

Logo Korak dalje

Pišemo kod kornjači

Druga knjiga programiranja

```
repeat 4 [  
  fd 100  
  rt 90  
]
```



```
make "duljina 5  
repeat 30 [  
  fd :duljina  
  rt 30  
  make "duljina :duljina + 3  
]
```



Željko Udovčič

Sadržaj

1	Logo: Korak dalje	1
I	Od blokova do koda	3
2	Od blokova do koda	4
3	Zašto pisati kod?	5
4	Prozor za kod	9
5	Tvoja prva naredba	13
II	Kretanje	19
6	Kretanje	20
7	Naprijed i natrag	21
8	Lijevo i desno	26
9	Skraćenice naredbi	30
10	Probaj sam!	33
III	Oblici	37
11	Oblici	38
12	Ravna crta	39
13	Kvadrat korak po korak	45
14	Trokut	51
15	Kuća	58
IV	Ponavljanje	63
16	Ponavljanje	64
17	Naredba repeat	65
18	Mnogokuti	74
19	Zvijezde	82
20	Ugniježđeni repeat	90
V	Boje i olovka	97
21	Boje i olovka	98
22	Olovka gore i dolje	100
23	Boja crte	106
24	Debljina crte	115
VI	Varijable i računanje	122
25	Varijable i računanje	123
26	Što je varijabla	124
27	Naredba make	129
28	Računanje	134
29	Popis u kutijici	142
VII	Procedure	151
30	Procedure	152
31	Što je procedura	153

32	Prva procedura	159
33	Procedura s parametrima	165
VIII	Uvjeti	177
34	Uvjeti	178
35	Naredba if	179
36	Usporedbe	186
37	Slučajna šetnja	193
IX	Projekti	198
38	Projekti	199
39	Projekt: Mnogokutni cvijet	200
40	Projekt: Stablo (fraktal)	204
41	Projekt: Spirala duga	210
42	Što dalje?	217
	Podsjetnik	220

Poglavlje 1

Logo: Korak dalje

Dobrodošao natrag, programeru! Tina, naša kornjača iz prve knjige, i dalje je s tobom.

Ako si već crtao s Tininim blokovima, znaš kako to ide: povučeš blok, spojiš ga s drugim, pritisneš **Pokreni** i Tina poslušno hoda po platnu. To je bila prva knjiga.

U ovoj drugoj knjizi učimo nešto novo i moćno: **pisati kod riječima**, baš kao pravi programeri. Umjesto da povučeš blok **forward**, upisat ćeš forward 100 (ili kraće, samo prvo i zadnje slovo riječi: fd 100) i Tina će krenuti.



1.1 Što ćeš naučiti

Knjiga je podijeljena u devet manjih dijelova. Svaki donosi novi alat u tvoju programersku torbu:

- Kako od blokova prijeći na **pisani kod**
- Sve naredbe za **kretanje** Tine napisane riječima
- Kako brže crtati **oblike** i **mnogokute**
- Petlju **repeat** koja zna ponavljati naredbe stotinu puta
- Kako biraš **boju** i **debljinu** crte
- **Variable**: kutijice u koje spremaš brojeve i s kojima računaš
- **Popise**: cijeli popis boja ili brojeva spremljen u jednu kutijicu
- **Procedure**: vlastite naredbe koje sam izmišljaš
- **Uvjete** s if, pomoću kojih Tina donosi odluke
- **Projekte** u kojima sastavljaš sve što si naučio

Mnogo toga već znaš iz prve knjige kao blokove: ponavljanje, boje, kutijice, vlastite blokove i popise. Sad ćeš iste ideje pisati kodom. Novo će biti **računanje** s varijablama i **odlučivanje** s if, baš kako je obećala prva knjiga.

1.2 Kako koristiti ovu knjigu

Knjiga je napravljena tako da je čitaš redom, ali ako te neka tema više zanima, slobodno preskoči naprijed. Najvažnije je da **isprobavaš svaki primjer**. Programiranje se ne uči gledanjem, nego pisanjem koda i gledanjem što se dogodi.

Greške su ok!

Kad nešto pogriješiš, konzola će ti napisati što ne valja. To je dobro! Svaki programer (i tvoj učitelj, i autor ove knjige) griješi svaki dan. Greške su način da naučiš nešto novo. Poruke su na engleskom, ali knjiga će ti objasniti što znače.

