiti tu veličinu i na pola (300 × 225 piksela). Spremite slikovnu datoteku pod nazivom: **screenshot.png**. Krenimo na sljedeći važan korak dodavanjem osnovnih kontejnera. Radi se o dijelovima primarnih sadržaja *headera*, sadržaja, *sidebara* i *footera*. Za stvaranje tih kontejnera, otvorite **style.css** i dodajte sljedeći kôd:

```
body{
  font-family:Arial;
  font-size:15px;
  color:#000;
  background:#fff;
}
h1 {
  font-size:50px;
}
h3 {
  font-size:22px;
}
#content-wrapper {
  width:960px;
  text-align:left;
  margin:0 auto;
  background:#fff;
  padding:15px;
}
#header{
  width:960px;
  height:120px;
}
#post-content {
  float:left;
  width:600px;
  padding:0 15px 15px 15px;
}
.sidebar{
  float:left;
  width:300px;
  margin:0 0 0 15px;
  font-size:15px;
  list-style:none;
}
#footer{
  height: 60px;
  background: #fff;
  padding: 15px;
  clear:both;
}
```

Nakon svakog dodanog kôda pogledajte kako izgleda tema. Aktivirajte temu i nakon toga kliknite na **Posjeti web-stranicu**. Ona je prazna. Ovaj kôd daje osnovni kostur mrežne stranice. Možete primijetiti neke poznate nazive kao što su *sidebar*, zaglavlje i podnožje. Sljedeći važan korak uključuje stvaranje kostura zaglavlja teme. On sadrži minimalni, najbitniji kôd za zaglavlje teme *WordPressa*. Kreirajmo predložak zaglavlja:

```
<!DOCTYPE html>
<html <?php language_attributes(); ?>>
<head>
<meta charset="<?php bloginfo( 'charset' ); ?>" />
```

```
<title><?php wp_title ( '|', true,'right' ); ?></title>
 <link rel="profile" href="http://gmpg.org/xfn/11" />
<link rel="stylesheet" type="text/css" media="all" href="<?php bloginfo( 'stylesheet_
      url' ); ?>" />
<link rel="pingback" href="<?php bloginfo( 'pingback_url' ); ?>" />
<?php
 if ( is_singular() && get_option( 'thread_comments' ) )
      wp_enqueue_script( 'comment-reply' );
 wp_head();
?>
</head>
<body>
<div id="content-wrapper">
   <div id="header">
      <h1><a href="<?php echo get_option('home'); ?>"><?php bloginfo('name'); ?></a>
       </ h1>
   </div>
```

Primijetimo da se radi o kôdu neophodnom za komentare. Ova tema će imati normalne komentare, ali ako temu modificirate i proširite da biste komentarima dodali mogućnost *threada*, ovaj kôd će omogućiti njegovu jednostavnu aktivaciju. Također možete primijetiti poziv prema funkciji **wp_head ()**;. To je jedna od bitnih funkcija koje moraju biti prisutne u datoteci zaglavlja predloška. Funkcija treba biti prisutna neposredno prije zatvaranja **</ head> taga**. Obratite pozornost i na uključivanje *content-wrapper* dijela koji će zaokružiti sve ostale u cjelinu. Dodajmo predložak početne stranice. U ovom početnom dijelu izrade mrežne stranice trebat će i malo strpljenja da biste vidjeli izgled svoje teme. Da bi to već sada bilo vidljivo, dodajte sljedeći kôd u datoteci **index.php**:

```
<?php get_header(); ?>
   <div id="post-content">
      <?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
      <div class="post">
      <h3><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h3>
        <div class="entry">
           <?php the_post_thumbnail(); ?>
           <?php the_content(); ?>
           <p class="postmetadata">
           <?php _e('Filed under&#58;'); ?> <?php the_category(', ') ?> <?php _e('by'); ?>
       <?php  the_author(); ?><br />
           <?php comments_popup_link('Nema komentara &#187;', '1 komentar &#187;',
      '% Komentara &#187;'); ?> <?php edit_post_link('Edit', ' &#124; ', ''); ?>
           </p>
        </div>
      </div>
```



```
<?php endwhile; ?>
      <div class="blog-pager">
      <?php posts_nav_link(); ?>
      </div>
      <?php endif; ?>
   </div>
<?php get_sidebar(); ?>
<?php get_footer(); ?>
```

Primijetite *post-content* u datoteci koji će omogućiti popunjenje najnovijih postova na vašem mrežnoj stranici. Možda ćete primijetiti pozive *sidebara* i *footera* koji su također uključeni u ovu datoteku. Iako ove sekcije još nisu kodirane, kôd u indeksu dovoljno je dobar da prikaže nešto na naslovnici. Spremite promjene, otvorite mrežni pretraživač, unesite **http://localhost/wordpress/wp-admin** i podatke: **Kor isničko ime** i **Lozinka** da se logirate. Osvježite stranicu.

Opaaa... dobili ste osnovni kostur teme! Ovako izgleda početni post *Wordpressa*. Važna stvar koja nedostaje je navigacijski izbornik i uobičajeno je da se postavlja odmah ispod zaglavlja. Dodajmo podršku za prilagođeni navigacijski izbornik. Koristeći se *WordPressom* 3.8.x možete jednostavno dodati podršku za stvaranje prilagođenih navigacijskih izbornika. To je moguće u dva jednostavna koraka. Najprije dodajte sljedeći kôd u datoteku **functions.php**:

```
<?php
add_theme_support( 'menus' );
```

nakon toga, dodajte sljedeći kôd na kraju datoteke **header.php**:

```
<?php wp_nav_menu( array( 'sort_column' => 'menu_order', 'container_class' =>
'menu-header' ) ); ?>
```

Nakon ovih dodataka možete u nadzornoj ploči pod **Izgled** pa **Izbornici** kreirati prilagođeni izbornik za svoju temu. Također, u nadzornoj ploči pod **Stranice** zatim **Dodaj Novu** možete dodati i nove stranice. Stavite naziv stranice i kliknite **Objavi**. No, bez oblikovanja izbornik će se pojaviti poput jednostavne poveznice. Da biste ga uredili, trebate postaviti CSS kôd. Ako pažljivo pogledate izgled kôda, možete vidjeti da je klasa povezana s izbornikom **menu-header**. Priložite sljedeći kôd na kraju datoteke **style.css**:

```
.menu-header {
   width:960px; background:#f8f8f8;
   display:block;
   float:left;
   position:relative;
}
.menu-header ul {
   list-style:none;
   margin:0;
```

```
}
.menu-header li{
   float:left;
}
.menu-header a {
background:#f8f8f8;
color:#111;
display: block;
font-size:11px;
font-weight:normal;
padding:0 20px;
line-height:38px;
text-transform:uppercase;
}
.menu-header a:hover {
background:#e6e6e6;
color:#000;
}
```

Osvježite stranicu i pogledajte promjene. Ovdje se koristimo klasom **menu-header** za stiliziranje izbornika. Time smo napravili minimalan *styling* izbornika. Također možete dodati i podizbornik u ovaj izbornik. Blogeri često žele pokazati umanjene prikaze slika, *thumbnail*, poput nekog teksta u najnovijim postovima na naslovnici. Da biste to omogućili, jednostavno dodajte dvije linije kôda odmah ispod kôda prilagođenog izbornika prethodno postavljenog u datoteci **functions.php**:

```
add_theme_support('post-thumbnails');
set_post_thumbnail_size(350, 200, true);
```

Prva linija omogućava podršku *postovima* za prikaz umanjenih sličica, *thumbnail*, dok druga određuje dimenzije same sličice. Izmjenom kôda možete lako promijeniti veličine sličica. Krenimo na sljedeći korak i dovršimo predložak stranice jednog posta da bismo mogli otvoriti i pregledati pojedinačne *postove*. Kôd koji ćemo upotrijebiti za ovu stranicu je gotovo isti kao i onaj u datoteci **index.php**. Jedina razlika je uključivanje kontejnera s komentarima. Kôd koji ćemo postaviti u datoteku **single.php** jest:

```
<?php get_header(); ?>
  <div id="post-content">
    <?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
    <div class="post">
    <h3><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h3>
      <div class="entry">
        <?php the_post_thumbnail(); ?>
        <?php the_content(); ?>
        <p class="postmetadata">
```



```
        <?php _e('Filed under&#58;'); ?> <?php the_category(', ') ?> <?php _e('by'); ?>
      <?php  the_author(); ?><br />
        <?php comments_popup_link('Nema komentara &#187;', '1 Komentar &#187;',
      '% Komentara &#187;'); ?> <?php edit_post_link('Edit', ' &#124; ', ''); ?>
        </p>
      </div>
      <!-- Komentar  -->
      <div class="comments-wrapper">
        <?php comments_template(); ?>
      </div>
  </div>
<?php endwhile; ?>
   <!—Navigacijske poveznice za postove-->
  <div class="blog-pager">
    <?php previous_post_link('< %link') ?> <?php next_post_link(' %link >') ?>
  </div>
<?php endif; ?>
</div>
<?php get_sidebar(); ?>
<?php get_footer(); ?>
```

Kao što vidite, poglavlje s komentarima uključeno je odmah nakon *postova*. Klasa povezana s ovim poglavljem je *comments-wrapper*, koji se može upotrijebiti za njegovo stiliziranje.

Kako kreirati rubni stupac (*sidebar*) i predložak podnožja (*footera*)? Rubni stupac je potrebno urediti u svrhu prihvaćanja *widgeta*. *WordPress widgeti* dodaju dodatni sadržaj i značajke rubnom stupcu. Nakon početne instalacije, *Wordpress* sadržava *widge* kao što su kategorije *postova*, navigacija, pretraživanje, itd. Pripremimo rubni stupac za prihvat *widgeta*. To možemo učiniti vrlo jednostavno dodajući sljedeći kôd u datoteku **functions.php**:

```
if ( function_exists('register_sidebar') )
   register_sidebar( array (
   'name' => 'Widget Area',
   'id' => 'widget_area',
   'before_widget' => '<li id="%1$s" class="widget-section %2$s">',
   'after_widget' => "</li>",
   'before_title' => '<h3 class="widget-title">',
   'after_title' => '</h3>',
  ) );
```

Pogledajte sada pod **Izgled » Widgeti** i vidjet ćete prazan kontejner spreman za uporabu. Možete dodati različite *widgete*, ali neće biti vidljivi jer je **sidebar.php** datoteka još uvijek prazna. No, prije nego u **sidebar.php** dodamo odgovarajući kôd, moramo prilagoditi funkciju za povla-

čenje statičnih *widgeta*, naravno, ukoliko ih ima. Ova prilagođena funkcija vraća povratne informacije o kontejneru pripremljenom za njihovo prihvaćanje. Slijedi kôd koji mora biti dodan u datoteku **functions.php**:

```
function is_sidebar_active( $count ){
 global $wp_registered_sidebars;
 $widgetcols = wp_get_sidebars_widgets();
 if ($widgetcols[$count]) return true;
  return false;}
```

Primijenit ćemo prilagođenu funkciju u predlošku našeg *sidebara* da bismo dohvatili informacije koje smo prethodno kreirali. Nakon toga slobodno dodajte *widgete*.

Evo predloška za datoteku **sidebar.php**:

```
<div class="sidebar">
<?php if ( is_sidebar_active('widget_area') ) : ?>
      <div id="primary" class="widget-area">
        <ul class="sidebar-list">
          <?php dynamic_sidebar('widget_area'); ?>
        </ul>
      </div>
<?php endif; ?>
</div>
```

Možete primijetiti da smo ovaj kôd upakirali u klasu *sidebar* koju smo deklarirali u datoteci **style.css**. Osim toga, upotrijebili smo tekst klase *widget* za prepoznavanje i aktiviranje ovog kontejnera jer to je isti ID koji smo upotrijebili prilikom izrade kontejnera u datoteci **functions.php**.

Kreirajmo i predložak podnožja za dovršetak osnovnog izgleda mrežne stranice. Slijedi kôd koji mora biti dodan u datoteku **footer.php**:

```
<div id="footer">
  <p>
  <strong>Copyright 2013 <?php bloginfo('name'); ?> All Rights Reserved.</strong> |
    Created by <a href="http://www.inf.hol.es">Informatika</a> |
    <a href="<?php bloginfo('rss2_url'); ?>">Posts RSS Feed</a> |
    Powered by <a href="http://wordpress.org/">WordPress</a></p>
  </div>
</div>
</body>
</html>
```


Osvježite mrežnu stranicu. Možete promijeniti kôd prema vašim zahtjevima. Primijetite da smo upotrijebili ID podnožja koji je u početku bilo deklariran u datoteci *style.css*. Kroz ovaj ID možete lako promijeniti izgled ovog kontejnera da bi poprimio ljepši i profesionalniji izgled. Posljednje dvije važne datoteke na koje treba obratiti pozornost su predlošci statične stranice i arhivske stranice. Krenimo s izradom osnovnog predloška za prikaz statične stranice kreirane od strane korisnika.

Evo kôda koji bi trebao biti dodan u datoteku **page.php**:

```
<?php get_header(); ?>
  <div id="post-content">
    <?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
    <div class="post">
    <h3><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h3>
      <div class="entry">
      <?php the_content(); ?>
        <p class="postmetadata">
          <?php _e('Filed under&#58;'); ?> <?php the_category(', ') ?> <?php _e('by'); ?>
     <?php  the_author(); ?><br />
     <?php comments_popup_link('No Comments &#187;', '1 Comment &#187;', '%
     Comments &#187;'); ?> <?php edit_post_link('Edit', ' &#124; ', ''); ?>
        </p>
      </div>
    </div>
<?php endwhile; ?>
<?php endif; ?>
</div>
<?php get_sidebar(); ?>
<?php get_footer(); ?>
```

Primjećujete da je kôd za **page.php** gotovo isti kao i za datoteku **single.php**. Postoje dvije male razlike: nedostaje kôd za sekciju komentara kojim se obično ne koristimo na statičkim stranicama, kao i kôd za navigaciju koji vodi na prethodnu ili sljedeću stranicu. Ako ih želite u svoj predložak, možete jednostavno dodati i te dvije sekcije. Prijeđimo na posljednju važnu datoteku koja će odrediti strukturu arhivske stranice. Uključimo sljedeći kôd u datoteku **archive.php**:

```
<?php get_header(); ?>
  <div id="post-content">
    <?php if(have_posts()) : ?><?php while(have_posts()) : the_post(); ?>
    <div class="post">
    <h3><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h3>
      <div class="entry">
      <?php the_excerpt(); ?>
        <p class="postmetadata">
          <?php _e('Filed under&#58;'); ?> <?php the_category(', ') ?> <?php _e('by'); ?>
```

```
<?php  the_author(); ?><br />
                <?php comments_popup_link('Nema komentara &#187;', '1 komentar &#187;',
'% Komentara &#187;'); ?> <?php edit_post_link('Edit', ' &#124; ', ''); ?>
              </p>
            </div>
          </div>
    <?php endwhile; ?>
      <div class="blog-pager">
        <?php posts_nav_link(); ?>
      </div>
    <?php endif; ?>
    </div>
    <?php get_sidebar(); ?>
    <?php get_footer(); ?>
```

Primijetite uključenu navigaciju u ovom predlošku koja prolazi kroz sve arhivirane stranice osim komentara koji nisu neophodni. Ovime smo napravili osnovnu temu WordPressa. Svrha je poznavati neophodne postupke potrebne za izradu teme od samih početaka, a ujedno vam može pomoći u kreiranju vlastite teme *WordPressa*.

4.3.3. Uređivanje teme na lokalnom hostu

Prije objave mrežne stranice na WWW-u, svakako bih vam preporučio da na *localhostu* dobro uvježbate:

- rad s medijskom zbirkom i dodavanje novih
- dodavanje i objavljivanje stranica:
    - rad s tekstom
    - medijskom zbirkom
- aktivaciju teme, prilagođavanje, uključivanje *widgeta*
- rad s izbornicima, uključivanje stranica i poveznica (važno pri radu s padajućim izbornicima)
- rad s dodacima, odabir dodataka , instalacija i aktivacija
- dodavanje i uključivanje korisnika
- alate za uvoz i izvoz (više o tome u sljedećem poglavlju)
- podešavanje općih postavki:
    - naziv mrežne stranice
    - adresa *WordPressa* (URL)
    - adresa mrežne stranice (URL)
    - vremenska zona
    - oblik datuma i vremena
    - opcije za podešavanje čitanja, pisanja i rasprave.

Uz malo uređivanja ova tema može poprimiti zanimljiv i privlačan izgled, a preuzeti ju možete na: **www.e-skola.com.hr**.

4.4. WordPress online

4.4.1. Kreiranje baze podataka WordPress MySQL

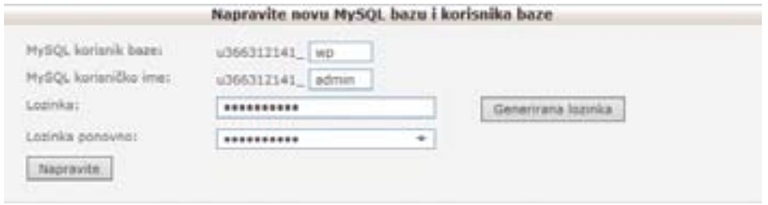
Ako ste preskočili prethodna poglavlja i želite raditi *online*, najprije preuzmite *WordPress* na poveznici: **http://hr.wordpress.org/** i raspakirajte instalacijski paket.

Kreiranje baze podataka i korisnika

Napravite bazu podataka za *WordPress* na mrežnom poslužitelju, kao i korisnika MySQL koji ima ovlasti pristupa i uređivanja baze rabeći *cPanel* ili *phpMyAdmin*. Prijavite se u svoj **cPanel** i kliknite na **MySQL baze**:

Slika 4.14: cPanel - napredne postavke

Upišite korisničko MySQL korisnika, MySQL korisničko ime i lozinku. Kliknite **Napravite**.



Slika 4.15: Kreiranje MySQL korisnika i baze

Zapišite naziv baze MySQL, korisnika MySQL i naziv *hosta* MySQL.

4.4.2. Instalacija WordPressa

4.4.2.1. Kopiranje datoteka na mrežni poslužitelj

Sada je potrebno kopirati cijeli sadržaj mape **wordpress** na mrežni poslužitelj. Upotrijebite neki FTP program poput *FileZilla* i slično. Osobno rabim WinSCP koji možete preuzeti na poveznici:

http://winscp.net/eng/download.php

Nakon instalacije kliknite **New Site**, odaberite protokol **FTP**, popunite podatke: **Host name**, **User name** i **Password**.

Sada se morate odlučiti kamo na mrežnom poslužitelju želite smjestiti *WordPress*:

- U *root*-direktoriju svog *sitea* (npr. http://primjer.com/)
- U poddirektoriju svog *sitea* (npr. http://primjer.com/blog/)

Napomena: Lokacija vašeg mrežnog *root*-direktorija na vašem mrežnom poslužitelju može se razlikovati od jednog do drugog pružatelja usluga i operacijskih sistema. Provjerite kod svog pružatelja usluga ili sistemskog administratora, ako ne znate gdje se nalazi.

***Root*-direktorij**

Ako trebate preslikati datoteke na mrežni poslužitelj, upotrijebite FTP klijent da biste preslikali sav sadržaj direktorija **wordpress** (ali ne i sam direktorij **wordpress**) u *root*-direktorij svog mrežnog *sitea*.

Poddirektorij

Ako trebate kopirati datoteke na mrežni poslužitelj, preimenujte direktorij **wordpress** u željeni naziv, a zatim s FTP-klijentom kopirajte cijeli direktorij na željenu lokaciju u *root*-direktorij svog mrežnog *sitea*. Ukoliko rabite WinSCP, jednostavno označite sav sadržaj iz mape **wordpress** s lijeve strane i povucite udesno u **public_html** ili kako je kod vas predviđeno mjesto za udomljavanje mrežnog *sitea*.

4.4.2.2. Konfiguriranje postavki

Napomena: ukoliko imate *hosting* koji pruža automatsku instalaciju *WordPressa*, preskočite sljedeće radnje i prijeđite na poglavlje **4.4.2.3. Pokretanje instalacijske skripte**.

Pomoću FTP klijenta preslikajte datoteku wp-**config-sample.php** na računalo i promijenite naziv datoteke u **wp-config.php**. Otvorite datoteku u nekom uređivaču teksta, npr. **Notepad**. Unesite informacije o bazi podataka ispod naznačene sekcije:

// ** MySQL postavke - informacije o postavkama možete dobiti od svog mrežnog hosta ** //

/** Ime vaše baze podataka za WordPress */

define('DB_NAME', 'ovdje_upisite_ime_baze');
/** MySQL korisničko ime baze podataka */
define('DB_USER', 'ovdje_upisite_korisnicko_ime');

/** MySQL lozinka baze podataka */
define('DB_PASSWORD', 'ovdje_upisite_lozinku');

/** MySQL naziv hosta */
define('DB_HOST', 'localhost');

/** Kodiranje znakova koje će se koristiti u kreiranju tabela baze podataka. */
define('DB_CHARSET', 'utf8');

/** Collate tip baze podataka. Ne mijenjate ako ne znate što radite. */
define('DB_COLLATE', '');

Snimite **wp-config.php** datoteku i kopirajte ju nazad u direktorij *WordPressa*.