1.3 Što trebaš znati prije

Najbolje je da si već prošao prvu knjigu (**Logo za početnike**, *Crtanje s blokovima*). Ako nisi, ne brini, jer ova knjiga objašnjava sve od početka. Samo ćeš trebati malo više vježbe.

Trebao bi znati ovo:

- Tko je Tina i što je platno
- Razliku između **naprijed** i **okreni se**
- Kako otvoriti novi projekt u Blocks Code Studiju
- Kako blok **repeat** ponavlja druge blokove i kako olovci mijenjaš **boju** i **debljinu**
- Što su **čarobne kutijice**, **vlastiti blokovi** i **popisi**

Ako nešto od toga zaboraviš, kratko ćemo se na to vratiti kad zatreba.

Spreman? Idemo upoznati prozor za pisanje koda!

Sljedeće: [Od blokova do koda →](#)

I. dio

Od blokova do koda

Poglavlje 2

Od blokova do koda



U prvoj knjizi si Tini davao upute slaganjem **blokova**. Sada ćeš joj iste te upute pisati **riječima**. Zvuči teže? Zapravo nije. Kad jednom upamtiš nekoliko naredbi, pisanje je **brže i fleksibilnije** od slaganja blokova.

U ovom dijelu ćemo:

- Vidjeti **zašto** programeri uopće pišu kod umjesto da koriste blokove
- Upoznati **prozor za kod** u Blocks Code Studiju
- Napisati Tini našu **prvu naredbu** i pokrenuti je

Kad završiš ovaj dio, znat ćeš Tini reći “naprijed sto koraka” pisanim kodom umjesto blokom.

Sljedeće: [Zašto pisati kod? →](#)

Poglavlje 3

Zašto pisati kod?

Možda se pitaš: ako blokovi rade isto što i kod, zašto se onda truditi i učiti pisati?

Razlog je jednostavan: **kod je brži, kraći i moćniji.**

3.1 Isto, ali kraće

Pogledaj ova dva načina kako reći Tini da nacrtá kvadrat.

S blokovima trebaš **osam blokova**: četiri bloka **forward** i četiri bloka **right**. U prvoj knjizi taj si kvadrat složio ovako:



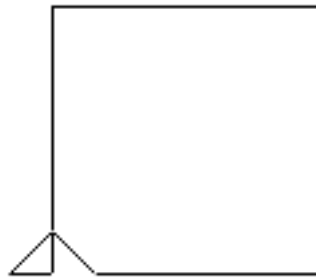
Pisanim kodom napišeš isto:

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

Osam redaka, ali ih tipkanjem napišeš brže nego što bi povukao osam blokova.

A s naredbom `repeat`, koju već znaš kao blok iz prve knjige, to postaje još kraće:

```
repeat 4 [ fd 100 rt 90 ]
```



Slika 3.1: Kvadrat: isti rezultat, jedan redak koda

Jedan redak! Naravno, i blokovi imaju blok **repeat**, ali u kodu možeš sve napisati u jednom retku, kopirati tekst, brzo mijenjati brojeve i lako ga slati prijateljima.

3.2 Brže pišeš

Kad jednom upamtiš nekoliko naredbi, pisanje koda postaje vrlo brzo. Tipke su brže od miša. Programeri koji rade cijeli dan najviše vremena provode upravo pišući kod, ne povlačeći blokove.

3.3 Možeš sve što i s blokovima, i još više

Sve što si naučio u prvoj knjizi možeš napraviti i kodom, od kretanja i ponavljanja do kutijica (varijabli), vlastitih blokova (procedura) i popisa. Kod ti daje i neke posebne

prednosti:

- **Brže čitanje:** veći programi su pregledniji kao tekst nego kao hrpa blokova
- **Lakše dijeljenje:** kod je običan tekst koji možeš kopirati, zalijepiti, poslati ili spremiti
- **Priprema za prave jezike:** svi ozbiljni programski jezici (Python, JavaScript, C++) koriste pisani kod, ne blokove

Sve to ćemo naučiti u ovoj knjizi.