4.4.2.3. Pokretanje instalacijske skripte

Da biste započeli instalaciju, usmjerite svoj mrežni preglednik na instalacijsku skriptu. Ako ste smjestili datoteke *WordPress* u *root*-direktorij, onda posjetite **http://primjer.com/wp-admin/install.php**. Ako ste smjestili datoteke *WordPress* u poddirektorij nazvan, na primjer **blog**, onda posjetite **http://primjer.com/blog/wp-admin/install.php**.

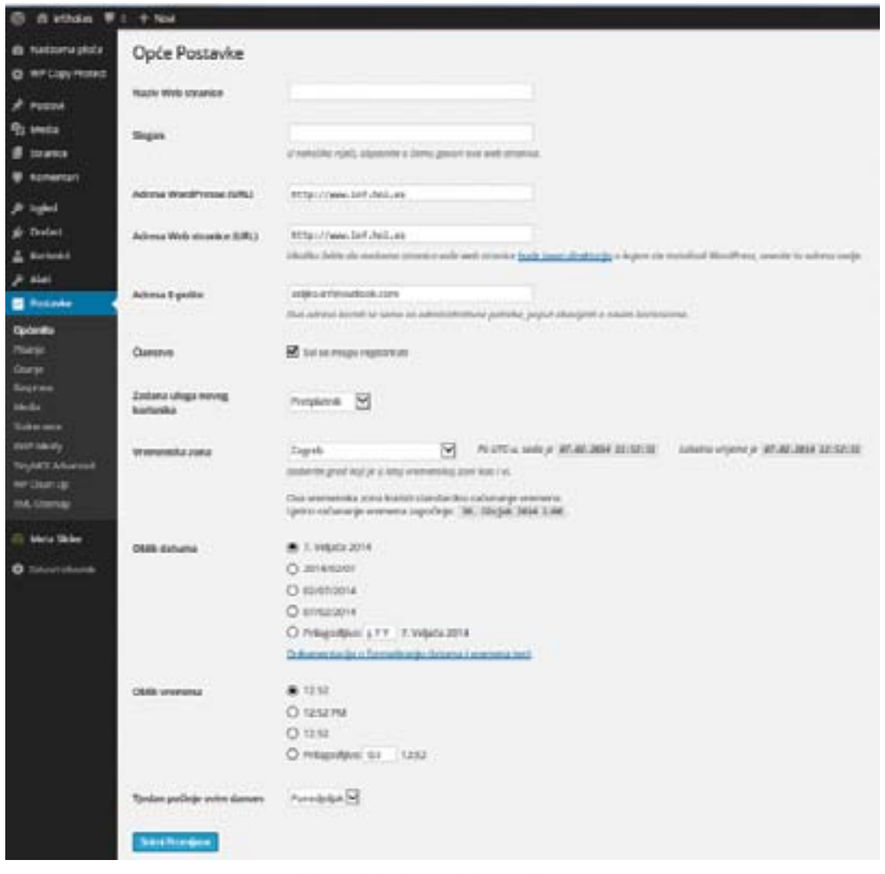
Ako *WordPress* ne može pronaći **wp-config.php** datoteku, prijavit će vam to i ponuditi da sam kreira i uredi datoteku. Slijedite upute. *WordPress* će vas upitati ime baze podataka, korisničko ime, lozinku i *host* baze podataka i zapisati ih u novu **wp-config.php** datoteku. Popunite tražene podatke i kliknite **Pošalji**. Ako ovaj dio instalacije ne prođe dobro, vratite se na poglavlje **4.4.2.2. Konfiguriranje postavki** i sami podesite datoteku **wp-config.php**.

Instalacija će vas sad odvesti na ekran na kojem morate unijeti osnovne informacije o svojoj stranici. Trebate unijeti naziv svoje mrežne stranice, svoju e-adresu kao i želite li da se vaša mrežna stranica pojavljuje na mrežnim tražilicama kao što su Google, Bing, Yahoo, itd. Pobrinite se da ste upisali ispravnu e-adresu jer ćete na nju dobiti podatke za pristup vašoj instalaciji *WordPressa* kao i sve ostale obavijesti. Na ovoj stranici također možete izabrati svoje korisničko ime i lozinku. Dobro razmislite o korisničkom imenu jer kasnije sve možete promijeniti, osim korisničkog imena. Prilikom instalacije WP-a nudi se izbor naziva administratorskog korisnika koji će imati pristup svim stavkama aplikacije kojom ćete se koristiti. Mnoge skripte automatski pokušavaju napasti **nazivdomene/wp-admin** pristupno sučelje uporabom standardnih (*admin*, *Admin*, administrator, Administrator, *root* ili *Root*) korisničkih imena pa je pametno i preporučljivo upotrijebiti neki alternativni naziv za glavni *login*.

Nakon što unesete sve potrebne podatke kliknite gumb **Instaliraj WordPress**, i time će se pokrenuti završni dio instalacije. Ukoliko se pojavi pogreška provjerite svoju **wp-config.php** datoteku i podatke koje ste unijeli u prethodnom koraku i zatim pokušajte ponovno. Nakon završetka instalacije pojavit će se ekran s podacima za prijavu. Kliknite na gumb **Prijava** za prijavu u upravljački dio *WordPressa* i prijavite se svojim korisničkim imenom i lozinkom. Nakon što se uspješno prijavite, otvorit će se **Nadzorna ploča** odakle možete upravljati svojom instalacijom *WordPressa*. Instalacija *WordPressa* je time gotova.

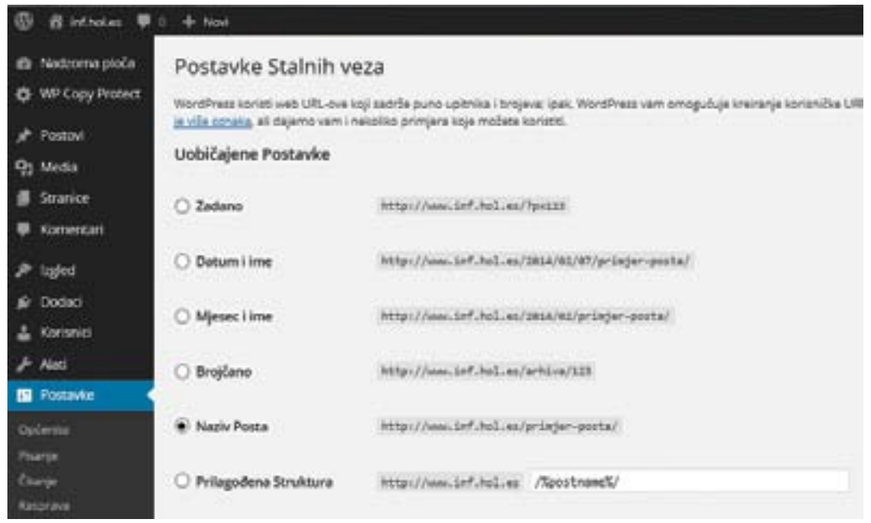
4.4.3. Podešavanje osnovnih postavki WordPressa

Podešavanjem osnovnih postavki *WordPressa* ujedno ćemo učiti sve prednosti *WordPressa* u odnosu na ostale CMS-ove (*Content Management System*). Pogledajmo na koji ćemo način izvršiti inicijalno podešavanje te instalaciju dodataka (*pluginova*) i tema. Prijavite se u WP admin sučelje na adresi **http://primjer.com/wp-admin/**. Unesite korisničko ime i lozinku koje ste definirali tijekom instalacije. Ukoliko smatrate da niste postavili dovoljno jaku i pouzdanu lozinku tijekom instalacije za glavnog *admin usera*, preporučujem da odmah nakon prve prijave u WP *admin* pod **Nadzorna ploča** zatim **korisnici** te **Vaš profil** i izmijenite lozinku.



Slika 4.16: Nadzorna ploča WordPress-a

Pod **Postavke** zatim **Općenito** podesite i osnovne parametre novonastale mrežne stranice (naziv, adresa, vremenska zona, oblik nadnevka i vremena). Pripazite na prefikse **www** pod **Adresa WordPressa (URL)** te **Adresa Web-stranice (URL)** po njima će na Googleu biti indeksirane sve stranice. Podesite izgled linkova pod **Postavke** i **Stalne veze**.



Slika 4.17: Postavke stalnih veza

Za definiranje stila izgleda mrežne stranice kliknite na **Izgled** pa na **Teme**. Klikom na **Dodaj novu temu** pronađite temu koja vam odgovara za izgled mrežne stranice. Instalirajte, aktivirajte i prilagodite postavke. Nakon toga kliknite na **Posjeti web-stranicu** da biste vidjeli novi izgled mrežne stranice.

Za kreiranje vlastite teme potražite više informacija na službenoj stranici *WordPress Codex*.Nakon postavljanja izgleda mrežne stranice, pridodat ćemo više funkcionalnosti samom *WordPressu* s dodacima (*pluginovima*). Kliknite na **Dodaci** zatim na **Dodaj novi**.

Jedna od prednosti *WordPressa* nad ostalim CMS-ovima upravo je velik broj dodataka koji proširuju njegovu funkcionalnost.

Predlažem vam da, između ostalih, odaberete dodatke koji će:

- minimalizirati CSS i JS datoteke za brže učitavanje stranica
- generirati posebni XML *sitemap* koji će pomoći tražilicama poput Googlea, Yahooa, Binga, itd. da bolje indeksiraju vaše stranice
- omogućiti uporabu vizualnih uređivača u *WordPressu* i dodati napredne značajke
- omogućiti uvoz *postova*, stranica, komentara, korisnička polja, kategorija, oznaka i svega ostaloga iz *WordPressove* izvozne datoteke
- očistiti i optimizirati bazu podataka, bez potrebe *phpMyAdmina*.

Redovito ažurirajte podatke samog *WordPressa* i njegovih dodataka i tema. U *WordPressu* je to zaista jednostavno jer će vas sam upozoriti o postojanju novih ažuriranja.

4.4.4. Sigurnosne postavke WordPressa

WordPress je zasigurno najpopularniji od postojećih CMS (*Content Management System*) sustava na kojem se, prema procjenama, „vrti" svaki šesti mrežni *site*. Upravo je zbog toga WP izrazito privlačna meta hakerima.

Iskorištavanjem sigurnosnih propusta same aplikacije i *pluginova* kojima se koriste na istoj, oni pokušavaju (nerijetko i uspijevaju) na poslužitelje postaviti nepoželjan sadržaj. Tim sadržajem su u mogućnosti pokretati DDOS napade, slati velike količine *spama* ili uzrokovati manje probleme koji utječu na rad poslužitelja na kojemu se nalazi kompromitirana mrežna stranica ili mrežna stranica drugih poslužitelja koje će ciljati njihov napad.

No, gotovo sve sigurnosne rupe su uzrokovane „lošom praksom" prilikom početne instalacije WP-a ili kasnijim izostankom redovitog održavanja aplikacije. Naime, katkada primimo upite oblika: „Zašto aplikacija sada ne radi, a radila je zadnjih nekoliko godina?" To pitanje u sebi sadrži odgovor: „ Upravo zato što se rabi ista, nenadograđena aplikacija nekoliko godina." Ako aplikaciju ne nadograđujemo samo nekoliko mjeseci, ona postaje nesigurna. Slikovito rečeno, kao da na ulaz svoga doma stavljamo natpis: "Slobodan ulaz, nema nikoga doma!". Hakeri često ostave poruku s potpisom: „Hej, zaboravili ste zatvoriti vrata!" Obrišu sve važne dokumente, postave maliciozne skripte na poslužitelj preko kojih se omogućuje pokretanje raznih napada, rušenje stabilnosti poslužitelja i slično.

Upotreba Admin-korisnika

Prilikom instalacije WP-a nudi se izbor naziva administratorskog korisnika koji će imati pristup svim stavkama aplikacije koju ćete rabiti. Mnoge skripte automatski pokušavaju napasti **nazivdomene/wp-admin** pristupno sučelje uporabom standardnih (*admin*, *Admin*, administrator, Administrator, *root* ili *Root*) korisničkih imena pa je preporučljivo koristiti se nekim alternativnim nazivom za glavni *login*.

Upotreba kompleksne lozinke

Standardna sigurnosna preporuka je upotreba kombinacije velikih i malih slova, brojeva i posebnih znakova, duljine od 10 znakova i više da biste otežali posao automatiziranim invazivnim skriptama. Najkorištenijih pet automatskih pokušaja probijanja lozinki upotrijebljenih u masovnim napadima su bili: *admin*, *123456*, *111111*, *666666*, *i 12345678*. Iako su zbog jednostavnosti ove (i slične) lozinke vrlo privlačne, također se njima otvara nepotrebna vrlo iskoristiva sigurnosna rupa kojom ćete uzrokovati probleme. Zato vrijedi truda upamtiti (ili zapisati) složeniju lozinku oblika *4vXlZn3!2$*.

Promjena WP *nicknamea*

Automatizirane skripte će često pregledati sve postove objavljene na vašoj mrežnoj stranici u potrazi za oznakama, odnosno nazivima autora te pokušati upotrijebiti navedene nazive za pristup administraciji stranice. Da biste izbjegli problem, u WP administraciji, pod **Profile** ili **Users** dodajte **Nickname** i za vrijednost **Display name publicly as** odaberite različitu vrijednost od one koju rabite za pristup.

Promjena prefiksa instalacije u bazi

Prilikom nove instalacije WP-a sama aplikacija će vam ponuditi prefiks u bazi podataka ***wp_***, koji je preporučljivo promijeniti da biste otežali posao skripti koja će eventualno pokušati provesti napade na samu bazu podataka.

Ograničavanje korištenja raznih *pluginova*, dodataka i tema

Osim podizanja razine nesigurnosti sustava, upotreba mnogih dodataka i tema u vašoj aplikaciji može uzrokovati i sporiji rad same mrežne stranice. Ograničite upotrebu *pluginova* samo na one koje zaista rabite te obrišite sve nekorištene dodatke i teme da biste smanjili mogućnost upada na stranicu. Također je važno instalirati sve najnovije nadogradnje za navedene dijelove aplikacije da biste održali razinu sigurnosti. Ako se neki od *pluginova* prestane nadograđivati ili postane nekompatibilan s novijim verzijama WP-a, svakako ga obrišite i pronađite zamjenu. Ako dođe do e-upada, održavanje „čistog" sustava (brisanjem svih nekorištenih komponenti) omogućit će vam lakše i bezbolnije dijagnosticiranje i otklanjanje problema.

Uklanjanje informacija o verziji aplikacije

Uklanjanje informacija o tome koju verziju aplikacije rabite je mali, ali ipak koristan korak u otežavanju posla automatiziranim skriptama koje se rabe za napade na WP. Da biste to učinili, u **functions.php** datoteci temi koju rabite dodajte:

```
// uklanjanje informacija o verziji
function complete_version_removal() {
 return ''; }
add_filter('the_generator', 'complete_version_removal');
```

Onemogućavanje registracije novih korisnika

Ako vodite *blog* ili mrežnu stranicu u kojoj nije predviđeno registriranje više korisnika, osim samog administratora, unutar administracije **General settings** onemogućite registraciju novih korisnika **Membership**, isključite **Anyone can register**. Također, da biste izbjegli daljnje moguće automatizirane napade, obrišite datoteku **wp-register.php** unutar vaše aplikacije ili ju preimenujte ako se pokaže potreba za daljnjom uporabom.

Zaštita *wp-config.php* datoteke

Da biste onemogućili pristup datoteci **wp-config.php** neodobrenim korisnicima, u *.htaccess* datoteku svoje aplikacije možete dodati sljedeći kod:

```
<Files "wp-config.php">
 order allow,deny
 deny from all
</Files>
```

Na taj način datoteka neće biti dostupna ni na koji način, osim putem *ftp-a* ili kroz *cPanel* administraciju. Drugi način zaštite je datoteku prenijeti u *parent* direktorij (ako ste instalirali WP unutar */public_html/* direktorija, tada prebacite **wp-config.php** u *home* direktorij). Također promijenite prava čitanja/pisanja po datoteci na 0600 da biste onemogućili ubacivanje izmjena.

Zabrana pristupa *include-only* datotekama

Uvedite dodatnu mjeru sigurnosti putem *.htaccess* datoteke te onemogućite pristup putem mrežnog preglednika dijelu aplikacije kojemu biste samo vi trebali imati pristup kroz administraciju. Dodajte sljedeće linije:

```
RewriteEngine On
RewriteBase /
RewriteRule ^wp-admin/includes/ - [F,L]
RewriteRule !^wp-includes/ - [S=3]
RewriteRule ^wp-includes/[^/]+\.php$ - [F,L]
RewriteRule ^wp-includes/js/tinymce/langs/.+\.php - [F,L]
RewriteRule ^wp-includes/theme-compat/ - [F,L]
```

Na ovaj način će svaki pokušaj pristupa standardnim *folderima* u svakoj WP instalaciji biti onemogućen vanjskim stranama.

Omogućavanje SSL prijave

Ako na svojoj mrežnojstranici rabite SSL certifikat, dodavanjem sljedećih linija u **wp-config.php** datoteku omogućit ćete slanje informacija o pristupnim podacima putem zaštićene veze:

```
//SSL za prijavu obicnih korisnika
define('FORCE_SSL_LOGIN', true);

//SSL za prijavu administratora
define('FORCE_SSL_ADMIN', true);
```

Brisanje readme i drugih nepotrebnih datoteka

WP i mnogi *pluginovi* se koriste datotekama *readme.html* u kojima se nalaze informacije o verzijama i drugi podaci za koje nema potrebe da su javno dostupni pa ih je dobro obrisati s poslužitelja. Također je uputno brisati sve datoteke za koje ste sigurni da ih niste sami dodali na poslužitelj.

Ostali savjeti za osnove sigurnosti korištenja WP-a:

Držite WP i pluginove na zadnjoj verziji

Držanje WP-a i *pluginova* na zadnjim verzijama je jedan od neophodnih postupaka održavanja sigurne aplikacije, a u najvećem broju slučajeva samo morate kliknuti **Update** gumb. Naime, budući da je WP besplatan, CMS javno je dostupan svim zainteresiranim stranama. Znači da i osobe s lošim namjerama, starenjem određene verzije CMS-a, imaju vremena pronaći i iskoristiti sigurnosne propuste u istom. Ako *plugin* kojim se koristite, razvojni tim prestane razvijati, pokušajte mu pronaći zamjenu. Budući da broj *pluginova* kreiranih za WP svakodnevno raste, gotovo je sigurno da ćete pronaći neki iste (ili bolje implementirane) funkcionalnosti koji će i dalje biti siguran.

Brinite se o lokalnoj sigurnosti

Rabite antivirusnu zaštitu na računalu s kojega se spajate na administraciju stranice i nemojte istoj pristupati s neprovjerenih računala. *Keyloggeri* (programi za praćenje utipkanih podataka na računalu) su jedan od najčešćih uzročnika proboja lozinki.

Preuzimajte *pluginove* i teme sa sigurnih lokacija

Tražite *pluginove* i teme preko same tražilice u WP administraciji ili putem službenih stranica. Većina stranica na koje naiđete potragom besplatnih dodataka za WP preko raznih tražilica je zaražena nekom vrstom *malwarea* te je samo pitanje vremena kad ćete naići na probleme.

Naravno, ovi koraci nisu svi koje možete poduzeti da biste bili sigurni. Koje radnje vi poduzimate da bi vaša aplikacija bila sigurna?

4.4.5. Arhiviranje podataka

Redovito arhivirajte (*backup*) datoteke i baze putem dodatka ili ručno. Svakako periodički snimajte lokalno, na sigurnu lokaciju, *backupe* cijelog *root foldera* aplikacije i same baze podataka. Način arhiviranja i vraćanja je vrlo jednostavan i bez uporabe posebnih dodataka od onih koje već postoje u samom *WordPressu*. Pokazat ću jednostavan način ručnog arhiviranja podataka bez uporabe dodataka osim onih koje se nalaze u samom *WordPressu*.

Arhiviranje (backup) podataka

Pristupite cPanel-u, MySQL baze:

Slika 4.18: Napredne postavke cPanel-a

Kliknite na **MySQL baze** i u listi trenutnih MySQL baza i korisnika kliknite **Kopija podataka**. Slijedite upute, obično u donjem desnom uglu možete vidjeti obavijest da su napravljene kopije baze podataka. Kliknite na **Pogledaj rezultate**, zatim na **Odaberite backup dostupan za download** i **Preuzimanje**. Spremite datoteku. Arhivirali ste MySQL bazu podataka.

Logirajte se u *WordPress*, u nadzornoj ploči pod **Alati** odaberite **Izvoz**. Odaberite **Sav sadržaj** za izvoz, kliknite na **Preuzmi Izvoznu Datoteku**. Vaša *.xml* datoteka je spremljena.

To je sve što je potrebno učiniti za arhiviranje podataka. Izvezena *.xml* datoteka sadrži sve vaše *postove*, stranice, komentare, prilagodljiva polja, pojmove, navigacijske izbornike, prilagođene (*custom*) *postove* i poveznice prema mutimedijskim dodacima sadržanima u bazi podataka.