3.4 Što ostaje isto

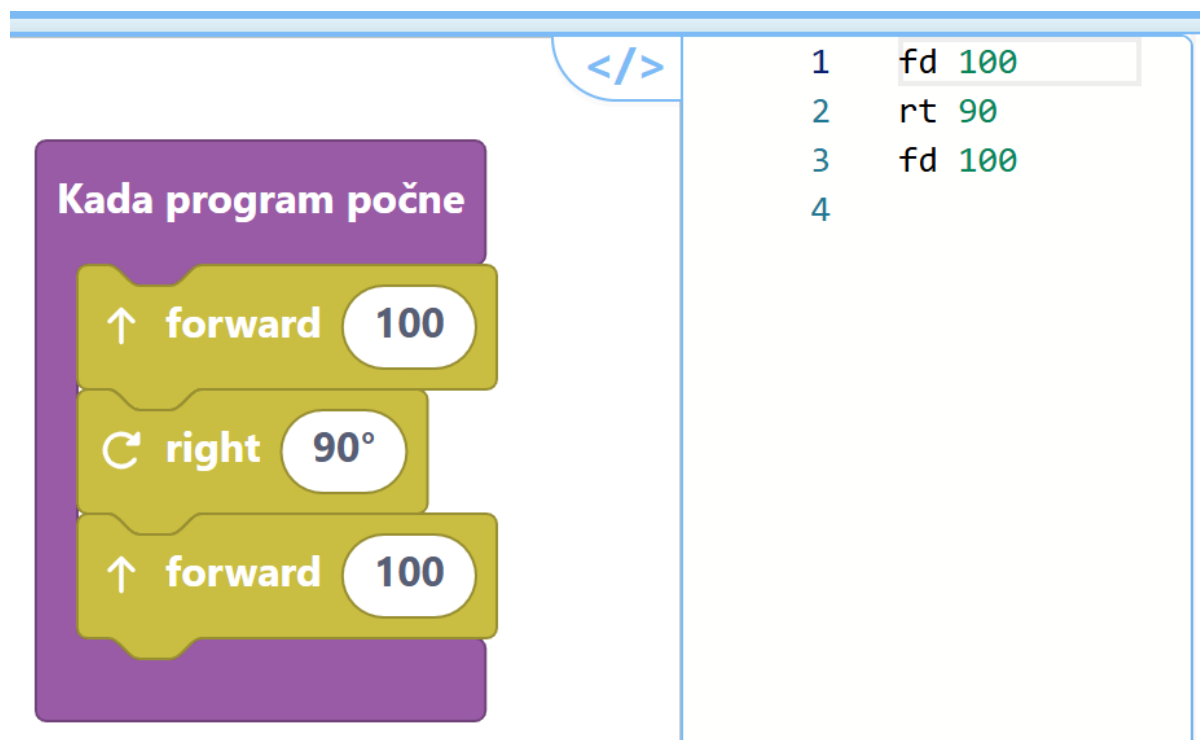
Dobre vijesti: **Tina je ista**. Platno je isto. Pravila su ista. `fd 100` znači potpuno isto što i blok **forward 100**: Tina će se pomaknuti naprijed za sto koraka i ostaviti crtu za sobom.

Usput, primijeti da na bloku piše cijela riječ: **forward**. I to je ispravna Logo naredba, `forward 100` radi potpuno isto kao `fd 100`. Ali programeri vole pisati kratko: od riječi `forward` uzmemo **prvo i zadnje slovo** i dobijemo `fd`, što je kraće i brže za tipkanje. Isti trik vrijedi i za okretanje: od `right` nastaje `rt`. Više o skraćenicama naučit ćeš u poglavlju [Skraćenice naredbi](#).

Mijenja se samo **kako joj govoriš**: umjesto blokova koristiš tipkovnicu.

3.5 Pogledaj kod iz blokova

Jesi li znao da Blocks Code Studio može **prikazati kod** koji odgovara tvojim blokovima? U projektu s blokovima (**Logo Blocks**) postoji gumb `</>`, u gornjem desnom uglu radne površine. On otvara prozor s Logo kodom koji nastaje iz tvojih blokova, pa vidiš kako bi tvoji blokovi izgledali napisani kao tekst. Primijetit ćeš da su naredbe ondje napisane kratko, kao `fd` i `rt`.



Slika 3.2: Blokovi i Logo kod koji iz njih nastaje, gumb </> otvara prozor s kodom

Taj kod možeš samo čitati, a mijenjaš ga slaganjem blokova. To je sjajan način da se navikneš na pisani kod: slažeš blokove kao i prije, a program sam pokaže kako to izgleda u kodu!

Možeš se uvijek vratiti

Ako negdje zapneš, otvori novi projekt s blokovima (**Logo Blocks**), složi isti crtež i pritisni </>. Usporedi kod koji vidiš sa svojim, pa ćeš lako provjeriti jesi li dobro razumio naredbu.