Vraćanje (*restore*) podataka

Prilikom vraćanja podataka, bilo zbog promjene domene ili premještanja, pratite sljedeće korake:

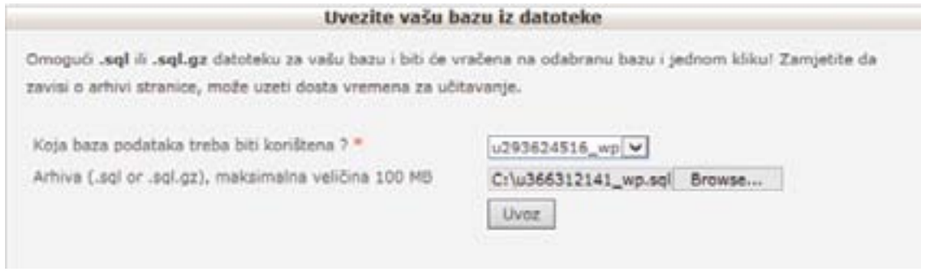
Napomena: točno pratite način kojim je opisan postupak vraćanja podataka!

1. Napravite čistu instalaciju *WordPressa*.

2. Pristupite cPanel-u, MySQL baze i odaberite **Uvezite bazu**:

Slika 4.19: cPanel - Web stranica

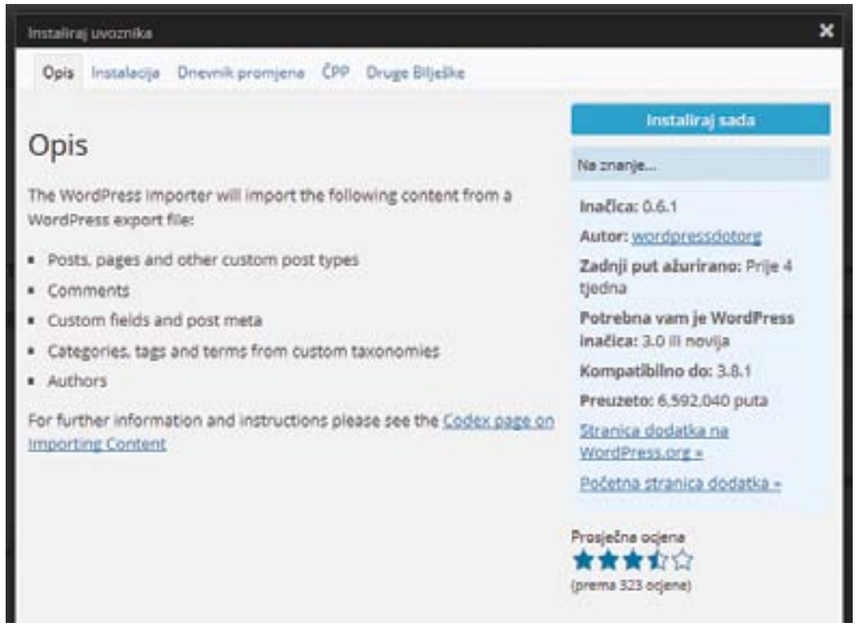
Uvezite svoju bazu:



Slika 4.20: Uvoz baze podataka

3. U ovom, važnom koraku *logirajte* se u *WordPress* i u nadzornoj ploči uvezite i aktivirajte sve dodatke koje ste prethodno rabili. To je naročito važno kako biste kod uvoza *.xml-datoteke* imali aktivirane poveznice prema multimedijskim dodacima.

4. U nadzornoj ploči *WordPressa* pritisnite na **Alati** pa na **Uvoz**. Kliknite na **Uvoz** i odaberite **WordPress**. Instalirajte uvoznika:



Slika 4.21: Nadzorna ploča - instalacija uvoznika

Kliknite na **Aktiviraj Dodatak & Pokreni Uvoznika**. Odaberite *.xml-datoteku* sa svog računala i kliknite **Prenesi datoteku i uvezi**. U sljedećem koraku odaberite **Download and import file attachments** i kliknite **Submit**. U nadzornoj ploči odaberite **Izgled** zatim **Izbornici** i u postavkama izbornika uključite **Gornji primarni izbornik** ili kako je već podešeno, ovisno o temi koju upotrebljavate. Snimite izbornik. Posjetite mrežnu stranicu. Time je ovaj jednostavan postupak vraćanja podataka završen.

Citirat ću s *wordpress.org*:
„Čestitamo! A sada učinite nešto lijepo za sebe! :)"

5. poglavlje

PHP - programski jezik

dodatni sadržaj

5.0 PHP

5.1 Uvod

PHP programski jezik najšire je prihvaćen *open-source* jezik za izradu web aplikacija, često korišten u kombinaciji s MySQL jezikom (jezik baza podataka). Zbog dugovječnosti i rasprostranjenosti PHP-a, danas postoji veliki broj sustava za upravljanje sadržajem (CMS) koji su u njemu napisani (npr. WordPress i Joomla), kao i cijela paleta pratećih pomoćnih aplikacija (npr. phpMyAdmin za rad sa MySQL bazom podataka). Stoga smo ovo poglavlje odlučili posvetiti PHP-u i napraviti mali uvod u učenje tog jezika. Dodatni motiv za obradu PHP-a bila nam je i činjenica da se u priručniku obrađuje programski jezik Logo te sustav za upravljanje sadržajem, pa se PHP činio kao logična nadopuna tim poglavljima. Željeli smo nastavnicima pružiti materijale koji će im olakšati izvođenje izborne nastave ili poslužiti kao podloga za provođenje dodatne nastave. U ovom poglavlju obradit ćemo samo osnove PHP-a, kako bi učenici usvojili pojmove u vezi s programskim jezicima i način razmišljanja potreban za programiranje jer oni čine pretpostavku za daljnje učenje skriptnih i programskih jezika.

Slika 5.1: Zaštitni znak jezika

PHP se brzo uči zbog jednostavne strukture i sintakse, a pritom daje jako bogate razvojne mogućnosti zbog velikog broja gotovih funkcija i mogućnosti proširivanja. Pored jednostavne strukture i sintakse, PHP je i objektno orijentiran jezik s punom podrškom za objektno programiranje, što ga čini vrlo fleksibilnim alatom za razvoj netipičnih web rješenja, kao što su interne stranice tvrtki, društvene usluge, razni administrativni sustavi, forumi itd. Dakle, forumi, društvene usluge i mreže, poput Facebook-a, temeljeni su međuostalim i na PHP-u. Objektna orijentiranost jezik znači i lakše preslikavanje i upravljanje podacima iz raznih baza podataka. Uz sve to – PHP ima i veliku „online" zajednicu korisnika u svijetu i u Hrvatskoj te je poznavanje PHP-a čest preduvjet ili plus prilikom zapošljavanja kao web dizajner ili web programer.

PHP je najjednostavnije definirati kao *open-source server-side* skriptni programski jezik za dinamičko generiranje HTML koda.

Početnicima u ovom području prethodna rečenica potpuno je nerazumljiva, pa ćemo je probati malo pojasniti. *Open-source* znači da svatko tko želi može preuzeti izvorne php kodove te ako ima dovoljno znanja prilagođavati ih svojim željama i potrebama te potom koristiti. *Open-source* zajednice otvorene su za korisnike i unutar njih svi su pozvani razvijati nove verzije jezika. Dakle, PHP je potpuno besplatan i na raspolaganju je svim zainteresiranim osobama, može se koristiti za privatne ili u komercijalne potrebe.

Kao što smo naveli, PHP je i *server-side* skriptni jezik, što znači da se skripte napisane u PHP-u izvršavaju na poslužitelju (serveru). Ukratko, poslužitelj zaprimi zahtjev i potom izvršava PHP kod na osnovu kojeg se generira html kod koji se proslijeđuje korisniku (klijentu). Ovim načinom generiranja sadržaja klijent ne može vidjeti orginalni kod (skriptu) koji je generirao sadržaj koji gleda, već ima pristup čistom HTML kodu. Zato kažemo i da služi za dinamičko generiranje html koda.

I zadnja preostala stavka iz definicije koja zahtjeva dodatno pojašnjenje je - skriptni jezik. PHP je nastao na temeljima programskih jezika, ali ga nazivamo i skriptnim jezikom jer koristi gotove programerske komponente koje može interpretirati, a zatim automatizirati izvršavanje zadataka i ne zahtijeva eksplicitan korak prevođenja programa, za razliku od klasičnih programskih jezika.

5.2 Povijest PHP-a

PHP je kreirao Rasmus Lerdorf 1994. godine, kao alate koji posreduju između C programskog jezika i HTML-a. Orginalno ih je koristio za praćenje posjeta na svojoj web stranici, pa ih je stoga i nazvao Personal Home Page Tools – PHP Tools (alati za osobnu početnu stranicu). S vremenom je razvio više funkcionalnosti jezika kao što je interakcija s bazom podataka, kreiranje knjige posjetitelja i sl. pa je već 1995. godine omogućio javnu upotrebu PHP-a i on zapravo postaje *open-source* jezik. Drugi korisnici počinju raditi na popravljanju *bugova*, izmjenama i dopunama PHP-a. Počela se stvarati zajednica korisnika koji su kontinuirano radili na unaprijeđenju jezika.

Današnji PHP značajno se razlikuje od prvotne verzije alata i prerastao je u skriptni programski jezik. Od prvotne verzije PHP-a iz 1994. god., preko verzija 2.0, 3.0 i 4.0 došli smo do danas aktualne verzije 5, točnije trenutno aktualne verzije 5.4.31. Pretpostavlja se da je PHP danas instaliran na desetcima, a možda i stotinama milijuna domena u cijelom svijetu i teško ćete pronaći web stranicu većeg obujma koja ne koristi PHP.

5.3 Instalacija PHP-a

Kako bi bilo moguće pisati, testirati i u konačnici aktivno koristiti PHP skripte, potrebno je imati pristup poslužitelju i na njemu instalirati PHP. S obzirom da je svrha ovog poglavlja uputiti vas u osnove PHP-a, ovdje nećemo davati detaljne upute o njegovoj instalaciji, već ćemo vas uputiti na njihovu službenu web stranicu www.php.net gdje se nalaze detaljne upute i sve potrebne datoteke za instalaciju PHP-a.

Za potrebe razvoja web aplikacija ili vježbanje PHP-a, može se na klijentsko računalo instalirati XAMPP ili neka slična jednostavna *multi-platforma* s kojom instalirate Apache, MySQL, PHP, phpMyAdmin i još mnoge druge aplikacije korisne za razvoj/testiranje dinamičkih web stranica izravno na vašem računalu te učenicima i na taj način omogućiti pisanje i testiranje PHP skripti.

Slika 5.2: Prikaz završnog dijela instalacije PHP-a

5.4 Sintaksa PHP-a

Kao što i u svakom jeziku postoje sintaktička pravila koja proučavaju i definiraju odnose među riječima, tako i u skriptnom jeziku postoje pravila koja definiraju načine pisanja skriptnog jezika.

PHP može se pisati unutar html dokumenta, ali i kao zaseban dokument s ekstenzijom **.php**. PHP skriptu smještamo između oznaka **<?php** i **?>**. Sve unutar ovih oznaka interpreter smatra PHP kodom i automatski će ga provesti.

Najjednostavnija PHP skripta je primjerice ona za ispisivanje teksta u internet pregledniku. Potrebno je unutar oznaka za PHP kod unijeti jednu od naredbi za ispis (**echo** ili **print**) te zatim pod navodnicima navesti tekst koji želimo ispisati.

PHP primjer:

```
<html>
<body>

<?php

echo "Dobar dan!!!" ;

?>

</body>
</html>
```

Rezultat (u internet pregledniku):

PHP kod nije osjetljiv na bjeline, odnosno razmake i prijelome retka. Stoga svejedno je piše li:

```
raise_prices($sadrzaj, $inflacija, %troskovi)
```

ili

```
Raise_prices (
$sadrzaj,
$inflacija,
$troskovi
)
```

Uvrježena je praksa pisanja svake tvrdnje u novi redak (kao u drugom primjeru), dakle prelamanje koda, kako bi bio što čitljiviji i kako bi se programeru olakšalo snalaženje unutar skripte.

Kako je PHP kod zapravo niz naredbi, znak točka-zarez (;) koristi se za razdvajanje tvrdnji, iako u nekim situacijama nije nužan (npr. prije zatvaranja PHP koda), ali radi jednostavnosti najbolje ga je uvijek koristiti.

```
<?php
echo "Hello World" ;
?>
```

ali može i:

```
<?php
echo "Hello World"
?>
```


Vježba:

Prepišite primjer ispravno:

```
<?php
$a = 5
$b = 10
function myTest()
{ global $a, $b
$b = $a + $b;
}
myTest()
echo $b
?>
```

Rješenje:

```
<?php
$a = 5;
$b = 10;

function myTest()
{
global $a, $b;
$b = $a + $b;
}
myTest();
echo $b;
?>
```

Nakon izvršavanja PHP koda, korisnik ne može vidjeti izvorni kod, već samo rezultat. To nije karakteristika svih skriptnih jezika, zato je praktično PHP koristiti i u onim situacijama kada želimo zaštititi vlastitu skriptu.

PHP kod moguće je također upisivati na jednom ili više mjesta unutar html dokumenta. Za upisivanje koda više puta unutar html-a potrebno je svaki put PHP smjestiti unutar njegovih oznaka (**<?php** i **?>**).

5.4.1 Komentari u PHP-u

Komentari služe isključivo programerima koji pišu skriptu kao smjernice i podsjetnici i time im olakšavaju snalaženje unutar skripte. Ono što je upisano kao komentar nije vidljivo korisnicima. U PHP-u razlikujemo dvije vrste komentara:

- **jednolinijske komentare** (kratki komentari u jednom retku)
- **blok komentare** (duži komentari koji se mogu protezati kroz više redova).

Pisanje jednolinijskih komentara započinje znakom **//** prije samog komentara, a blok komentari započinju znakom **/***, a završavaju znakom ***/**.

Primjer:

```
<html>
<body>
<?php

//Ovo je jednolinijski komentar

?>
</body>
</html>
```

```
<html>
<body>
<?php

/*
 Ovo je blok
komentar, koji se proteže
kroz  više redaka
*/

?>
</body>
</html>
```

5.5 PHP varijable

Mjesto u memoriji rezervirano za pohranu podatka naziva se varijabla pa se može reći da je varijabla spremnik podataka. Varijable unutar PHP-a moguće je, kao i sve ostale varijable, mijenjati. Učenicima je slikovito moguće objasniti varijablu kao kutiju u koju spremamo određene stvari (odnosno podatke) te dalje rukujemo kutijom, a ne stvarima pojedinačno. Stvari koje smo spremili u kutiju možemo po potrebi zamijeniti drugim stvarima.

5.5.1 Deklariranje varijable

Sve PHP varijable počinju znakom **$**, nakon kojeg se upisuje ime varijable, a zatim se pomoću znaka jednakosti varijabli pridružuju određena vrijednosti, odnosno podatak. Jednom deklarirana varijabla može se koristiti, odnosno pozivati više puta unutar skripte.

Općeniti oblik deklariranja varijable:

$*ime_varijable* **=***vrijednost***;**

Pravila imenovanja varijabli u PHP-u

1. Ime varijable mora početi slovom ili donjom crtom (_), a ostali znakovi u imenu mogu biti brojevi, slova i crtica (-).
2. Ime varijable ne može sadržavati razmak.
3. Ime varijable osjetljivo je na mala i velika slova ($Ime i $ime dvije su različite varijable).

5.5.2 Vrste varijabli

U određenu varijablu mogu se smještati različite vrste podataka. Iako u PHP-u nije potrebno prilikom deklariranja varijable odrediti njenu vrstu, dobro je učenicima navesti vrste varijabli te ih uputiti u njihove međusobne razlike jer to predstavlja temelje programiranja. Vrsta varijable definirana je vrstom podataka koje u nju pohranjujemo.

PHP-u razlikujemo:

- *String* (tekstualne) varijable,
- *Integer* (brojčane) varijable i
- Logičke varijable.

Primjer *string* (tekstualne) i *integer* (brojčane) varijable:

```
<?php
    $txt="Hello World!";
    $x=16;
?>
```


5.5.2.1 *String* varijable u PHP-u

String varijable su tekstualne varijable koje se koriste za unos vrijednosti koja se sastoji od različitih znakova. To mogu biti podaci koji se sastoje od teksta, brojeva, hrvatskih slovnih znakova i sl. Sadržaj *string* tipa varijable uvijek se upisuje između navodnika.

Primjer:

```
<?php
  $ime="Ante";
?>
```

NAPOMENA: Sve *stringove* (nizove znakova, tj. tekst), kao i vrijednost *string* varijable, potrebno je u PHP-u uvijek upisivati unutar navodnika. Pri tom je moguće koristiti dvostruke (" ") i jednostruke navodnike (' '). Osnovna razlika je u tome hoće li se ispisivati vrijednost varijable ili ne.

Primjer 1:

```
<?
$str_ime="Marija";
echo ("Moje ime je $str_ime");
?>
```

Rezultat primjera 1:

Primjer 2:

```
<?
$str_ime=„Ivan";
echo ('Moje ime je $str_ime');
?>
```

Rezultat primjera 2:

5.5.2.2 *Integer* varijable u PHP-u

Integer ili cjelobrojnim varijablama nazivamo one varijable u koje pohranjujemo cijele brojeve. U ovaj tip varijable možemo pohraniti pozitivne i negativne brojeve u rasponu od -2 147 483 648 do 2 147 483 647, tj. 32 bita podataka. Vrijednost *integer* varijable ne upisuje se pod navodnike!

Primjer:

```
$godine=14;
```

5.5.2.3 Logičke varijable

Logički tip varijable ima samo dvije moguće vrijednosti, a to su istina (točno) i laž (netočno). Koriste se unaprijed rezervirane vrijednosti preuzete su iz engleskog jezika: **true** i **false**.

Deklaracija logičke varijable:

```
$logicka1 = true;
$logicka2 = false;
```

Vježba:

Koje varijable su točno napisane, a koje ne? Ispravite krivo napisane varijable i razvrstajte ih u skupine (string, integer, logičke).

$x=5
$sunce="rozo"
$Date=sutra
$_ime="Nikola"
$123=123
$Škola=true
$broj="12345"
$number="12345_6"
$num="false"

Rješenje:

Integer varijable:
$x=5;
$jedan23=123;

String varijable:
$sunce="rozo";
$Date="sutra";
$_ime="Nikola";

Logičke varijable:
$Skola=true
$num="false"

Imenujte varijablu AUTO i pridodajte joj vrijednost MAZDA. Zatim imenujte varijablu GODINE i pridodajte joj vrijednost 24. Kreirajte varijable koje će za vrijednost imat nazive mjeseci u godini. Kao posljednju deklarirajte varijablu DATUM ROĐENJA i kao vrijednost joj pridodajte svoj datum rođenja. Ispišite sve varijable.

Rješenje:

```
<?
//deklariramo varijable

$auto="Mazda";
$godine=24;
$prvi_mj="Siječanj";
$drugi_mj="Veljača";
$treci_mj="Ožujak",
$cetvrti_mj="Travanj";
$datum_rodenja="1.1.1991."

//ispisujemo varijable

echo $auto;
echo $godine;
echo $prvi_mj;
echo $drugi_mj;
echo $treci_mj,
echo $cetvrti_mj;
echo $datum_rodenja;
?>
```


5.6 Konstante

Osim varijabli, dakle promijenjivih jedinica, u PHP-u se možemo koristiti i konstantama. Razlika u deklariranju varijable i konstante je u tome što se za deklariranje konstante ne koristi znak $. Konstanta se deklarira pomoću ključne riječi define iza koje u zagradama, unutar navodnika, unosimo ime, a zatim i vrijednost konstante.

Primjer:

```
<?php
    define („A“, 6);
    echo A;
?>
```

Ovime smo deklarirali konstantu naziva A i pridodali joj vrijednost 6.

U PHP jeziku postoje unaprijed definirane konstante, kao što je broj Pi, pa tu konstantu nije potrebno definirati prije korištenja, već se može koristiti prethodno definirana konstanta M_PI (iznosi 3.14159265358979323846). Osim konstante M_PI postoje i konstante M_PI_2 (Pi/2), M_PI_4 (Pi/4), M_1_PI (1/Pi) itd.

5.7 PHP operatori

Operatori omogućavaju izvršavanje operacija nad varijablama i konstantama. To su simboli koji predstavljaju (zamjenjuju) određene operacije. Svi smo se s operatorima, kao što su oni za zbrajanje, oduzimanje, množenje i dijeljenje, već mnogo puta susreli u matematici. U PHP-u razlikujemo četiri osnovne skupine operatora:

1. Aritmetički operatori
2. Operatori dodjeljivanja (pridruživanja)
3. Operatori usporedbe
4. Logički operatori

5.7.1 Aritmetički operatori

Aritmetički operatori su operatori zbrajanja (+), oduzimanja (-), množenja (*), dijeljenja (/), ostatka dijeljenja (%), operator uvećanja za 1 (++) i umanjenja za 1 (--). U tablici 1 se nalazi pregled svih aritmetičkih operatora.