3.6 Kratak rječnik

Prije nego krenemo, evo nekoliko riječi koje ćeš često čitati:

- **naredba**: jedna riječ koja Tini govori što da napravi (fd, rt, repeat)
- **kod**: skup naredbi koji ti pišeš
- **program**: cijela “priča” koju si napisao Tini
- **pokreni**: reci računalu da izvrši tvoj kod (gumb  gore desno)

Sljedeće: [Prozor za kod →](#)

Poglavlje 4

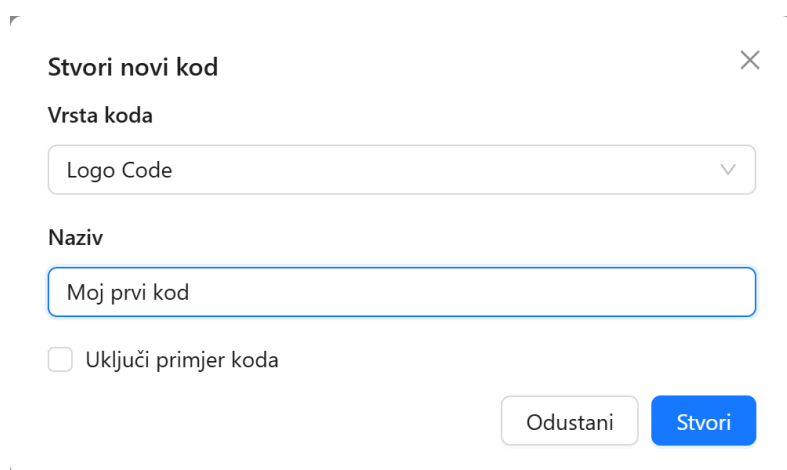
Prozor za kod

Da bismo počeli pisati Logo kod, trebamo otvoriti **prozor za kod** u Blocks Code Studiju. To je tekstualno polje u koje upisujemo naredbe.

4.1 Otvaranje novog projekta

Postupak je gotovo isti kao za blokove u prvoj knjizi. Razlika je samo u vrsti koda i jednoj kvačici:

1. Otvori studio.blocks-code.com u pregledniku.
2. Pritisni mali gumb **+ Kreiraj** na vrhu kartice **Lokalna pohrana preglednika**. Ako si prijavljen, isti gumb imaš i na kartici **Oblak**.
3. Otvorit će se prozorčić **Stvori novi kod**. U polju **Vrsta koda** odaberi **Logo Code** (ne Logo Blocks).
4. U polje **Naziv** upiši ime, npr. **“Moj prvi kod”**.
5. Makni kvačicu **Uključi primjer koda** da dobiješ prazan uređivač. Preglednik će zapamtiti tvoj izbor za sljedeći put.
6. Pritisni **Stvori**.

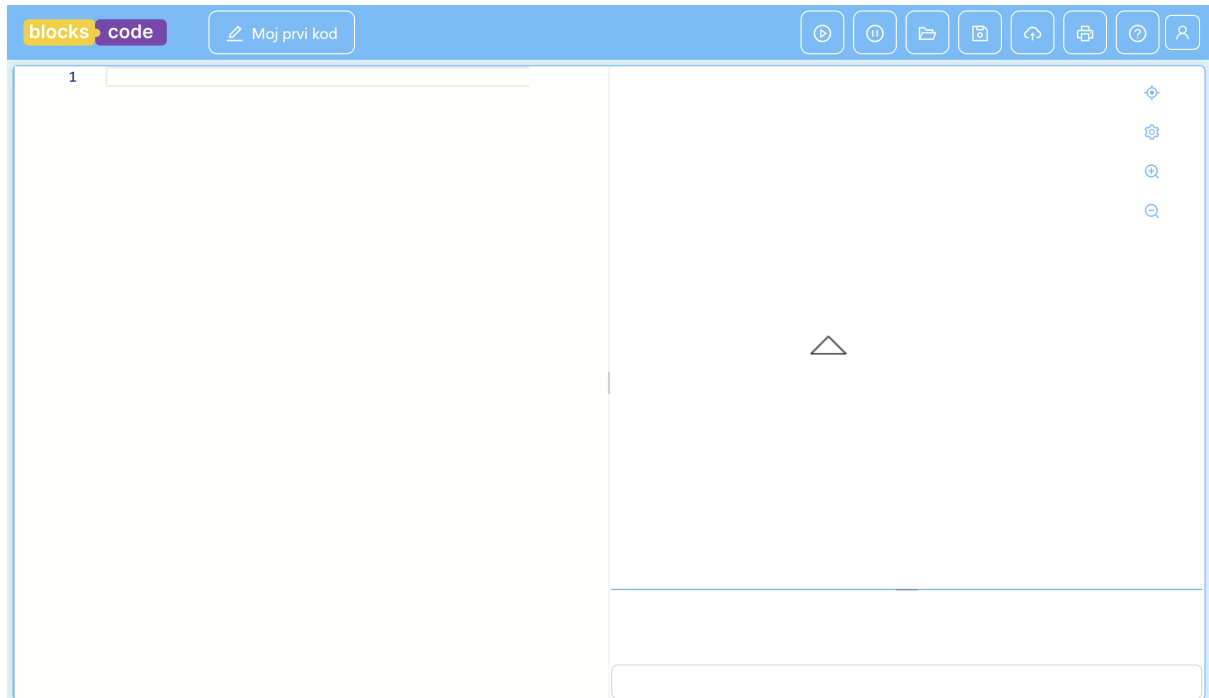


Otvorit će se novi izgled radne površine. Umjesto ladice s blokovima, sada vidiš **uređivač teksta**: prazno polje s brojevima redaka uz lijevi rub.

Ako je kvačica **Uključi primjer koda** ostala uključena, u uređivaču je već primjer programa koji se sam pokrene čim se projekt otvori. Ništa strašno: pogledaj što Tina

nacrta, a onda obriši primjer prije nego kreneš pisati svoj kod.

4.2 Što je gdje



Slika 4.1: Radna površina prozora za kod: uređivač lijevo, platno desno, konzola dolje desno

Prozor za kod ima tri glavna dijela. Prozor s blokovima imao ih je četiri, a sad ladica i radna površina postaju jedan uređivač koda:

- **Uređivač koda** (lijevo): tu pišeš naredbe
- **Platno** (desno gore): tu Tina hoda i crta
- **Konzola** (desno dolje): tu se ispisuju greške i poruke

Ispod konzole je još jedan prazan redak, **redak za naredbe**. U njega možeš upisati jednu naredbu i pritisnuti **Enter**, pa je Tina odmah izvrši od mjesta gdje je stala, bez pokretanja cijelog programa.

4.3 Gumb Pokreni

Gore desno, na plavoj traci, nalazi se gumb  (kad mišem prijeđeš preko njega, piše **Pokreni**). Pritisneš ga kad si spreman da Tina izvrši tvoj kod. Isto radi i tipka **F5**. Gumb  pokraj njega zaustavlja program koji predugo radi.

U prvoj knjizi gumb Pokreni bio je zelen, a Tina je sama sve crtala iznova čim bi promijenio neki blok. U prozoru za kod toga nema: Tina crta tek kad pritisneš  ili **F5**. Tada sve kreće ispočetka: platno je čisto, Tina je u sredini, a konzola prazna.

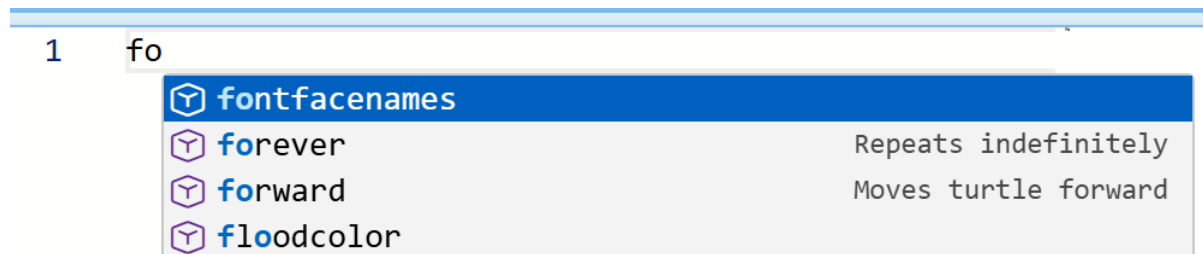
4.4 Brojevi redaka

Lijevo od svakog retka vidiš mali broj: 1, 2, 3, 4...

Ti brojevi pomažu da se snađeš u kodu. Poruka o grešci u konzoli ne kaže broj retka, ali kaže koja naredba nije uspjela, pa taj redak lako pronađeš. Ako je greška unutar procedure (procedure ćeš upoznati kasnije), poruka na kraju doda i ime procedure, na primjer **fd doesn't like x as input in kvadrat**, a ispod nje ispiše cijeli redak u kojem je greška.