Prva četiri operatora su svima poznati operatori. Ostatak dijeljenja kao rezultat vraća ostatak pri dijeljenju cijelobrojnih varijabli. Operatori uvećanja, odnosno umanjenja za 1, određenu varijablu uvijek uvećavaju, odnosno umanjuju za 1. Ti operatori predstavljaju skraćeni oblik zapisa dodavanja broja 1 nekoj varijabli, npr.: $x=2, $x=2+1 ili skraćeno $x++.

Tablica 1: Pregled aritmetičkih operatora

Operator	Opis	Primjer	Rezultat
+	Zbrajanje	x=2 x+2	4
-	Oduzimanje	x=2 5-x	3
*	Množenje	x=4 x*5	20
/	Dijeljenje	15/5 5/2	3 2.5
%	Ostatak dijeljenja	5%2 10%8 10%2	1 2 0
++	Uvećanje za 1	x=5 x++	x=6
--	Umanjenje za 1	x=5 x--	x=4

5.7.2 Operatori dodjeljivanja

Operatori dodjeljivanja ili pridruživanja su oni operatori koji dodjeljuju određene vrijednosti ili varijable drugim varijablama. Najčešće korišten je operator pridruživanja (**=**) koji pridružuje određenu vrijednost varijabli (bio da se radi o *string* ili *integer* varijabli).

Osim operatora pridruživanja postoji cijeli niz kombiniranih operatora dodijeljivanja, koji uz aritmetički operator imaju i operator pridruživanja. Ti operatori zapravo određenoj varijabli dodjeljuju novu vrijednost. Možemo reći da pomoću kombiniranih operatora dodjeljivanja skraćujemo zapise u kojima se koriste aritmetički operatori. U sljedećoj tablici je pregled operatora pridruživanja.

Tablica 2: Pregled operatora dodjeljivanja

Operator	Primjer	Isto je kao da piše:
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

5.7.3 Operatori usporedbe

Operatori usporedbe omogućavaju nam uspoređivanje dviju vrijednosti, a najčešće uspoređujemo vrijednosti dviju varijabli. Kao rezultat usporedbe uvijek dobijemo logičku vrijednost **TRUE** (istina/točno) ili **FALSE** (laž/netočno).

Ovim operatorima možemo provjeravati je li jedna varijabla manja ili jednaka (**<=**) odnosno veća ili jednaka (**>=**) drugoj varijabli ili je li samo manja (**<**) ili samo veća (**>**) od druge varijable. Također možemo provjeriti i jesu li dvije varijable jednake (**==**) ili su im vrijednosti različite, tj. nisu jednake (**!=**).

NAPOMENA: Operator jednakosti (**==**) često se mijenja s operatorom pridruživanja (**=**) i u programima dolazi do pogreške zbog primjene krivog operatora. Dakle, operator pridruživanja (**=**) varijabli samo dodjeljuje određenu vrijednost, a operator jednakosti (**==**) provjerava jesu li dvije vrijednosti jednake. U tablici pogledajte pregled operatora usporedbe.

Tablica 3: Pregled operatora usporedbe

Operator	Opis	Primjer
==	Je jednako	5==8 točno
!=	Nije jednako	5!=8 točno
>	Je veće od	5>8 točno
<	Je manje od	5<8 točno
>=	Je veće ili jednako	5>=8 točno
<=	Je manje ili jednako	5<=8 točno

5.7.4 Logički operatori

Osnovni logički operatori su **AND** (i), **OR** (ili) i **NOT** (ne). Najčešće se koriste u kombinaciji s operatorima usporedbe, kada treba provjeriti određeni uvijet. Na primjer: potrebno je provjeriti je li vrijednost varijable *a* između 10 i 100. Kako bismo proveli tu provjeru koristit ćemo logički operator **AND** (**&&**), pa ćemo pisati *$a > 10* **&&** *$a < 100*. U tablici 4 pogledajte pregled logičkih operatora.

Tablica 4: Pregled logičkih operatora

Operator	Opis	Primjer
&&	And (i)	x=6 y=3 (x < 10 && y > 1)
||	Or (ili)	x=6 y=3 (x==5 || y==5)
!	Not (ne)	x=6 y=3 !(x==y)

5.7.5 Operator spajanja – točka (.)

Osim prethodno objašnjene 4 skupine operatora, u PHP-u postoji još različitih operatora, no za početnike je od tih operatora važan samo operator spajanja.

Za povezivanje vrijednosti dvaju varijabli koristi se operator spajanja - točka (**.**). Taj se operator najčešće koristiti pri ispisu vrijednosti više varijabli.

Primjer:

```
<?php
    $txt1="Dobar dan!";
    $txt2="Svaki dan je dobar dan!";

    echo $txt1 . " " . $txt2;
?>
```

Rezultat u pregledniku:

Vježba:

1. U Priloženom kodu nađite operatore, a zatim skriptu prepišite i skratite tako da postojeće operatore zamijenite OPERATORIMA DODJELJIVANJA

```
<?php
$x = 1;
$x = $x + 1;
echo $x . " ";
$y = 5;
$y = $y * $x;
echo $y . " ";
$z = 180;
$y = $y - 1;
$z = $z / $y;
echo $z;
?>
```

2. Ispiši tekst „Dobar dan“ u prozoru preglednika.

3. Pomoću varijabli i aritmetičkih operatora u PHP-u izračunaj:
10 + 7, 10 - 7,
10 * 7, 10 / 7,
ostatak dijeljenja 10 sa 7.
U prozoru preglednika ispišite zadatak i rezultat (npr. 10+7=17).

Rješenje 1:

```
<?php
$x = 1;
$x++;
echo $x . " ";
$y = 5;
$y *= $x;
echo $y . " ";
$z = 180;
$z /= --$y;
echo $z;
?>
```

Rješenje 2:

```
<?php
echo "Dobar dan!";
?>
```

Rješenje 3:

```
<?php
$x=10;
$y=7;
$r=$x+$y;
print "$x + $y = $r <br />";    $r=$x-$y;
print "$x - $y = $r <br />";    $r=$x*$y;
print "$x * $y = $r <br />";    $r=$x/$y;
print "$x / $y = $r <br />";    $r=$x%$y;
print "$x % $y = $r<br />";
 ?>
```


5.8 Uvjetne naredbe u PHP-u

Uvjetne naredbe su one naredbe koje omogućavaju grananje u skripti, odnosno programu. Kako sam pojam govori, u takvim naredbama mora postojati uvjet. Ako je uvjet zadovoljen izvršit će se određena akcija.

Učenicima je najlakše objasniti ulogu uvjetnih naredbi na primjeru. Zamislimo proljetni dan. Ivan kreće u školu. Ako prognoza predviđa da će taj proljetni dan biti kišan, Ivan treba ponijeti kišobran. Ako prognoza predviđa sunce, Ivan se treba namazati kremom sa zaštitnim faktorom. Dakle, Ivan prvo treba provjeriti naš uvjet – hoće li padati kiša ili će biti sunčano, a ovisno o tom uvjetu treba izvršiti jednu od radnji – ili ponijeti kišobran ili se namazati kremom.

Dakle, uvjetne naredbe najčešće se koriste u situacijama kada je potrebno izvršiti različite radnje, ovisno o istinitosti određene tvrdnje.
U PHP-u postoje četiri vrste uvjetnih naredbi, a to su:

- if
- if...else
- if...elseif....else
- switch

5.8.1 If naredba

If naredba koristi se kako bi se određene akcije (određeni blok koda) izvršile **jedino AKO** je određeni uvjet ispunjen. Ključna riječ naredbe je **if**, iza ključne riječi u zagradama se definira uvjet, a iza uvjeta u vitičastim zagradama definiraju se radnje koje će se izvršiti ukoliko je uvjet zadovoljen.

Opći obilk zapisa if uvjetne naredbe:

```
if (uvjet)
{
kod koji se izvršava ukoliko je uvjet ispunjen;
}
```

Primjer:

```
<?php
$a=3;
$b=5;

if ($a==$b)
{
echo "Varijable su jednake."
};
?>
```

Dakle, u primjeru varijabla a ima vrijednost 3, a varijabla b ima vrijednost 5. Postavljen je uvjet koji provjerava jednakost varijable a i b. Ukoliko je uvjet ispunjen, dakle ukoliko je vrijednost varijable a jednaka vrijednosti varijable b, ispisat će se poruka „Varijable su jednake“. Ako uvjet nije ispunjen (kao što je slučaj u ovom primjeru) neće se dogoditi ništa.

5.8.2 If...else naredba

If...else naredba koristi se kako bi se određene radnje (određeni blok koda) izvršile jedino AKO je određeni uvjet ispunjen, a ako uvjet NIJE ISPUNJEN izvršavaju se druge radnje (drugi blok koda). Naredba if...else u odnosu na naredbu if, predviđa radnje koje će se izvršiti ako uvjet nije ispunjen. Ključne riječi naredbe su **if** i **else**. Naredbu počinjemo riječju if, iza ključne riječi u zagradama definira se uvjet, a iza uvjeta u vitičastim zagradama definiraju se radnje koje će se izvršiti ukoliko je uvijet zadovoljen, nakon toga ide ključna riječ else i u vitičastim zagradama iza nje radnje (blok koda) koje će se izvršiti u slučaju da uvjet nije zadovoljen.

Opći obilk zapisa if...else uvjetne naredbe:

```
if (uvjet)
        {
           kod koji se izvršava ukoliko je uvjet ispunjen;
        }
else
        {
           kod koji se izvrava ukoliko uvjet nije ispunjen;
        }
```

Primjer:

```
<?php

$a=3;
$b=4;

if ($a==$b)
 {
 echo "Varijable su jednake";
}
else
{
 echo "Varijable nisu jednake";
}
?>
```

Dakle, u primjeru varijabla a ima vrijednost 3, a varijabla b ima vrijednost 5. Postavljen je uvjet koji provjera je li varijabla a jednaka varijabli b. Ukoliko je uvjet ispunjen ispisat će se poruka „Varijable su jednake". Ako uvjet nije ispunjen (kao što je slučaj u ovom primjeru) ispisat će se poruka „Varijable nisu jednake".

5.8.3 If...elseif...else naredba

If...elseif...else naredba omogućava postavljanje više uvjeta i njihovu provjeru te izvršenje zadane radnje ovisno o tome koji je od postavljenih uvjeta ispunjen. Za razliku od if...else naredbe u kojoj je moguće definirati samo jedan uvjet te radnje koje će se izvršiti u slučaju kada je uvjet zadovoljen i u slučaju kada uvjet nije zadovoljen, unutar if...elseif...else naredbe moguće je definirati 2 ili više uvjeta te radnje koje će se izvršiti ovisno o tome koji je od definiranih uvjeta ispunjen i radnje koje će se izvršiti ako niti jedan od postavljenih uvjeta nije ispunjen.

Ključne riječi naredbe su **if**, **elseif** i **else** . Naredbu počinjemo riječju **if**, iza ključne riječi u zagradama definira se uvjet, a iza uvjeta u vitičastim zagradama definiraju se radnje koje će se izvršiti ukoliko je uvjet zadovoljen.

Nakon toga ide ključna riječ **elseif** i u zagradama drugi uvjet koji će se provjeriti u slučaju da prvi uvjet nije zadovoljen. U vitičastim zagradama iza drugog uvjeta potrebno je definirati radnje koje će se izvršiti ukoliko je drugi uvjet ispunjen. Naredbu **elseif** s uvjetom i radnjama za izvršenje možete ponavljati više puta, mijenjajući definirane uvjete.

Na kraju se pomoću ključne riječi **else** definira radnja koja se upisuje u vitičaste zagrade i koja će se izvršiti ako niti jedan od postavljenih uvjeta nije zadovoljen. Važno je napomenuti da uz ključnu riječ else nikada ne pišemo uvjet!

Opći obilk zapisa if...elseif...else uvjetne naredbe:

```
if (uvjet1)
        {
           kod koji se izvršava ukoliko je uvjet ispunjen;
        }
elseif (uvjet2)
        {
           kod koji se izvrava ukoliko prvi uvjet nije, a drugi uvjet je ispunjen;
        }
else
        {
           kod koji se izvršava ako niti jedan od postavljenih uvjeta nije ispunjen;
        }
```

Primjer:

```
<?php
$a=3;
$b=4;

if ($a==$b)
{  echo "Varijable su jednake";
}
elseif ($a<=$b)
 { echo "Varijabla a je manja od varijable b";
}
elseif ($a>=$b)
 { echo "Varijabla a je veća od varijable b";
}
else
{ echo "Jedna od varijabli nije broj";
}
?>
```

U primjeru varijabla a ima vrijednost 3, a varijabla b ima vrijednost 5. Prvim uvjetom provjerava se je li varijabla a jednaka varijabli b. Ukoliko je uvjet ispunjen ispisat će se poruka „Varijable su jednake". Ako taj uvjet nije ispunjen, a varijabla a je manja ili jednaka varijabli b (kao što je slučaj u ovom primjeru) ispisat će se poruka „Varijabla a je manja od varijable b". Kada taj uvjet ne bi bio ispunjen, došlo bi do provjere trećeg uvjeta tj. provjere je li varijabla a veća ili jednaka varijabli b i ako bi se uvjet ispostavio točnim ispisala bi se poruka "Varijabla a je veća od varijable b". U slučaju kada ni jedan od postavljenih uvjeta ne bi bio ispunjen ispisala bi se poruka "Jedna od varijabli nije broj".

5.8.4 Switch nareba

Naredba *switch* koristi se za izbor jednog od više predviđenih blokova koda za izvršavanje, ovisno o istinitosti postavljenih uvjeta. Odnosno, naredba switch ponaša se jednako kao i naredba if...elseif...else, samo što omogućava jednostavnije i preglednije pisanje naredbe.

Ključne riječi su **switch** i **case**. Naredbu počinjemo riječju **switch**, zatim u zagradama navodimo varijablu čiju je vrijednost potrebno provjeravati. U vitičastim zagradama upisujemo riječ **case** i iza nje, pod navodnike, vrijednost varijable koju provjeravamo, zatim stavljamo dvotočku i upisujemo akcije za koje želimo da se izvrše ukoliko provjeravana varijabla ima navedenu vrijednost. Potom je potrebno upisati rezerviranu riječ break te **točku-zarez**. Ponavljamo ponovno isti unos za sve vrijednosti varijable koje želimo provjeriti. Naredbu *switch* možemo završiti unosom ključne riječi **default** i unosom potrebnih naredbi. Naredbe upisane u default dijelu one su naredbe koje će se izvršiti ako ni jedan postavljenih uvjeta nije ispunjen.

Opći obilk zapisa *switch* naredbe:

```
<?
switch (ime varijable)
        {
case "vrijednost varijable 1":
        kod koji se izvršava ukoliko je uvjet ispunjen;
                break;
case "vrijednost varijable 2":
        kod koji se izvršava ukoliko je uvjet ispunjen;
                break;
case "vrijednost varijable 3":
        kod koji se izvršava ukoliko je uvjet ispunjen;
                break;
case "vrijednost varijable 4":
        kod koji se izvršava ukoliko je uvjet ispunjen;
                break;
default:
        kod koji se izvršava ukoliko niti jedan uvjet nije ispunjen;
                break;
        } ?>
```

Primjer:

```
<?php
$dan = "Utorak";

switch ($dan)
{ case "Ponedjeljak":
echo "To je očajan dan";
break;
case "Utorak":
 echo "Ništa bolji dan od ponedjeljka";
 break;
case "Srijeda":
echo "Srijeda je tračak svjetla na
        kraju tunela";
break;
case "Četvrtak":
echo "Još malo pa Petak";
 break;
case "Petak":
echo "Vikeeeeeend!!! :D";
break;
default:
echo "Dobar dan!";
break;
}
?>
```

NAPOMENA: Upotreba naredbe break vrlo je važna. Ona sprječava izvođenje naredbi koje slijede u switch bloku i nastavlja izvođenje nakon switch bloka. Ako bi se u ovom primjeru ispustila naredba break na svim mjestima tada bi se u slučaju kada varijabla $dan ima vrijednost „Ponedjeljak“ izvršile sve naredbe u cijelom switch bloku, dakle i naredbe pod akcijom „Utorak“, „Srijeda“, „Četvrtak“ itd. i default naredbe. Ako bi varijabla imala vrijednost „Petak“ tada bi se izvršile i naredbe u default dijelu.

S obzirom na to da je na početku ove skripte definirano da varijabla dan ima vrijednost Utorak, rezultat primjene skripte bio bi ispis rečenice „Ništa bolji dan od ponedjeljka“ u prozoru preglednika.

5.9 Nizovi u PHP-u

Varijable kojima smo se do sada bavili sadržavale su po jednu vrijednost. No, ponekad se javi potreba da u jednu varijablu stavimo više vrijednosti. To nam omogućavaju nizovi. Dakle, varijable koje sadrže dvije i više vrijednosti nazivamo nizovima. Točnije rečeno, niz se sastoji od više elemenata, a svaki od tih elemenata sadrži jednu vrijednost. Nizovi se koriste kako bismo u njih pohranjivali razne popise, elemente tablica i sl. te lakše upravljali njima. Nizove često susrećemo u radu s bazama podataka, pa su tako prisutni i u skriptama unutar CMS-a (sustava za upravljanje sadržajem).

Učenicima je zgodno slikovito objasniti ulogu nizova pomoću sljedećeg primjera. Zamislimo farmu pilića. U određenom trenutku potrebno je svako pile uhvatiti, izvagati i podatke o težini upisati u PHP skriptu. Ako na farmi ima 1000 pilića, potrebno je u skripti deklarirati 1000 varijabli i svakoj od njih pridružiti težinu pilića. Kako ne bismo morali deklarirati 1000 varijabli pojedinačno, mi kreiramo niz.

Slika 5.3: Pilići kao elementi niza

Postoji više vrsta nizova, a možemo ih podijeliti na jednodimenzionalne i višedimenzionalne te cjelobrojne i asocijativne nizove. Kada se niz sastoji od više elemenata koji imaju svoje vrijednosti govorimo o jednodimenzionalnom nizu. Kada su elementi niza također nizovi govorimo o višedimenzionalnom nizu.

Kako bismo mogli upravljati vrijednostima koje spremamo u niz potrebno je svaki element niza označiti, odnosno pridodati mu indeks. Ako svakom elementu niza pridružimo brojčani indeks govorimo o cjelobrojnom nizu, a ako kao oznaku indeksa koristimo slova, takav niz nazivat ćemo asocijativnim.

5.9.1 Jednodimenzionalni cjelobrojni nizovi

U jedan niz moguće je spremiti veliki broj vrijednosti, bez deklariranja velikog broja varijabli. Jednodimenzionalni cjelobrojni niz deklariramo pomoću znaka **$** iza kojeg navodimo ime niza, zatim u uglate zagrade upisujemo indeks člana niza te mu pomoću operatora pridruživanja (**=**) dodjeljujemo vrijednost.

Opći oblik deklariranja niza:

*$naziv_niza [**index_člana_niza**] = vrijednost;*

Kod cjelobrojnih nizova indeks člana niza definiran je brojevima, odnosno svakom članu niza pridružuje se brojčana vrijednost i to počevši od nule. Dakle, prvi član niza ima indeks 0, drugi član niza 1, treći 2 itd. Za naš primjer s pilićima niz bi izgledao ovako:

```
$ pile[0]= 115;
$ pile[1]= 116;
$ pile[2]= 311;
$ pile[3]= 318;
$ pile[4]= 415;
      ...
$ pile[999]= 113;
```

Prvi član niza, dakle pile broj jedan, bio bi označen indeksom 0, a tisućiti član niza, odnosno posljednje pile indeksom 999.

Niz se kraće može zapisati i na sljedeći način:

*$naziv_niza = **array** (vrijednosti člana niza)*

Odnosno za gornji primjer:

$pile = array (115, 116, 311, 318, 415);

I kod kraćeg zapisivanja PHP svakoj unesenoj vrijednosti dodjeljuje indeks, počevši od nule koju dodjeljuje prvoj unesenoj vrijednosti.

5.9.1.1 Pristup jednodimenzionalnom cjelobrojnom nizu

Da bi se putem PHP-a pozvao ili ispisao određeni član ili članovi niza potrebno je naznačiti indeks člana niza. Dakle, potrebno je znati i PHP-u zadati brojčanu oznaku člana niza za koji želimo ispisati vrijednost. Na primjeru naših pilića to bi izgledalo ovako:

```
<?
    $pile = array (115, 116, 311, 318, 415);
    echo $pile[0], $pile [1], $pile [2], $pile [4];
?>
```

Rezultat u pregledniku:

5.9.2 Jednodimenzionalni asocijativni niz

Jednodimenzionalni asocijativni niz se od cjelobrojnog razlikuje samo u načinu definiranja indeksa članova niza. Umjesto brojem, član niza definiran je određenom tekstualnom vrijednosti, bilo da je to jedna riječi ili samo jedno slovo.