Pomoć dok pišeš

Dok pišeš naredbu, uređivač ti pomaže prijedlozima. Počneš tipkati fo i pojavi se popis naredbi koje tako počinju: fontfacenames, forever, forward i floodcolor. Strelicama ↑ i ↓ biraš prijedlog, a tipkom **Enter** ga upišeš, pa ne moraš sve tipkati sam.



Slika 4.2: Uređivač nakon tipkanja fo: popis prijedloga fontfacenames, forever, forward i floodcolor

4.5 Sprema se samo od sebe

Kao i kod blokova, **ne moraš ništa spremati ručno**. Sve što napišeš sprema se automatski. Sljedeći put kad otvoriš projekt, tvoj kod će te čekati. Projekt s kartice **Lokalna pohrana preglednika** čuva se samo u tom pregledniku, na tom računalu.

4.6 Mali test

Prije nego napišemo prvu naredbu, isprobaj ovo:

1. Klikni mišem u prazno polje uređivača.
2. Tipkaj svoje ime.
3. Pritisni gumb  **Pokreni**.

Što se dogodilo? Konzola će crvenim slovima ispisati grešku, na primjer **I don't know how to Ana** (“ne znam kako se radi Ana”). To je zato što tvoje ime nije Logo naredba!



```
I don't know how to Ana
```

Slika 4.3: Konzola s crvenom porukom I don't know how to Ana

To je u redu. Upravo si napravio prvu pravu programersku grešku, a Tina ti je rekla što ne valja. Poruke u konzoli su na engleskom, jer Logo govori engleski, ali u knjizi ćemo ti uvijek reći što znače.

Sada obriši svoje ime i krećemo na pravu naredbu.

Sljedeće: [Tvoja prva naredba →](#)

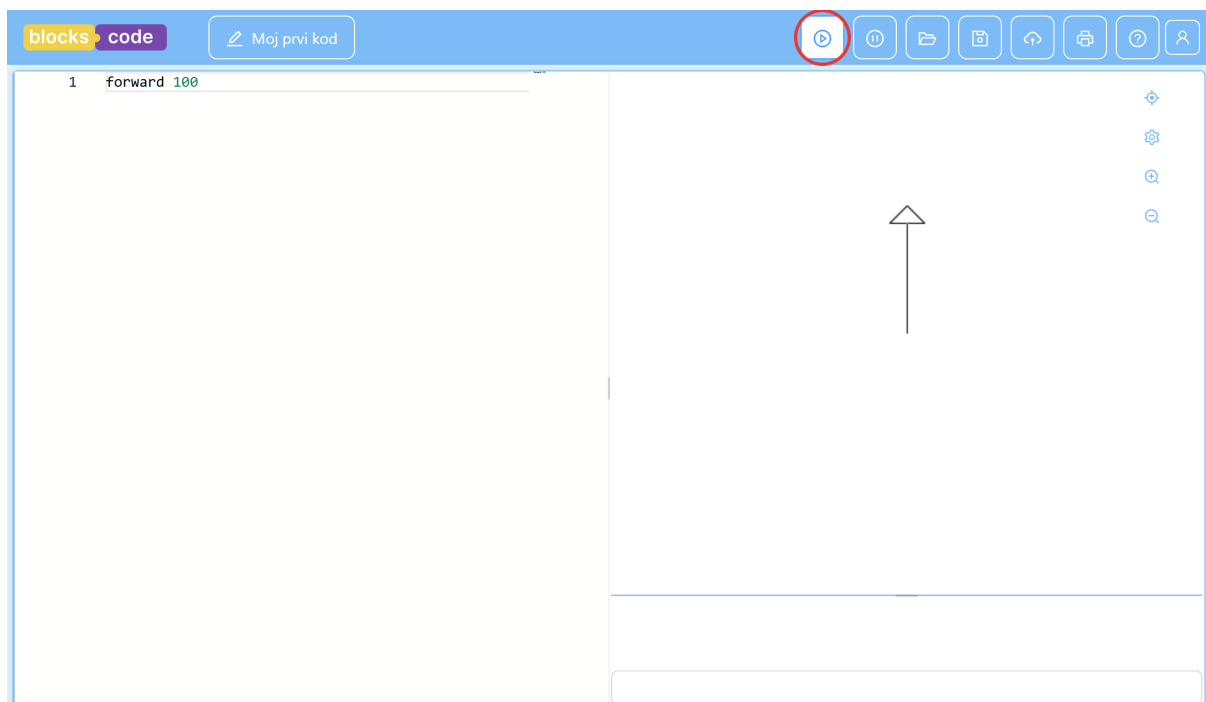
Poglavlje 5

Tvoja prva naredba

Vrijeme je za prvu pravu Logo naredbu! Otvori prozor za kod (ako već nije otvoren) i upiši točno ovo:

```
forward 100
```

Ovaj redak već poznaješ: u prvoj knjizi stajao je ispod bloka **forward**. Sad ga pišeš sam. Pritisni  **Pokreni** (ili tipku **F5**).