Primjer niza:

```
$boja_kose[Marko] = "plava";
$boja_kose[Petar] = "smedja";
$boja_kose[Vlatka] ="crvena";
$boja_kose[Sanja] = "crna";
$boja_kose[Ivan] = "crna";
```

Ovaj niz može se nešto kraće zapisati i na način:

```
$boja_kose=array (
    "Marko"=>"plava",
    "Petar"=>"smedja",
    "Vlatka"=>"crvena",
    "Sanja"=>"crna",
    „Ivan"=>"crna" );
```

U primjeru *boja_kose* je ime niza, a indeksi su definirani u obliku imena osoba.

5.9.3 Višedimenzionalni nizovi

Jednodimenzionalni nizovi su oni nizovi čiji su elementi definirani jednim indeksom. Višedimenzionalni nizovi su oni nizovi čiji su elementi nizovi, odnosno svaki element višedimenzionalnih nizova definiran je pomoću dva i više indeksa. Ako se definiraju pomoću dva indeksa nazivaju se **dvodimenzionalni** nizovi, s tri indeksa **trodimenzionalni** nizovi itd.

Ovdje ćemo pokazati samo primjer deklariranja dvodimenzionalnog asocijativnog niza:

```
$osoba["Ana"]["visina"]=185;
$osoba["Ana"]["tezina"]=80;
$osoba["Ivan"]["visina"]=180;
$osoba["Ivan"]["tezina"]=90;
```

Ili

```
$osoba = array (
    "Ana"=>array("visina"=>185,
        "tezina" => 80),
    "Ivan"=>array("visina"=>180,
        "tezina"=>90)
);
```

Višedimenzionalni nizovi zahtjevaju naprednije poznavanje PHP-a, a s obzirom da je ovo poglavlje zamišljeno kao uvod u PHP jezik, nećemo ulaziti u dubinu.

5.10 Petlje u PHP-u

Često je potrebno istu naredbu ili grupu naredbi izvesti veći broj puta što znači da određeni, gotovo isti kod, trebamo upisati veliki broj put. Kako bismo uštedjeli vrijeme, povećali preglednost skripte i olakšali si posao možemo koristiti petlje. Petlje istu radnju ponavljaju više puta, odnosno sve dok je postavljeni uvjet zadovoljen.

U PHP-u postoje 4 vrste petlji:

- *for* petlja,
- *for... in* petlja,
- *while* petlja i
- *do... while* petlja.

5.10.1 For petlja

For petlja izvršava blok naredbi ako je izraz u uvjetu istinit (*true*). Uvjet se ispituje prije izvođenja bloka naredbi. Zbog toga je moguće da se blok naredbi ne izvrši niti jednom ukoliko uvjet postavljen na početku nije zadovoljen (*false*). For petlju koristimo najčešće u onim situacijama kada znamo točan broj potrebnih ponavljanja.

Logika for petlje:

```
for (početni izraz; uvjet; dodatni izraz)
 {
  blok naredbi;
 }
```

Petlja se definira ključnom riječju **for**, zatim se u zagradama navodi **početni izraz**, **uvjet** i **dodatni izraz**. U vitičaste zagrade upisuju se radnje, tj. blok koda koji će se izvoditi dok god je uvjet zadovoljen. Kako smo spomenuli, u for petlji postoje tri parametra koji se odvajaju točka zarezom.

Pogledajmo primjer:

```
<? php

for ($i=10; $i>=0; $i--)
    {
        echo '$i = ' . $i . '<br>';
    }

?>
```

Rezultat u prozoru preglednika:

U gornjem primjeru **početni izraz** ($i=10) pridružuje varijabli **i** vrijednost 10. Drugim parametrom postavljamo **uvjet** ($i>0) koji mora biti zadovoljen da bi petlja nastavila s izvođenjem. Treći parametar - **dodatni izraz** ($i--) umanjuje vrijednost varijable i za 1. Dakle, sve dok je uvjet zadovoljen, petlja će ispisivati vrijednost varijable **i**.

5.10.2 While petlja

While petlja izvršava blok naredbi dokle god je izraz u uvjetu istinit (*true*). Uvjet se ispituje prije izvođenja bloka naredbi. Zbog toga je moguće da se blok ne izvrši niti jednom ukoliko uvjet postavljen na početku nije zadovoljen (*false*). Razlika izmedju **while** i **for** petlje nije velika, no *for* petlja izvršava odredjene akcije jednom, ukoliko je postavljeni uvijet ispunjen, a *while* petlja izvršava blok koda sve dok je uvijet ispunjen. *While* petlja koristi se kada ne znamo unaprijed potrebni broj ponavljanja. Kada je poznat broj ponavljanja obično se koristi *for* petlja jer je brža.

Logika while petlje:

```
while ( uvjet ) {
// naredbe koje se izvršavaju dok je uvjet true
}
```

Petlja započinje ključnom riječju **while**, zatim se u zagrade piše **uvjet**, a u vitičaste zagrade blok koda koji će se ponavljati dok god je uvjet zadovoljen.

U odnosu na *for* petlju, razlika je u postavljanju početnog i dodatnog izraza. U *while* petlji **početni izraz** definiramo izvan same petlje, dakle deklariramo varijablu i pridružujemo joj određenu vrijednost. **Uvjet** postavljamo nakon postavljanja same petlje, odnosno uvjet upisujemo u oble zagrade nakon riječi *while*. **Dodatni izraz** definiramo nakon bloka koda koji se treba izvršiti ukoliko je uvijet zadovoljen.

Primjer:

```
<?

$brojac = 1;

while ( $brojac <= 3 )
        {
            echo $brojac." ";
            $brojac++;
        }
?>
```

Rezultat u prozoru preglednika:

Početni izraz određen je varijablom *brojac* koja ima vrijednost 1, **uvjet** petlje je da je vrijednost varijable *brojac* manja ili jednaka 3. Ako je uvjet zadovoljen ispisat će se vrijednost varijable, a zatim će se varijabla uvećati za 1 (**dodatni izraz**) i petlja će krenuti od početka.

5.10.3 Do ... while

Do...while petlja obavlja iste radnje kao i *while* i *for* petlja, odnosno ima sposobnost ponavljanja određenih radnji veći broj puta. No, razlika je u tome, što se u *do...while* petlji uvjet ispituje tek nakon izvršavanja bloka naredbi. Dakle, blok naredbi se prvo izvrši, a onda se provjerava uvjet, što znači da će se petlja sigurno jednom izvršiti i u situaciji kada uvjet nije zadovoljen. Naravno, ako je uvjet zadovoljen, petlja će se izvršavati sve dok je to tako.

Logika do...while petlje:

Petlja započinje ključnom riječju **do**, a zatim se u vitičaste zagrade upisuje blok naredbi, po zatvaranju zagrada upisujemo ključnu riječ **while** i u oble zagrade upisujemo **uvjet**.

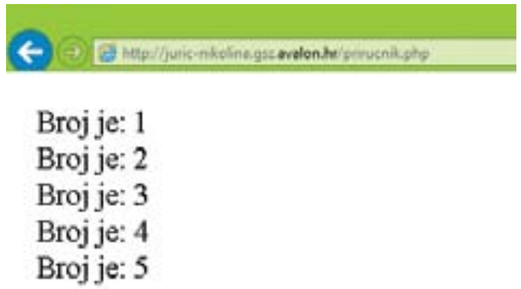
```
do
    {
    // naredbe koje se izvršavaju dok je uvjet true
    }
while ( uvjet )
```

No, potrebno je prije početka same petlje definirati **početni izraz**. Zatim započeti samu petlju ključnom riječju **do**, i u vitičaste zagrade osim bloka koda koji će se izvršiti, upisati i **dodatni izraz**. Nakon vitičastih zagrada upisuje se riječ **while** i **uvjet** u zagrade.

Primjer:

```
<?php
    $x=1;
    do
      {
       echo "Broj je: $x <br>";
       $x++;
       }
    while ($x<=5)
?>
```

Rezultat u prozoru preglednika:



5.10.4 For...each petlja

For...each petlja složenija je i naprednija petlja od ostalih petlji. Ovdje ćemo ukratko pojasniti njezinu najčešću upotrebu, a to je pristup nizovima. Petlja prolazi kroz svaki element danog niza i za njega izvršava definirani blok naredbi. Može spremiti indeks i vrijednost svakog elementa niza u posebne varijable u kojima se za svako ponavljanje petlje nalaze indeks i vrijednost elementa niza na kojem se trenutno nalazi unutarnji pokazivač. Unutarnji pokazivač se prije ulaska u petlju nalazi na nuli i svakim novim krugom u petlji povećava se za 1. Novim zvanjem *for...each* petlje unutarnji pokazivač se resetira. Petlja se vrti sve dok ne ostane bez elemenata niza.

Logika foreach petlje:

Petlja počinje ključnom riječju **foreach** i zatim se u zagradama upisuje **naziv niza**, ključna riječ **as** i zatim se unosi naziv varijable koja označava vrijednosti članova niza. Nakon što smo to definirali u vitičaste zagrade upisujemo blok naredbi koje je potrebno izvršiti nad članovima niza.

```
foreach ($ime_niza as $vrijednosti_niza)
    {
    // naredbe koje se izvršavaju za svaki element niza
    }
                                        ili
foreach ($neki niz as $kljuc => $vrijednost)
    {
    // naredbe koje se izvršavaju za svaki element niza
    }
```

5.10.4.1 Pristup jednodimenzionalnim asocijativnim nizovima pomoću for...each petlje

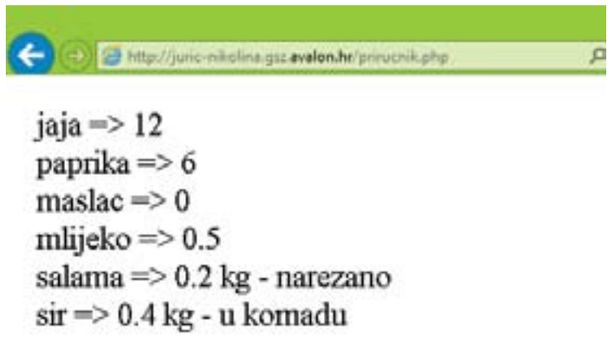
Primjer niza:

```
$ hladnjak=array(
"jaja"=>12,
"paprika"=>6,
"maslac"=>0,
"mlijeko"=>0.5,
"salama"=>"0.2 kg - narezano",
"sir"=>"0.4 kg - u komadu"
);
```

Primjer ispisa popisa i količine namirnica u hladnjaku (niza iz primjera):

```
foreach ($hladnjak as $kljuc => $vrijednost)
    {
    echo "$kljuc => $vrijednost <br>";
    }
```

Rezultat u prozoru preglednika bit će:



5.10.4.2 Pristup višedimenzionalnim asocijativnim nizovima pomoću for...each petlje

Primjer ispisa niza pomoću ugnježđene for...each petlje:

```
$osoba = array (
    "Ana"=>array("visina"=>185,
                 "tezina" => 80),
    "Ivan"=>array("visina"=>180,
                 "tezina"=>90)
                   );

foreach ($osoba as $indeks1 => $vrijednost1)
    {
    foreach ($vrijednost1 as $indeks2 => $vrijednost2)
        {
        echo "Osoba[".$indeks1."]
        [".$indeks2."]=".$vrijednost2."<br>";
        }
    }
```

Rezultat u prozoru preglednika:

5.11 Funkcije

U PHP-u, kao i u mnogim drugim skriptim i programskim jezicima, postoje prethodno definirane funkcije, ali je moguće kreirati i vlastite. Dakle, funkcijama nazivamo onaj dio koda koji smo sami napisali, a moguće ga je upotrijebiti, odnosno pozvati, veći broj puta, ovisno o potrebi. Ako nam se unutar skripte dio nekog koda, primjerice neki izračun, često ponavljaju moguće ih je smjestiti u funkciju i pozivati po potrebi, što ubrzava pisanje koda, skraćuje dužinu skripte, olakšava naknadne izmjene i omogućava veću točnost u pisanju skripti.

Deklariranje funkcije

Funkcija se deklarira ključnom riječju **function** iza koje upisujemo ime funkcije, u obične zagrade se upisuju parametri funkcije koji se odvajaju zarezom, a u vitičaste zagrade smještamo tijelo funkcije, odnodno blok koda koji će se izvršiti svaki put kada funkciju pozovemo.

NAPOMENA: Za imenovanje funkcija vrijede jednaka pravila kao i za imenovanja varijabli.

```
function mojaprvafunkcija (parametri funkcije)
    {
    naredbe unutar funkcije;
    }
```

U tijelo funkcije može se smjestiti bilo koji oblik PHP naredbi, dakle od naredbi za ispis varijabli, do petlji, nizova, uvjetnih naredbi i sl. Jednom napisanu funkciju možete pozivati više puta. Pogledajmo primjer ukojem jednom deklarirana funkcija, pod nazivom *funkcija1*, pozvana je još dva puta te će se dva puta ispisati tekst "Wohooo!!!".


```
<?php

function funkcija1()
 {
 echo "Wohooo!!!!<br />";
 }

 echo "Baš je fora ovaj PHP<br />";
        funkcija1();
 echo "Jedva čekate da sve naučite!! <br />";
 echo "PHP je uvijek... <br />";
        funkcija1();
?>
```

Rezultat u prozoru preglednika:

5.12 PHP i HTML forme - POST i GET metode slanja i prihvata podataka

Vrlo je česta upotreba PHP-a u proslijeđivanju podataka iz html skripti. S obzirom da primjena metoda zahtjeva i poznavanje html-a, ovdje ćemo informativno objasniti dvije osnovne metode i navesti njihove razlike. Dakle u PHP-u postoje dvije metode prosljeđivanja podataka HTML forme PHP dokumentu:

- **POST** metoda
- **GET** metoda

Podatke forme šaljemo pomoću GET metode putem linije za naredbe (*query string*), tj. iza znaka **?** u *URL*-u. Odabirom metode POST podaci nisu vidljivi u liniji za naredbe već se šalju kroz *header* HTTP *requesta* i time se na podatke ne može utjecati izmjenom linka, kao što to može biti slučaj s GET metodom.

Postoji ograničenje u količini podataka koji se mogu poslati putem GET metode. Ograničenje ovisi o postavkama samog servera i kreće se od 256 bita do čak 32 KB. Zato, ukoliko se radi o formi koja u sebi sadrži polja za unos dužeg teksta metoda, GET nije dobra metoda prosljeđivanja podataka.

Odabir metode koja se koristi za prosljeđivanje podataka ovisi o situaciji. Svaka metoda ima svoje vrline i mane. S obzirom da se u GET metodi podaci lijepe na sam URL, ova je metoda pogodna u slučajevima kada se želi omogućiti posjetiteljima spremanje stranice u svoje favorite jer će spremiti URL zajedno s query stringom. No, GET je vrlo nesigurna metoda jer ju korisnik vrlo lako može izmijeniti, pa se ne preporuča koristiti ovu metodu za prosljeđivanje recimo korisničkih imena i lozinki i sličnih podataka.

Postavljanje metode prosljeđivanja vrši se navođenjem u *method argumentu* **<form>** *taga*. Moguće vrijednosti method argumenta su "post" ili "get".

U primjeru se radi o istoj formi, u slučaju a) koristi se GET metoda, a u slučaju b) POST metoda. Primijenite skripte i promatrajte što se događa s podacima u naredbenoj liniji preglednika.

Slučaj a)

Prvi dokument nazvat ćemo **forma.html** i u njemu ćemo izraditi skriptu s html formom.

```
<html>
<head>
<title>Forma a) GET metoda</title>
</head>
<body>
    <form method="get" action="prihvat.php">
        Unesi svoje ime :<br>
        <input name="ime" type="text" ><br>
        <input name="submit" type="submit" value="Posalji">
    </form>
</body>
</html>
```

Drugi dokument nazvat ćemo **prihvat.php** i u njemu nalazit će se PHP skripta koja će prihvatiti informacije iz html forme i ispisati ih.

```
<?
    echo "Pozdrav".$_GET["ime"];
?>
```


Slučaj b)

Prvi dokument nazvat ćemo **forma2.html** i u njemu ćemo izraditi skriptu s html formom.

```
<html>
<head>
<title>Forma b) POST metoda</title>
</head>
<body>
    <form method="post" action="prihvat.php">
        Unesi svoje ime :<br>
        <input name="ime" type="text" ><br>
    <input name="submit" type="submit" value="Posalji">
</form>
</body>
</html>
```

Drugi dokument nazvat ćemo **prihvat.php** i u njemu nalazit će se PHP skripta koja će prihvatiti informacije iz html forme i ispisati ih.

```
<?
    echo "Pozdrav".$_POST["ime"];
?>
```

Više vježbi iz svih poglavlja PHP-a potražite na portalu **www.e-skola.com.hr**!

6. poglavlje

STOP MOTION animacija

dodatni sadržaj

6.1 Uvod

Stop motion animacija jednostavan je i danas svima dostupan put za otkrivanje svijeta animacije. Riječ animacija dolazi od lat. riječi *animatio* što znači oživjeti, dati dušu, pa je i proces animacije pomalo mističan i animatoru daje osjećaj posebne moći, što učenike često i privuče, a zatim i zadrži u svijetu animacije. Stop motion animacija, za razliku od većine drugih animatorskih tehnika, relativno je jednostavna, prilagodljiva raznim uvjetima, zahtijeva malo opreme i može se izrađivati s djecom različite dobi. Ova tehnika animacije omogućava izradu vrhunskih animacija, ali je praktična i za početnike jer čak i gruba, jednostavna stop motion animacija vrlo je efektna i kroz nju se stječe spoznaja o animatorskom procesu.

Cilj ovog poglavlja je predstaviti temu iz svijeta multimedije. Kako su danas animacije, animirani filmovi i animirani likovi prisutniji nego ikada, a svaki multimedijalni uređaj svakodnevno reproducira mnoštvo animacija, odlučili smo predstaviti taj segment multimedije. Prostor u ovom priručniku ograničen je i bilo je potrebno precizirati temu, pa je odluka pala na stop motion animaciju iz nekoliko razloga. Osim što zahtijeva malo opreme i praktična je za rad s djecom, posebno za uvođenje djece u svijet animiranja, stop motion animacija vezana je za fotografiju koja je danas također iznimno važan i neizostavan medij u multimedijalnom svijetu. Svi posjedujemo fotoaparate, bilo da se radi o kamerama na mobilnim ili pametnim telefonima, običnim digitalnim fotoaparatima, poluprofesionalnim fotoaparatima ili profesionalnim fotoaparatima. Snimanje fotografija postalo je neizostavan dio svakodnevice.

Kroz ovo poglavlje o stop motion animaciji želimo pružiti, kako nastavnicima, tako indirektno i učenicima, priliku za usavršavanje fotografskih znanja i njihovu kreativnu primjenu na jedan sasvim nov način.

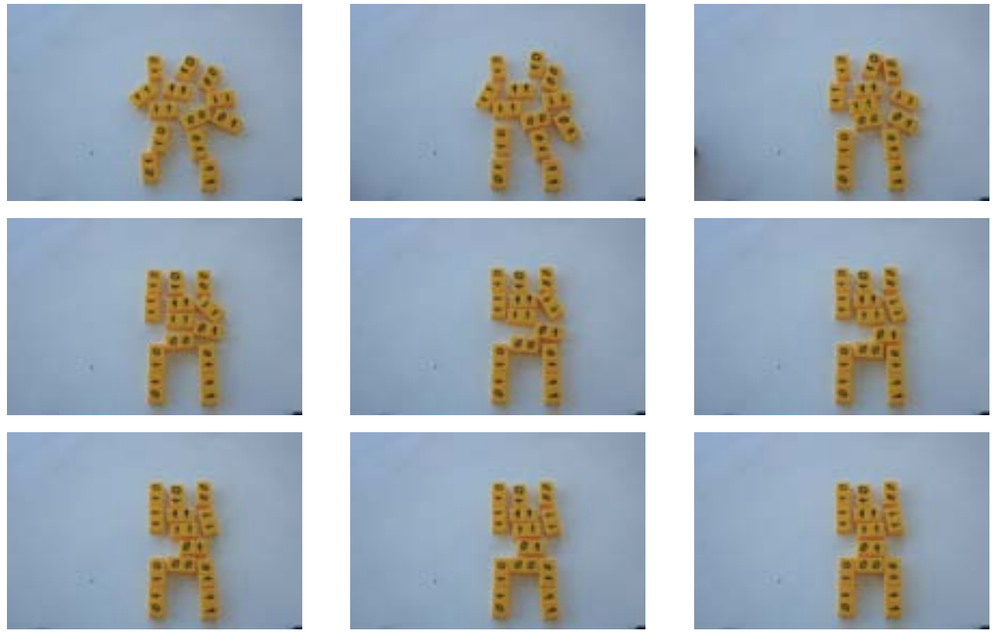
6.1.1 Osnovni pojmovi stop motion animacije

Stop motion animacija nastala je kao filmski postupak, odnosno tehnika, koja omogućava prikaz kretanja statičnih objekata. Dakle, objekte koje nije bilo moguće prikazati kao pokretne nijednom filmskom tehnikom, animiralo se pomoću tehnike stop motion.