Slika 5.1: Uređivač s naredbom forward 100 u prvom retku i crtom od 100 koraka na platnu

Tina je krenula naprijed za 100 koraka i ostavila ravnu crtu! Upravo si napisao svoj prvi kod.



Slika 5.2: Ravna crta od 100 koraka

5.1 Što smo napisali

Pogledajmo što ovaj redak znači, riječ po riječ:

- **forward**: naredba koja kaže Tini “kreni naprijed”
- **100**: broj koraka

Između naredbe i broja stavili smo **razmak**. To je jako važno, jer bez razmaka Tina ne razumije gdje završava naredba, a gdje počinje broj. Ako napišeš `forward100`, konzola će javiti **I don't know how to forward100**: Logo misli da je to jedna nepoznata riječ.

5.2 Pokušaj druge brojeve

Obriši ono što si napisao i upiši:

```
forward 50
```

Pritisni **Pokreni**. Tina je sada nacrtala kraću crtu.



Slika 5.3: Kraća crta od 50 koraka

Probaj još:

```
forward 200
```



Slika 5.4: Duga crta od 200 koraka

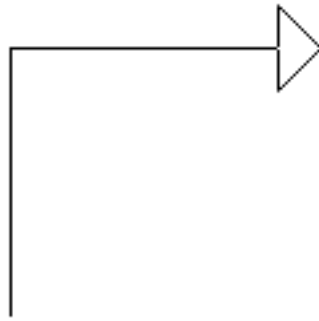
Crta je puno dulja. Broj iza naredbe forward određuje **koliko koraka** Tina prelazi.

5.3 Više naredbi u istom programu

Sada upiši tri naredbe, jednu ispod druge:

```
forward 100  
right 90  
forward 100
```

Pokreni. Tina je nacrtala dvije spojene crte, kao obrnuto slovo **L**.



Slika 5.5: Dvije spojene crte: gore pa desno

Tina izvršava naredbe **redom**, jednu za drugom:

1. Krenula je 100 koraka naprijed.
2. Okrenula se 90 stupnjeva udesno.
3. Ponovno je krenula 100 koraka naprijed.

To se zove **redoslijed izvršavanja**: Tina uvijek čita kod od vrha prema dnu.

Veliko ili malo slovo?

Logo ne pravi razliku između velikih i malih slova u naredbama. `forward 100`, `Forward 100` i `FORWARD 100` rade istu stvar. Mi ćemo u ovoj knjizi koristiti **mala slova**, jer tako piše većina programera.

5.4 Kad nešto ne valja

Probaj namjerno napraviti grešku. Upiši:

```
forward
```

(bez broja) i pokreni. Konzola će crvenim slovima ispisati **not enough inputs to forward**. Ta poruka znači da naredba `forward` nije dobila dovoljno ulaza, to jest nedostaje joj broj koraka.

```
not enough inputs to forward
```

Slika 5.6: Konzola s crvenom porukom `not enough inputs to forward`

Drugi pokušaj:

```
naprijed 100
```

Konzola će reći **I don't know how to naprijed**, što znači “ne znam kako se radi naprijed”. Logo govori engleski, pa Tina ne razumije hrvatske riječi za naredbe. Zato uvijek koristimo forward, right, left, back...

```
I don't know how to naprijed
```

Slika 5.7: Konzola s crvenom porukom I don't know how to naprijed

Pazi još na jedno: kad Logo naiđe na grešku, preskoči ostatak tog retka, ali retke ispod njega ipak izvrši. Zato crtež može biti napola gotov iako konzola javlja grešku. Kad nešto izgleda čudno, uvijek pogledaj konzolu!