Kako bismo mogli definirati stop motion animaciju potrebno je prvo definirati samu animaciju. Postoji više različitih definicija, pa se kaže da je animacija optička iluzija ili jednostavno pokretna slika. No, animacija je,tehnički gledano, zapravo prikaz većeg broja slika (bilo ručnih ili računalnih crteža ili fotografija) koje se razlikuju u što manjim detaljima u što kraćem vremenu. Što je vrijeme prikaza kraće, broj slika veći, a razlika u detaljima među slikama manja, animacija će biti „glađa". Oko gledatelja zapravo zavaravamo, pa statične slike doživljava kao pokretne. Mogućnost ove optičke iluzije dugujemo fenomenu „tromosti oka", tj. činjenici da se slika koju naše oko vidi zadržava otprilike četvrtinu sekunde u našem oku. Dakle, kako bi stvorili efekt kretanja objekata na slikama, potrebno je prikazati min. 4 – 5 slika u jednoj sekundi. Što je više slika u jednoj sekundi to je animacija uvjerljivija. Broj sličica koje prikažemo u jednoj sekundi označava se oznakom fps (frames per second). U današnjoj računalnoj animaciji potrebno je barem 12 sličica u sekundi, dok se za „klasične" (Disneyeve) animacije crtalo i po 24 sličice za jednu sekundu animiranog filma.

Sada kada smo definirali i objasnili animaciju, možemo se vratiti na stop motion animaciju. U ovoj animatorskoj tehnici zapravo koristimo fotografije objekata koje postavljamo ispred objektiva. Pri tom je važno da je razlika između pojedinih fotografija što manja, odnosno da je pomak objekata ispred objektiva minimalan. Brzi prikaz tih fotografija rezultira stop motion animacijom.

Pogledajmo sljedeći niz fotografija koje sačinjavaju jednu sekvencu animacije.



Slika 6.1: dio fotografija iz stop motion animacije Od bita do robota

Pomak objekata koji je izvršen između uzimanja pojedine fotografije je minimalan. Rezultat prikaza ovih fotografija bit će privid samostalnog gibanja kockica.

Proces nastajanja stop motion animacije možemo ukratko opisati na sljedeći način: postavite objekt, fotografirate ga, minimalno ga pomaknete i ponovno fotografirate. Taj postupak ponavljate sve dok objekt ne pomaknete duž željenog puta.

6.2 Povijest

Povijesno gledano, teško je definirati vrijeme u kojem je započeo razvoj animacije, posebice jer se može ustvrditi da prvi pokušaji animacije datiraju još iz starijeg kamenog doba, kada su ljudi u pećinama prikazivali životinje s različitim položajima nogu pokušavajući ostvariti dojam kretanja. Intenzivniji razvoj animacije započinje tek početkom 19. stoljeća kada nastaju prve igračke i slikovne knjige koje izazivaju iluziju pokreta, a razvojem filmske industrije bilježi se i veliki napredak animacije.

Općenito gledano za animaciju možemo reći kako je starija od filma. Stop motion animacija stara je gotovo kao i film. Orginalno, pod nazivom stop motion animacije podrazumijevale su se sve animacije (animiranje igračaka, kocaka i svih drugih predmeta) izuzev klasične crtane animacijze. Razvojem celuloidne animacije razvila se i glinena i lutkarska stop motion animacija. Začetke stop motiona možemo vidjeti u filmovima The Humpty Dumpty Circus (1898.) i Fun in a Bakery Shop (1902.). Već 1907. godine u kinima se prikazivao prilično zapažen film The Haunted Hotel realiziran ovom tehnikom. Prvi glineni stop motion film prikazan je 1912. godine, a 1916. prva žena animatorica počinje s izradom svog prvog stopm motion filma (glinena animacija).

Današnja stop motion animacija značajno se razlikuje od prvih uradaka s početka 20. stoljeća. S razvojem tehnologije i 3D animacije, animiranje 3D likova na filmu postalo je mnogo lakše, brže i jeftinije, pa su danas rijetki filmovi snimljeni stop motion tehnikom. Tom se tehnikom uglavnom snimaju filmovi namijenjeni djeci ili filmovi koji se iz određenih razloga snimaju upravo tom tehnikom (reklame, glazbeni spotovi i sl.). No, iako se tehnologija razvija svjetlosnom brzinom, tehnika stop motiona nije napuštena, štoviše i u tom području dolazi do tehnoloških unaprijeđenja. Razvojem digitalne fotografije i fotoaparata, te softvera za obradu fotografije, montažu i animiranje značajno je olakšana i ubrzana izrada i stop motion animacija. Tako je 2005. godine snimljen prilično popularan animirani film Corpse Bride.

6.3 Vrste stop motion animacije

Postoji mnogo vrsta stop motion animacije, uglavnom imenovanih ovisno o mediju koji se koristi za kreiranje animacije. Tako razlikujemo lutkarsku stop motion animaciju koju odlikuje posebno izrađene lutke i makete okoliša (kuće, priroda i slično). Lutkice obično imaju žičani kostur obložen drugim materijalima. Primjer ovakve vrste animacije je film The Nightmare Before Christmas (1993.). Lutkama je moguće manipulirati u određenim zglobovima. Postoji i podvrsta lutkarske stop motion animacije gdje se za svaki kadar koristi druga lutka, umjesto da se jedna lutka za svaki kadar preoblikuje.

Osim lutkarske zastupljena je i glinena ili plastelinska stop motion animacija, koja koristi likove, tj. figure izrađene od gline, plastelina ili nekog drugog sličnog materijala. Figure mogu i ne moraju imati žičani kostur. Primjer ove vrste animacije je film Wallace and Gromit (1989.).

Kolažna animacija ili animacija izrezotina podrazumijeva pomicanje, odnosno animiranje dvodimenzionalnih komada papira ili tkanine. Primjer ove animacijske tehnike je Monty Python's Flying Circus (1969. – 1974.). Podvrsta kolažne animacije je siluetna animacija u kojoj se izrezani komadi osvjetljavaju pozadinskim svjetlom te su vidljive samo njihove siluete. Primjer filma snimljenog ovom tehnikom je The Adventures of Prince Achmed (1926.).

Animacija modela odnosi se na stop motion animaciju napravljenu u vidu dopune za druge pokretne slike. Najbolji primjer je film King Kong (1933.). Podvrsta animacije modela je „go motion" animacija u kojoj se različitim tehnikama postiže efekt zamućenja između kadrova u filmu.

Animacija objekata oživljava nepokretne objekte unutar stop motion animacije i razlikuje se od ostalih tehnika jer objekti nisu specijalno izrađivani za potrebe animacije. Podvrste animacije objekata su grafička animacija u kojoj se koriste grafički materijali (fotografije, isječci iz novina, časopisi i sl.) bilo da se pomiče materijale dok je kamera statična ili obrnuto i animacija kockica u kojoj se koriste Lego kockice ili slične igračke za postizanje animacije.

Posljednja vrsta stop motion animacije koristi glumce čiji se pokreti bilježe fotografijom, a zatim spajaju u stop motion animaciju. Primjer je kratki film Angry Kid.

6.4 Izrada stop motion animacije

Za svaki animirani film ili animaciju potrebno je prvo razviti ideju. Nakon inicijalne ideje vrlo je važan proces izrade storyboarda. Animacija je posao za strpljive i jako je bitna priprema rada. Kada u konačnici uvidimo da nam neki dio stop motion animacije nije dobar i želimo ga ponovno snimiti, jako je teško postići jednake uvjete (posebno ako ne radite u profesionalnom studiju), a razlika između prvih i ponovljenih fotografija uvijek je uočljiva. Osim toga, često je potrebno ponovno snimiti stotine fotografija što uvelike produžuje i otežava rad. Stoga, dobro planiranje je pola posla. Učenicima često pripremne faze nisu toliko zanimljive, ali im je potrebno dobro objasniti njihovu važnost. Pripremne faze imaju važnu odgojnu ulogu u razvoju radnih navika, ali i u razvoju kreativnosti.

Ideja, odnosno idejno rješenje početak je izrade svake animacije. U prvim vježbama iz stop motion animacije uvijek je dobro učenike na neki način ograničiti. Učenici pri prvom susretu s tehnikom i procesom animiranja ne poznaju mogućnosti same tehnike, stoga se ne može od njih očekivati da imaju lako izvedive ideje i da mogu osmisliti kvalitetnu priču za animaciju. Bitno je uvažiti svaku ideju, no za animaciju koristiti praktične i izvedive priče.

Prije početka samog rada poželjno je učenicima koji se prvi put susreću s ovom tehnikom pokazati što više primjera kako bi uvidjeli što se sve može postići ovom tehnikom i kako bi im sinule razne ideje. Pri tom je dobro pokazati im i par profesionalnih uradaka, no i one amaterske, kako se ne bi obeshrabrili i u konačnici bili nezadovoljni svojim radom.

Izrada animacije kreativan je, ali i dosta složen posao, pa je praktično podijeliti učenike u parove ili manje skupine (maks. 3 – 4 člana). Ovisno o uzrastu učenika i broju članova skupine može se odrediti potrebno vrijeme trajanja animacije. Prve animacije ograničite na vrijeme trajanja 15 do 30 sekundi. Animacija u tom trajanju zahtjeva od 180 do 360 fotografija (fps 12).

Kada govorimo o za početnike pogodnoj vrsti stop motion animacije obično se radi o animaciji objekata, pa je zgodno odabrati kockice, bombone ili nešto slično kao rekvizite za animiranje. Oni zahtjevaju relativno malo pripreme (nema oblikovanja gline, izrezivanja papira i sl.), lako je manipulirati njima i nije potreban velik prostor za pozornicu.

Kada ste ograničili temu, rekvizit i/ili trajanje animacije, prepustite učenicima daljnje osmišljavanje same animacije. Sljedeća faza je pisanje sinopsisa, a zatim crtanje storyboarda.

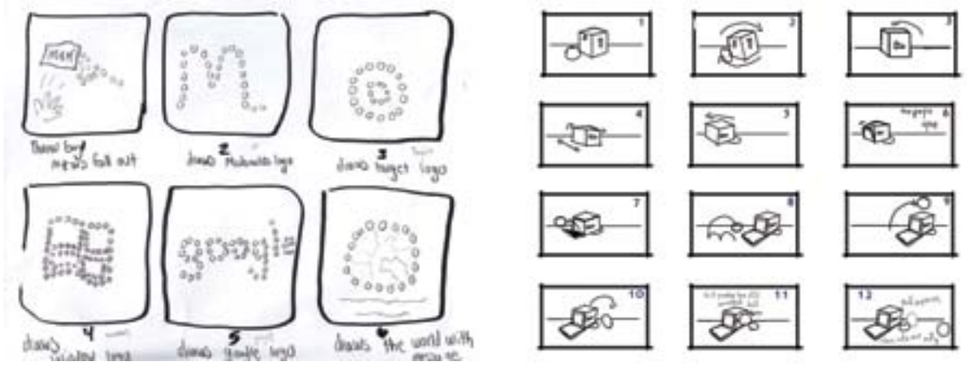
6.4.1 Sinopsis i storyboard

Sinopsis možemo definirati kao sažetak, odnosno sažeti tekstualni opis animacije, a ukoliko govorimo o animiranom filmu onda je to sažeti tekstualni opis radnje. Sinopsisom se obuhvaćaju sve bitne točke uvoda, zapleta i raspleta priče, bez prevelikih detalja. Ukoliko u animaciji postoji dijalog, on se ne navodi u sinopsisu, već u scenariju, tj. u storyboardu. Uloga sinopsisa nije tako značajna u kraćim animacijama. Korisno je radi upoznavanja cijelog procesa izrade animacije i uvoda u duže animacije i animirani film, učenicima objasniti pojam sinopsisa i pomoći im da ga napišu. Za kratke animacije i sinopsis je kratak, a ponekad se može svesti na jednu – dvije rečenice.

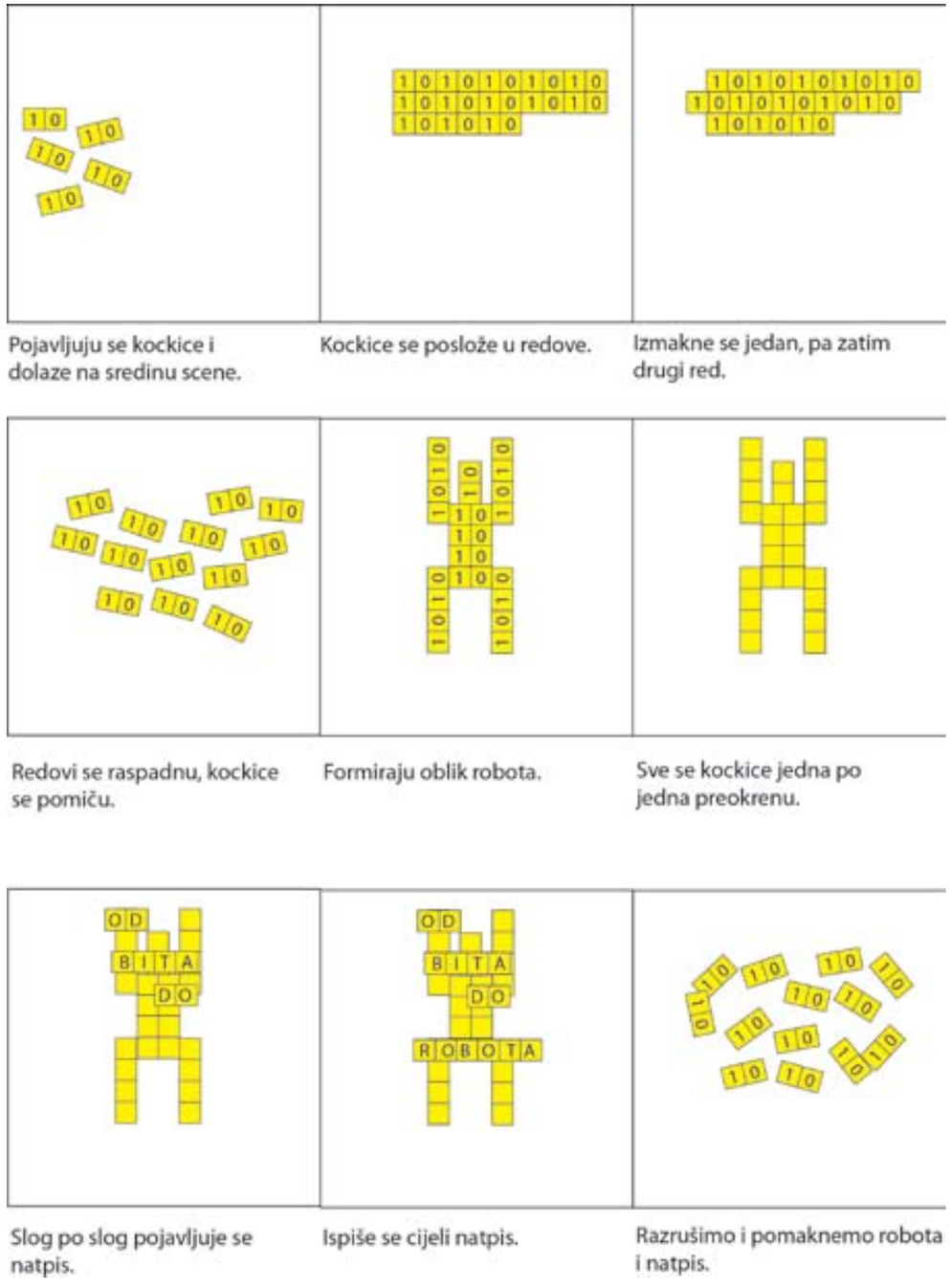
U filmskoj se industriji nakon pisanja sinopsisa pristupa pisanju scenarija. U svijetu animacije vrlo često se umjesto klasičnog scenarija koristi ilustrirani scenarij – storyboard. Jedan od najpoznatijih animatora, Walt Disney, osmislio je storyboard za potrebe izrade svojih animiranih filmova, a ubrzo se njegova primjena uvelike proširila. Uloga storyboarda je iznimno važna u pripremi animacije. Crtanjem niza sličica koje predstavljaju scene ili dijelove scena animacije učenici zapravo vizualiziraju tijek animacije, razrađuju ideju, uočavaju kritična mjesta (što je moguće, a što nije moguće realizirati) i postupno razrađuju ideju. Uz crteže često se navode tekstualne bilješke koje opisuju stvari i događaje na sceni i koje mogu sadržavati dijaloge ili napomene u vezi sa zvučnim efektima.

Velika prednost ilustriranog scenarija je mogućnost eksperimentiranja, mijenjanja priče, scena, kompozicije kadra i sl., uz mogućnost vizualizacije svega navedenog. Vrlo često učenici nemaju detaljno razrađenu priču i nisu svjesni tehničkih poteškoća pa putem storyboarda sve te probleme mogu i moraju riješiti. Sve neriješene poteškoće i nedostatke u priči prekasno je korigirati u fazi izrade, jer tada nemamo pregled nad cjelokupnim procesom pa nismo u stanju odrediti što i kako mijenjati, dok je u svim sljedećim fazama nemoguće ispraviti takve pogreške.

Storyboard učenici mogu crtati rukom ili na računalu, ovisno o njihovim željama i predznanju. Važno je učenicima naglasiti da u izradi storyboarda nije važna kvaliteta, tj. „ljepota" samog crteža, već njegova razumljivost i slijed. Storyboard može biti iznimno dobro nacrtan, kao pravi strip, ali to mogu biti i crteži likova sastavljeni od 4 do 5 linija. Pogledajte neke primjere storyboarda, te nacrtan storyboard za animaciju Od bita do robota koju možete pogledati na našem portalu www.e-skola.com.hr.



Slika 6.2: Primjeri storyboarda



Slika 6.3: Storyboard za animaciju Od bita do robota

Nakon što smo nacrtali storyboard potrebno je prikupiti potrebne rekvizite. Ovisno o vrsti stop motion animacije za koju smo se odlučili, trebamo pripremiti kockice, bombone, papire ili glinene, plastelinske, papirnate likove i ostale dijelove svih scena. U animaciji se mogu koristiti i razni drugi predmeti i materijali te njihove kombinacije. Osim pripreme likova važno je pripremiti i pozadine. U primjeru stop motion animacije rađenom za početnike korištene su žute plastične kockice na kojima smo ispisali brojeve 0 i 1. Za pozadinu je korišten bijeli stol.

Slika 6.4: Primjer korištenih objekata i pozadine za animaciju Od bita do robota

6.4.2 Pozornica

Prilikom izrade stop motion animacije važno je pronaći radni prostor koji treba biti dovoljno velik da se u njega može smjestiti pozornica, a u nju svi željeni likovi i pozadine. Vrlo često su to profesionalni studiji, no za školske potrebe pozornicu možemo izraditi u nekoj učionici, na radnom stolu, podu ili zidu. Snimanja, posebice snimanja višeminutnih filmova mogu trajati duži period. U idealnim okolnostima pozornicu bi bilo najbolje smjestiti u prostor koji je lako zamračiti te snimati uz upotrebu umjetnog osvjetljenja što obično zahtijeva barem osnovnu studijsku rasvjetu. Za pretpostaviti je da većina škola nema takve tehničke uvjete, pa ćemo se usredotočiti na druge načine snimanja. Pozornicu možete postaviti pored nekog prozora ili u prostor koji je dobro osvijetljen dnevnim svjetlom. Pri tom pazite na kut iz kojeg svjetlo pada na scenu, kako objekti složeni na sceni ne bi pravili nepotrebne sjene.

Slika 6.5: Studijska rasvjeta

Fotografije potrebne za animaciju u trajanju od 20 do 30 sekundi, pa i za animaciju od jedne minute, moguće je snimiti za nekoliko sati. U uvjetima u kojima nemate studijsku rasvjetu, planirajte snimanje za vrijeme dnevnog svjetla i potrudite se odraditi ga bez prekidanja.

Za prve vježbe, najpraktičnije je koristiti zid ili površinu radnog stola kao pozornicu. Na njima složite pozadinu, odnosno prvu scenu. Označite prostor unutar kojeg se nalazi pozadina, kako biste ako dođe do slučajnog pomicanja, sve potrebne elemente mogli vratiti na prvobitno mjesto. Vrlo je važno da kamera/fotoaparat prilikom snimanja miruje kako na fotografijama ne bi bilo pomaka cijele scene. Stoga, prilikom snimanja nikako ne držite kameru/fotoaparat u ruci. Poželjno je imati stativ i kameru postaviti na njega. Pritom označite položaj stativa ukoliko dođe do nehotičnog pomicanja kako biste ga mogli vratiti na početno mjesto. Ukoliko nemate stativ, kameru naslonite na stol te ju probajte postaviti na pravu visinu pomoću knjiga ili sličnih čvrstih i stabilnih predmeta. Ukoliko nemate mogućnost korištenja stativa i kameru ne možete učvrstiti na neki drugi način, pa je nužno da je netko drži u ruci, pokušajte je osloniti na neku stabilnu podlogu. Osoba koja drži kameru mora biti vrlo mirna i ne smije raditi velike pomake kamerom. U toj situaciji dobro je imati barem jednog ili dva dobro upućena asistenta koji mogu raditi izmjene na sceni vrlo brzo i precizno. Ovakva snimanja pogodna su samo kao uvod u animaciju, tj. za kratke animacije (10 – 15 sekundi).

Slika 6.6: Profesionalno studijsko snimanje stop motion animacije s profesionalnim pozadinama u studijskim uvjetima

Slika 6.7: Snimanje stop motion animacije pri dnevnom svjetlu

6.4.3 Snimanje

U fazi snimanja, ukoliko učenici rade u parovima ili u grupama, vrlo je važno svakom članu dodijeliti određene uloge. Snimanje je najosjetljivija faza procesa, pogotovo kada se radi u školskim uvjetima pa je podjela rada praktična i poželjna. Najčešće se jednog učenika zaduži za kameru i snimanje. Korisno je tom učeniku prethodno zadati da prouči rad konkretnog fotoaparata s kojim će raditi, sve njegove mogućnosti i načine snimanja. Druge učenike potrebno je zadužiti za pozornicu, tj. za pomicanje likova i objekata te za izmjenu pozadina na sceni. Precizno i brzo pomicanje objekata vrlo je bitno i uvelike olakšava i ubrzava rad pa je dobro da učenici uvježbaju pomake koje trebaju izvesti prije početka snimanja.

Obično snimanje započinjemo snimanjem prve scene, odnosno prvog kadra i nastavljamo slijedeći storyboard. Ukoliko se radi o složenijim animacijama, snimanje može početi i s nekom drugom, a ne s početnom scenom, sve ovisi o organizaciji rada i zahtjevima animacije.

Slika 6.8: Snimanje stop motion animacije u improviziranom okruženje

Prilikom snimanja važno je imati neka osnovna fotografska znanja pa ćemo ovdje objasniti dva osnovna pojma: ekspoziciju i kompoziciju. Ekspozicija u kontekstu fotografije označava ukupnu količinu svjetla koja kroz otvor objektiva pada na fotografski medij u nekom vremenu. S obzirom na to da ne postoji fotografski medij koji može zabilježiti raspon intenziteta svjetlosti kao ljudsko oko, potrebno je prilagoditi ukupnu količinu svjetla koja pada na medij. Jedinica ISO označava osjetljivost medija i ukazuje na sposobnost određenog medija da zabilježi prizore s malo svjetla. Snimatelj na raspolaganju ima potpunu slobodu u odabiru kombinacija otvora objektiva i vremena eksponiranja. No, to ne znači da su sve kombinacije prikladne za određenu situaciju. Pravilno eksponiranje označava postizanje cjelokupnog raspona svjetloće nekog motiva smještenog u ekspozicijski raspon materijala. Drugim riječima, pravilno eksponirati znači na fotografiji ostvariti što vjerniju reprodukciju tonova čime se postiže što vjernija slika stvarne scene. U kontekstu stop motion animacije važno je ovladati pravilnim eksponiranjem kako bismo dobili kvalitetne fotografije kao osnovu za animaciju. Fotografija može biti podeksponirana, pravilno eksponirana ili preeksponirana. Kao što sami izrazi kažu, podeksponirana fotografija nastaje ako na fotoosjetljivi medij nije palo dovoljno svjetla, a preeksponirana fotografija nastaje ako footosjetljivi medij ne može zabilježiti veliku količinu svjetla. Blago podeksponiranim ili preeksponiranim fotografijama mogu se postići određeni efekti ili naglasak na određenom raspoloženju u animaciji. Tako se podeeksponiranim kadrovima može stvoriti dojam tajnovitosti i mističnosti, a preeksponiranima osjećaj čistoće i vedrine.

Osim ekspozicije za stop motion animaciju važna je i kompozicija. Kada govorimo o kompoziciji fotografije, odnosno u kontekst stop motion animacije o kompoziciji kadra, okvir fotografije bit će definiran našom pozornicom. Kod fotografiranja je važno koji se elementi scene nalaze na fotografiji, a koji se izostavljaju, tj. kako će elementi koji čine sliku biti raspoređeni unutar okvira. Upravo to čini kompoziciju fotografije. Pomoću kompozicije snimatelj usmjerava pozornost na nešto bitno te se izražava vizualni sklad. Fotografija s horizontalnom kompozicijom izražava čvrstoću, mirnoću, stabilnost i prostornost, dok vertikalna kompozicija promatraču stvara osjećaj veličine, nadmoćnosti i dostojanstva.

Uz kompoziciju vežemo i pojam zlatnog reza. Pojam zlatni rez označava odnose koji se smatraju oku ugodnima s obzirom na proporcije i međusobne odnose objekata na fotografiji. U tom kontekstu treba spomenuti i ravnotežu, tj. simetriju na slici. Potpuna simetrija na fotografiji može djelovati dosadno, pretjerano statično ili nezanimljivo. S druge strane, neki su motivi sami po sebi statični i zahtijevaju od fotografa da poštuje navedenu statičnost. Zgodan je primjer zgrada sa središnjim vratima i po jednim prozorom s lijeve i desne strane vrata. Fotograf mora poštovati i zadržati takvu simetriju. No, nije dobro umjetno stvarati i forsirati simetriju na slici. Isto vrijedi i za svjetlo, boju, tonove, oštrinu i sve ostalo što može preusmjeriti našu pozornost. Iz tog razloga trebamo voditi računa i o pozadini i o planu, kao i o svjetlu i sjeni. Promatrač će prvo primijetiti sklad ili nesklad na fotografiji, a zadatak je snimatelja da kroz postizanje dobre kompozicije usmjeri pažnju promatrača na željeno mjesto. Kompozicija svake fotografije odnosno svakog kadra utječe na gotovu animaciju. Zato je praktično učenike usmjeriti da već pri izradi storyboarda razmišljaju o kompoziciji svakog pojedinog kadra.

6.4.4 Montaža

Montaža u filmskom žargonu označava postupak spajanja, odnosno sklapanja filma. Montažu možemo objasniti i kao postupak odabira i povezivanja kadrova u scene, scena u sekvence, a sekvenci u cjelovit film.

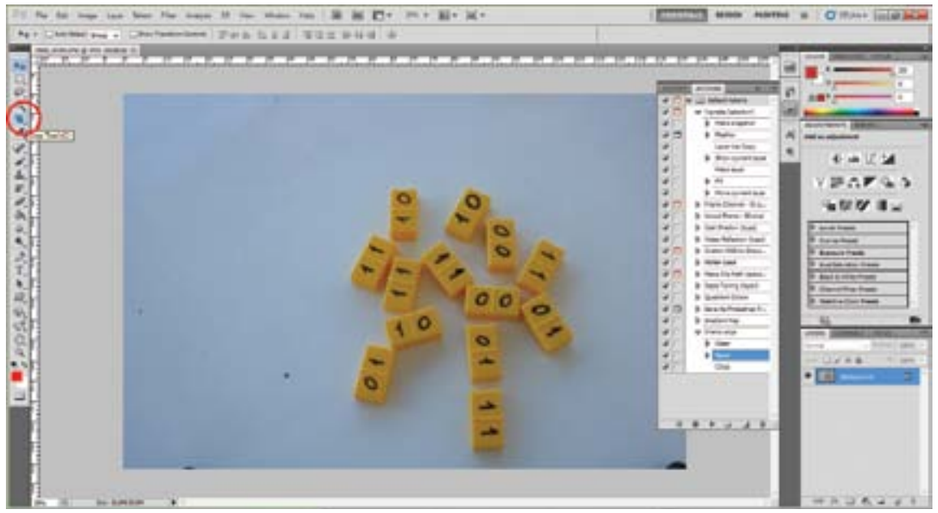
Kada govorimo o stop motion animaciji montaža podrazumijeva spajanje fotografija u animiranu cjelinu. Naravno, iskusniji montažer i u stop motion animaciji može primjenjivati različite montažne prijelaze i efekte, kako bi izrazio i naglasio određene ideje. No, kako je ovo poglavlje samo uvod u svijet stop motion animacije, ovdje ćemo objasniti isključivo osnovnu obradu fotografije i izradu animacije.

Ovdje navedena objašnjenja i upute odnose se na najpopularniji program za obradu fotografije: Adobe Photoshop. Slijedeći iste upute to je moguće izvesti i u nekom sličnom programu kao što je besplatni program za obradu fotografija Gimp. Osim tih programa moguće je besplatno skinuti i instalirati i razne druge programe već namijenjene izradi stop motion animacije. Kako nam je u cilju dotaknuti i objasniti osnove fotografije, ovdje ćemo se usredotočiti na program Photoshop.

Prije montaže pregledajte sve fotografije i izdvojite one koje nisu dobro uspjele ili su na bilo koji drugi način nepotrebne. Dakle, ako ste prilikom snimanja određene kadrove ponavljali, izaberite najbolju fotografiju, a ostale uklonite. Važno je da ne uklonite ključne fotografije, odnosno one čiji bi izostanak u animaciji bio vidljiv. To možete procijeniti i brzim pregledom fotografija u pregledniku. Otvorite prvu fotografiju i što je brže moguće prebacujte na slijedeću, čime ćete dobiti privid animacije.

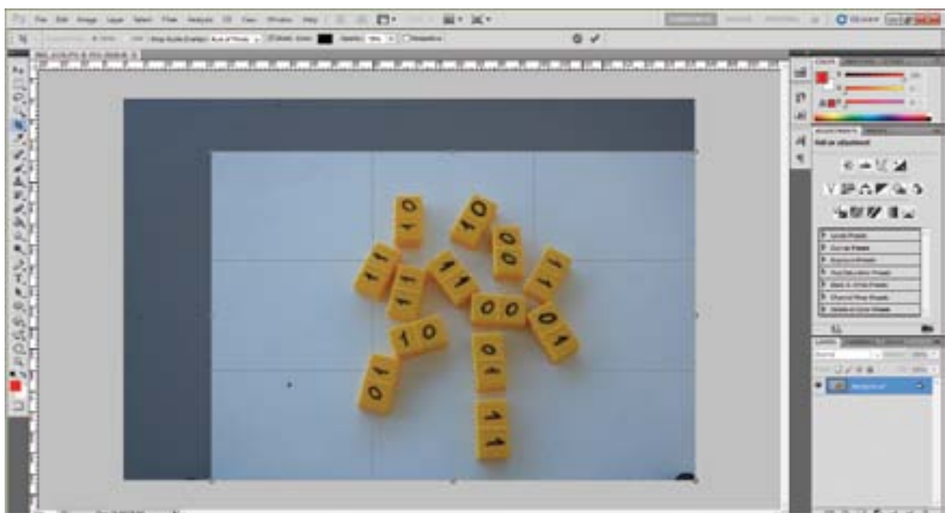
Nakon odabira fotografija, potrebno je izvršiti osnovne korekcije. Neke je fotografije potrebno obrezati (eng. crop) ili izvršiti osnovne korekcije boje. Također, efektno je ponekad (ovisno o temi animacije) sve fotografije prebaciti u crno-bijeli način ili na njih primijeniti neki drugi efekt ili filtar. Ovdje ćemo objasniti proces obrezivanja fotografija te automatskog popravljanja kontrasta i boje.

Kako biste obrezali fotografiju na željene dimenzije potrebno je kroz izbornik File>Open otvoriti željenu fotografiju. Nakon otvaranja fotografije odaberite alat „crop tool".



Slika 6.9: Odabir alata CROP u Photoshopu

Dobro je odrediti jednu točku na fotografiji kao referentnu točku i od tud početi obrezivanje. To može biti primjerice donji desni kut. Nakon što se odabrali alat crop pokazivač miša smjestite u željenu točku i povucite, dio fotografije koji niste obuhvatili alatom će se zatamniti. Pritisnite enter kako biste potvrdili obrezivanje fotografije.

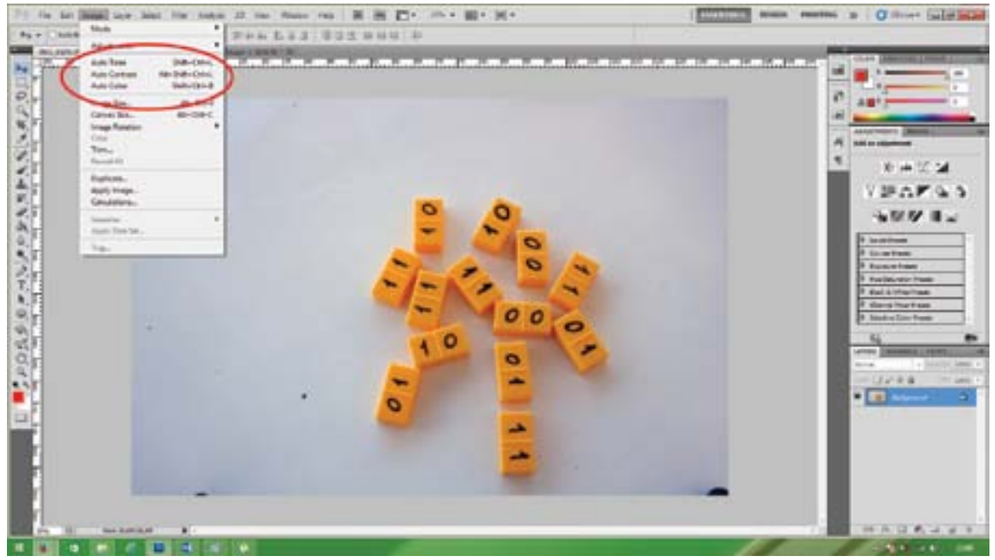


Slika 6.10: Obrezivanje fotografije u Photoshopu

Ako želite sve fotografije obrezati na jednake dimenzije dok je aktivan alat crop na alatnoj traci možete unijeti željene dimenzije te zatim alat primijeniti na fotografiju.

Slika 6.11: Definiranje područja obrezivanja fiksnih dimenzija u Photoshopu

Boju i kontrast moguće je popraviti primjenom naredbi Image>Auto Contrast, Image>Auto Tone i Image>Auto Color.

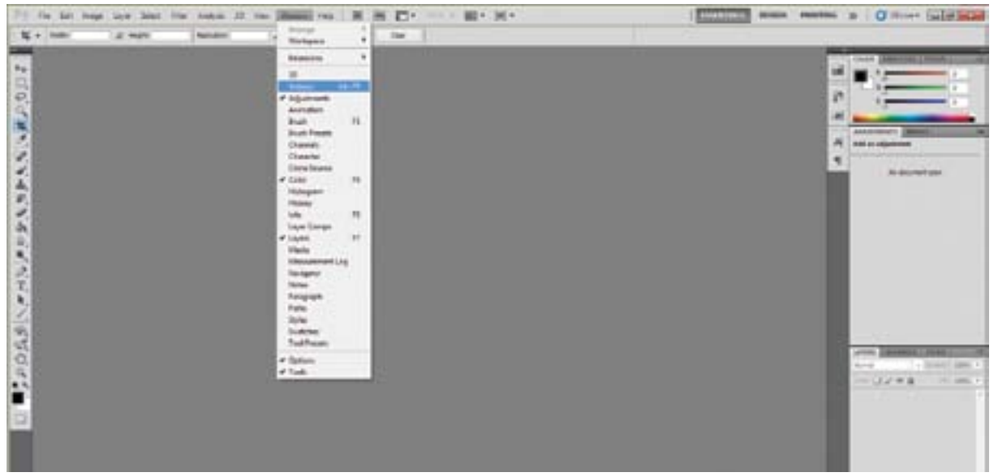


Slika 6.12: Prikaz naredbi za automatsku korekciju tona, boje i kontrasta u Photoshopu

Nakon obrade svih fotografija potrebno ih je imenovati na način da im je u imenu sadržan broj, i da ti brojevi slijede stvarni poredak fotografija (npr. Imefotke_001 do Imefotke_540). Vrlo često to neće biti slučaj. I iako je ponekad potrebno većinu fotografija obrađivati pojedinačno, imenovanje se može odraditi i postupkom automatizacije u Photoshopu.

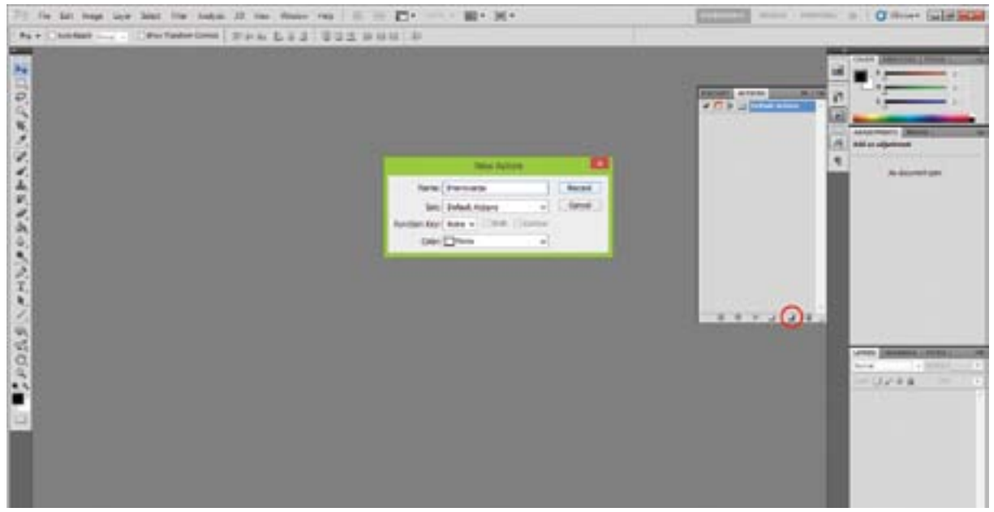
Ako imate više učenika u grupi, a relativno mali broj fotografija (za kratku animaciju) možete im i podijeliti zadatke pa da svatko preimenuje dio fotografija. U tom slučaju važno je da netko prekontrolira sva imena jer često dođe do pogreške.

Brži i učinkovitiji način je provesti to automatski u Photoshopu.Nakon otvaranja programa potrebno je otvoriti paletu Actions putem izbornika Windows > Actions.



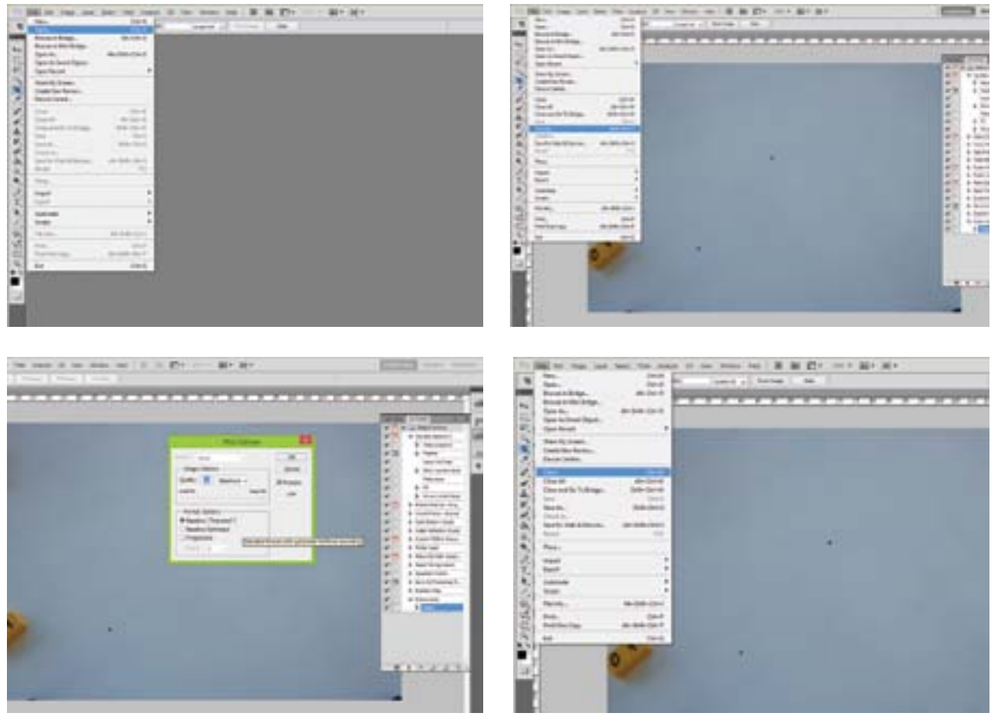
Slika 6.13: Aktiviranje palete Actions u Photoshopu

U paleti Actions potrebno je kreirati novu akciju za preimenovanje imena fotografija. Klikom na gumb New (u crvenom kružiću) otvara se dijaloški okvir New Actions.



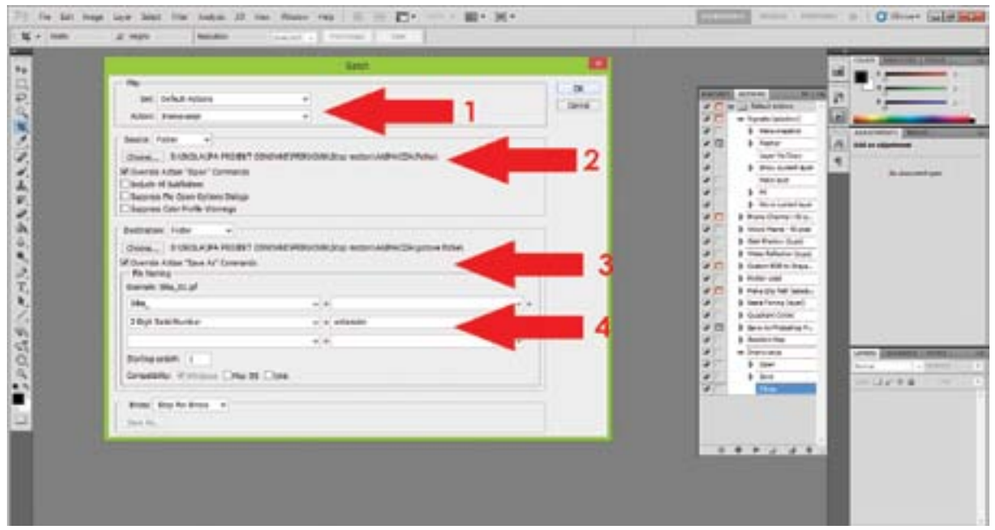
Slika 6.14: Kreiranje novih akcija u Photoshopu

U otvorenom dijaloškom okviru potrebno je imenovati vaše akcije, a zatim ih snimiti. Automatski će se uključiti način za snimanje i bit će vam aktivna crvena točka u dnu palete Actions. Zatim je potrebno provesti potrebne akcije. Kod preimenovanja fotografija potrebno je provesti otvaranje, spremanje i zatvaranje fotografija (kao na donjim slikama) i te akcije snimiti.



Slika 6.15: Provođenje snimanja potrebnih akcija kako bi se mogla provesti automatizacija preimenovanja fotografija

Kada smo izveli potrebne akcije potrebno je snimanje akcija zaustaviti klikom na gumb Stop na dnu palete Actions. Sada je potrebno primijeniti snimljene akcije na sve fotografije potrebne za animaciju. Odaberite File>Automate>Batch... i otvorit će vam se dijaloški okvir kao na slici.

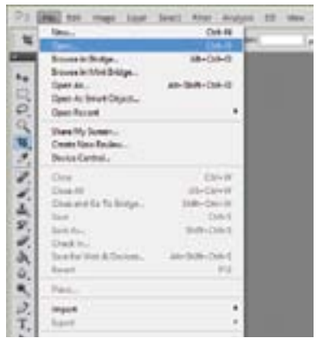


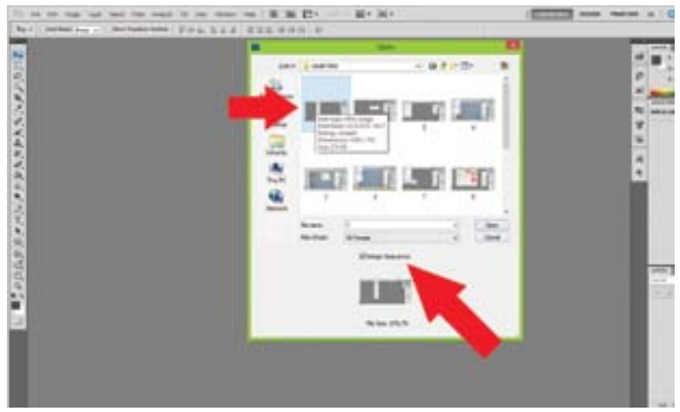
Slika 6.16: Provođenje aktivacije automatizacije u Photoshopu

Potrebno je unijeti naziv vaših snimljenih akcija (1), odrediti mapu u kojoj se nalaze spremljene fotografije (2), mapu u koju će se spremiti preimenovane fotografije (3), te definirati način imenovanja svih fotografija (4). Pri tom je važno označiti kućice "Override action Open" i „Override action Save as". Detaljnije upute o procesu automatizacije potražite na portalu www.e-skola.com.hr.

Jednom kada ste preimenovali fotografije i dodali im brojeve nemojte brisati fotografije iz niza, jer ćete morati preimenovati ponovno sve fotografije. Sve fotografije spremite u jednu mapu (u toj mapi ne smije biti ništa osim fotografija koje će se koristiti za animaciju) i ponovno otvorite Photoshop.

Kroz izbornik File>Open odaberite mapu u kojoj se nalaze vaše fotografije i kliknite na prvu fotografiju (onu s najmanjim brojem) i zatim označite kućicu "Image Sequence" na dnu dijaloškog okvira. Stisnite gumb „Open".



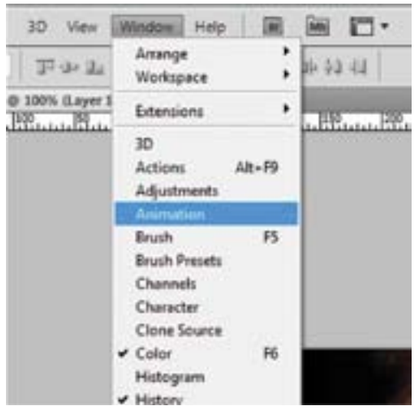


Slika 6.17: Provođenje aktivacije automatizacije u Photoshopu

U novom prozoru unesite frame rate – 12.

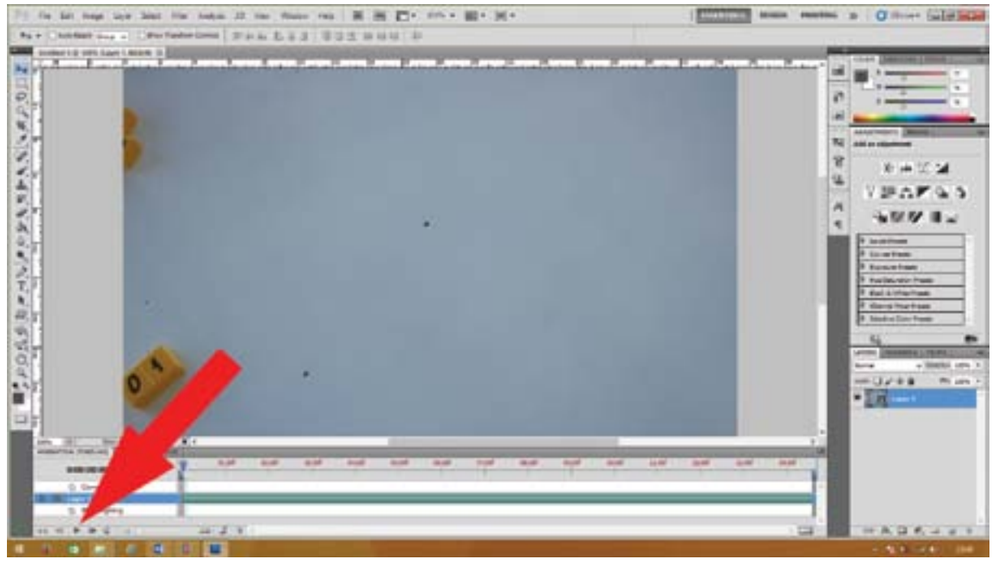
Slika 6.18: Definiranje „frame ratea" u Photoshopu

Zatim u izborniku Window otvorite paletu Animation.



Slika 6.19: Aktiviranje palete Animation u Photoshopu

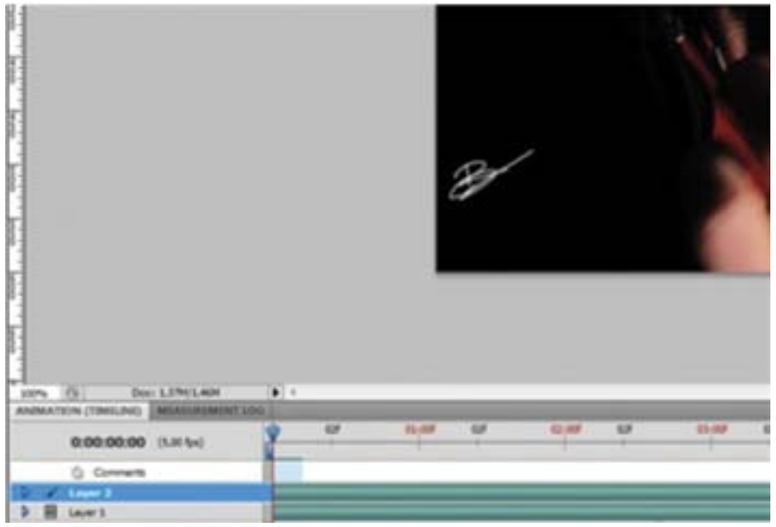
U paleti Animation kliknite na gumb Play kako biste pregledali animaciju.



Slika 6.20: Provođenje reprodukcije animacije u Photoshopu

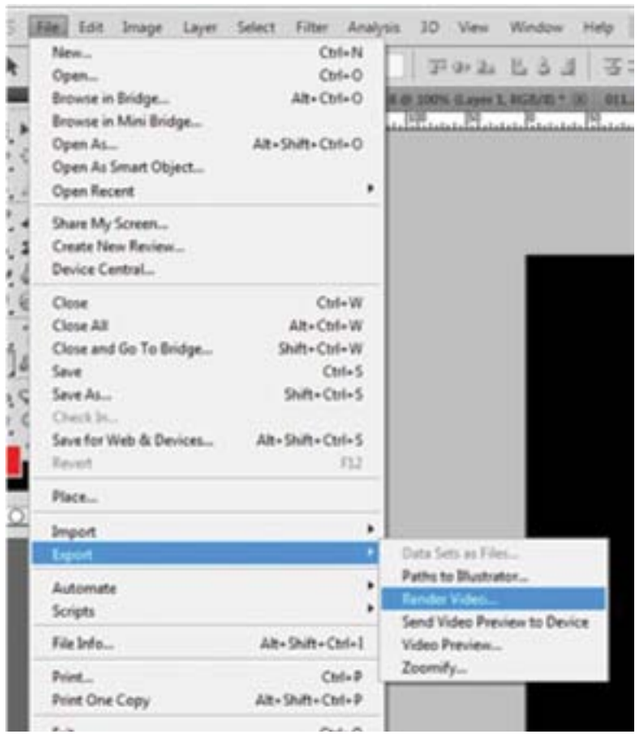
Ako želite u Photoshopu, možete dodati određene efekte u određene dijelove animacije na način da kreirate novi sloj i u njemu izradite željeni sadržaj (npr. tekst, efekte, dijelove neke druge fotografije i sl.)

Trajanje prikaza novog sloja možete definirati tako da mišem, u paleti Animation dođete na početak ili kraj trajanja sloja i povučete strelice koje vam se prikažu.



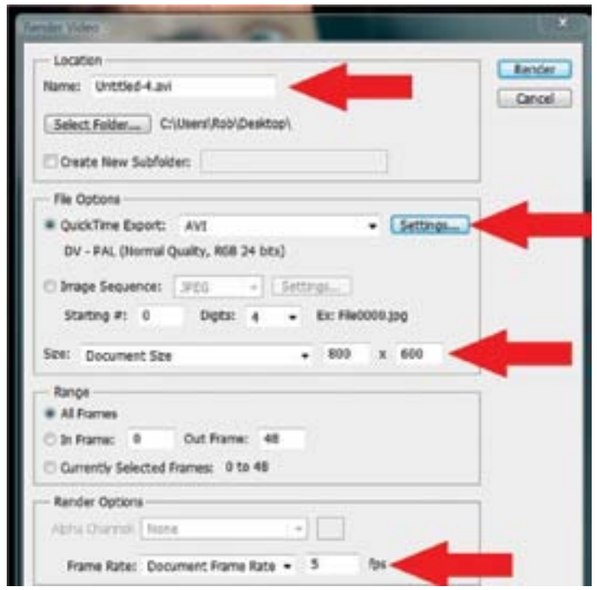
Slika 6.21: Upravljanje trajanjem slojeva unutar animacije

Kada ste gotovi s uređivanjem, animaciju je potrebno izvesti, što provodite putem izbornika File > Export > Render Video.



Slika 6.22: Izvoz gotove animacije

U novootvorenom prozoru potrebno je prvo imenovati vašu animaciju, odabrati mapu u koju ćete je spremiti, a zatim i format za spremanje datoteke. Od ponuđenih formata najpraktičnije je koristiti format .avi, jer ga podržava većina programa za reprodukciju animiranih i video uradaka. Prje nego završimo s izvozom animacije potrebno je u u izborniku Settings > Settings > Compression type odabrati "None" i "medium" kvalitetu, a zatim u dijaloški okvir „Size" unijeti veličinu vaših fotografija te na dnu prozora izabrati željeni "Frame Rate".



Slika 6.23: Određivanje postavki izvoza animacije

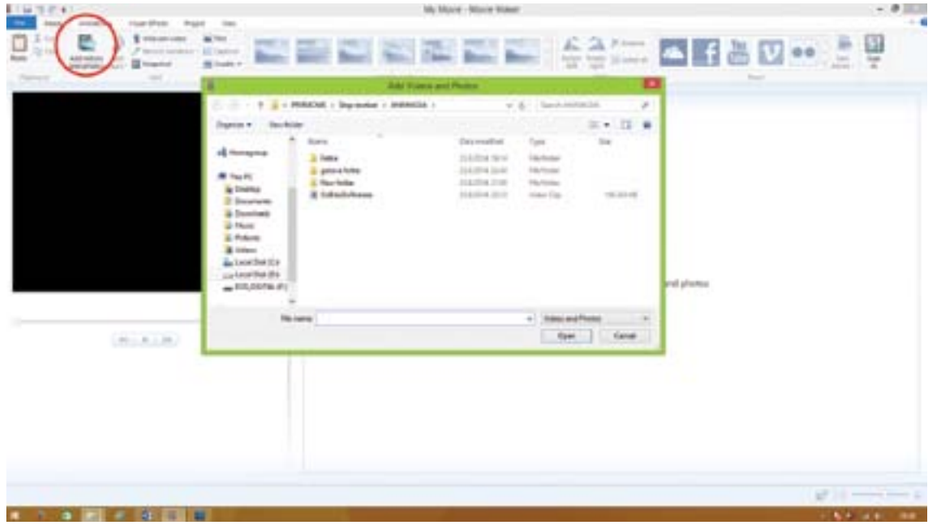
Kada ste unijeli sve potrebne podatke kliknite na gumb Render.

Vaša animacija je gotova!

Spremite orginal .psd dokument (File> Save as) kako biste mogli unijeti određene promijene ako poželite. Detaljnije upute o izradi animacije potražite na portalu www.e-skola.com.hr.

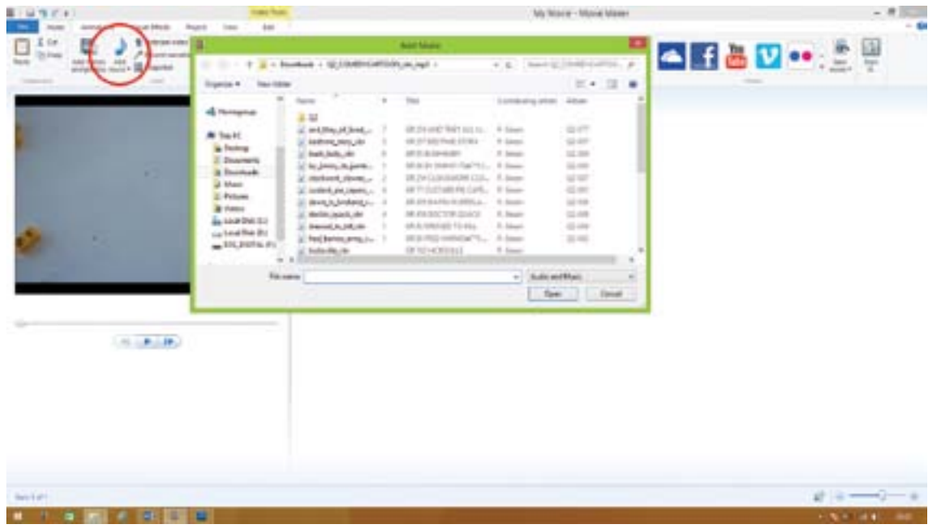
6.4.5 Dodavanje zvuka

Nakon što ste gotovu animaciju izvezli iz programa Photoshop, dodavanje zvuka možete provesti u različitim programima, od onih za obradu animacije do programa za obradu videa. Za školske potrebe najpraktičnije je to provesti u programu Windows Movie Maker, s kojim se osnovnoškolski učenici ionako susreću, a besplatan je za korištenje. Otvorite program Windows Movie Maker te uvezite animaciju klikom na gumb Add videos and photos.



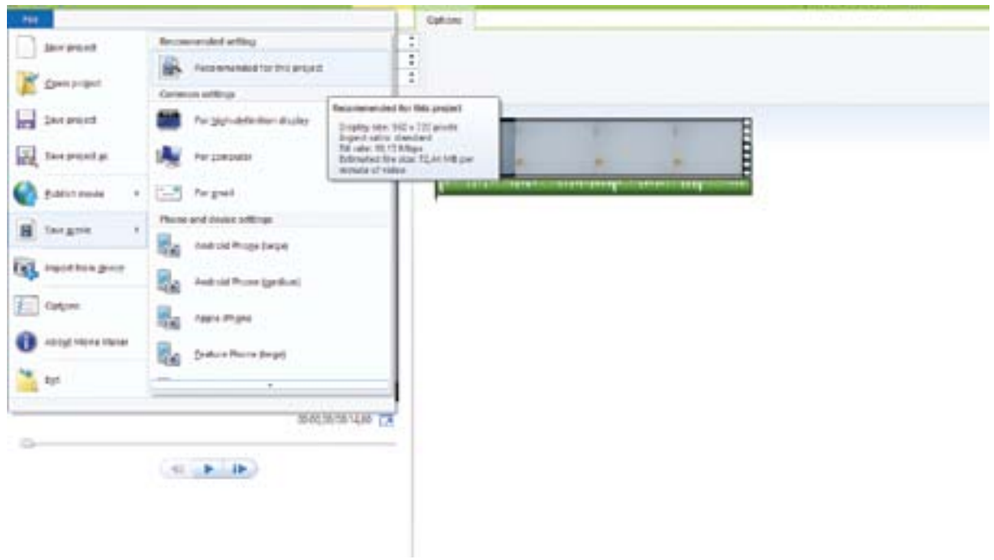
Slika 6.24: Uvoz animiranog uratka u Windows Movie Maker

Zatim uvezite zvuk klikom na gumb Add music.



Slika 6.25: Uvoz zvučnih zapisa u Windows Movie Maker

Zatim gotov uradak izvezite odabirom naredbe File>Save movie>Recommended for this project.



Slika 6.26: Izvoz gotove animacije

U Movie Makeru možete dodavati i titlove, naraciju ili neke druge efekte. Također možete spajati više animacija u jednu cjelinu, no ovime je vaša animacija prva stop motion animacija potpuno završena. Pregledajte je u nekom video pregledniku i obvezno izvršite postavljanje na neki Internet video servis (npr. Youtube), time ćete dodatno motivirati učenike jer će njihov rad biti dostupan javnosti.

Further development and implementation of the Croatian Qualifications Framework
/ **Daljnji razvoj i provedba hrvatskog kvalifikacijskog okvira**
Lot 1: Development of school curricula for general education system based on learning outcomes
/ Lot 1: **Razvoj školskih kurikula za opći obrazovni sustav temeljen na ishodima učenja**
Project is co-financed by the European Union from the European Social Fund
/ **Projekt je sufinancirala Europska unija iz Europskog socijalnog fonda**

IT skills for successful and creative future/ Informatička znanja za uspješnu i kreativnu budućnost
Acronym / **Akronim:** (ICT4SCF)
Total costs / **Vrijednost projekta:** 204.224,23 Eur
Reference: Europeaid/131254/M/ACT/HR
Contract No. / **Broj ugovora:** IPA4.1.3.1.06.01.c08

The content of this publication material are the sole responsibility of Primary school Eugen Kumičić, Slatina / Sadržaj ove publikacije isključiva je odgovornost OŠ Eugena Kumičića, Slatina

Project beneficiary / **Korisnik:** Osnovna škola Eugena Kumičića
Dobriše Cesarića 24, Slatina
Tel. Num /Tel. br. . +385 33 551 213
Fax. num /Br. Faxa: +385 33 400 086
Web site / web stranica: http://os-ekumicica-slatina.skole.hr

Contact informatiofor relevant institutions from the EU fund management system:

Ministarstva znanosti, obrazovanja i sporta
www.mzos.hr ; e-mail: esf@mzos.hr
Ministarstvo rada i mirovinskog sustava
www.mrms.hr ; e-mail: info@mrms.hr
Ministarstvo socijalne politike i mladih
www.mspm.hr ; e-mail: ministarstvo@mspm.hr
Agencija za strukovno obrazovanje i obrazovanje odraslih
Organizacijska jedinica za upravljanje strukturnim instrumentima (DEFCO)
www.asoo.hr ; e-mail: defco@asoo.hr

For further information on EU funds contact:

Ministarstvo regionalnog razvoja i fondova Europske unije
www.mrrfeu.hr ; fondovi@mrrfeu.hr
Strukturni i investicijski fondovi
www.strukturnifondovi.hr

Parties / **Partneri:**
1. OŠ Dobriše Cesarića, Zagreb
2. OŠ Ivana Granđe, Soblinec
3. OŠ Otona Ivekovića, Zagreb
4. Grafička škola u Zagrebu, Zagreb


ICT4SCF