5.5 Najčešće poruke o greškama

Evo pet poruka koje ćeš najčešće vidjeti. Dvije s riječju **strana** pojavit će se tek kad upoznaš varijable, ali dobro je imati sve na jednom mjestu:

Poruka u konzoli	Kad se pojavi	Što znači
not enough inputs to forward	forward bez broja	Naredbi nedostaje ulaz, ovdje broj koraka.
I don't know how to naprijed	naprijed 100	Logo ne poznaje naredbu s tim imenom.
fd doesn't like strana as input	fd "strana	Naredba je dobila ulaz koji ne može upotrijebiti, ovdje riječ umjesto broja.
strana has no value	fd :strana bez make	Kutijica strana još nema vrijednost.
You don't say what to do with 0	fd 10 0 umjesto fd 100	Ostao je broj 0, a nijedna naredba ne kaže što da s njim radi.

Greška nije katastrofa

Kad u konzoli vidiš poruku crvenim slovima, ne brini! Pročitaj poruku, popravi to što piše i pokreni ponovno. Programeri ovo rade desetke puta dnevno.

5.6 Ukratko

Naučio si:

- Kako pisati Logo naredbu: **naredba, razmak, broj**
- Da Tina izvršava naredbe **redom, od vrha prema dnu**
- Kako pročitati poruku o grešci u konzoli

Sada smo spremni naučiti sve naredbe za kretanje!

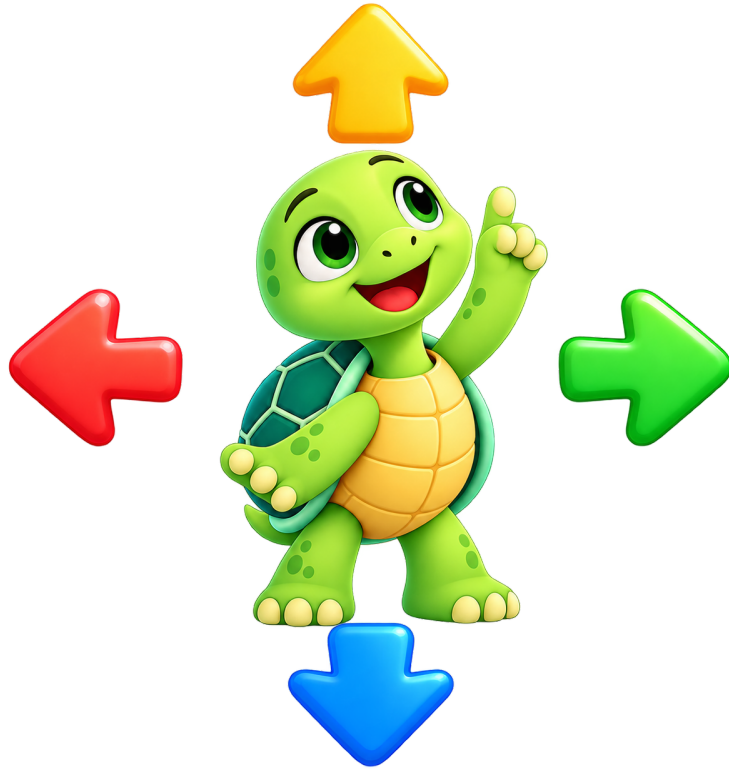
Sljedeće: [Kretanje →](#)

II. dio

Kretanje

Poglavlje 6

Kretanje



Tina zna četiri osnovna pokreta: **naprijed**, **natrag**, **lijevo** i **desno**. Već ih znaš kao četiri žuta bloka iz prve knjige: **forward**, **back**, **left** i **right**. Sve što ćeš ikad nacrtati sastoji se od ta četiri pokreta.

U ovom dijelu ćemo:

- Naučiti naredbe `forward`, `back`, `right` i `left`
- Vidjeti kako brojevi (koraci i stupnjevi) mijenjaju ono što Tina radi
- Upoznati **kratke verzije** naredbi (`fd`, `bk`, `rt`, `lt`)
- Vježbati s nekoliko zadataka

Sljedeće: *Naprijed i natrag* →

Poglavlje 7

Naprijed i natrag

Najvažnije naredbe za Tinu su one koje je pomiču kroz prostor. Imamo dvije, iste kao blokovi **forward** i **back** iz prve knjige:

- **forward**: kreni naprijed
- **back**: kreni natrag

Objekti očekuju **broj** iza naredbe. To je broj koraka koje Tina prelazi.

7.1 Naredba forward

Naredba forward pomiče Tinu naprijed (u smjeru u kojem trenutno gleda).

```
forward 100
```



Slika 7.1: Ravna crta od 100 koraka

Tina će napraviti 100 koraka u smjeru svog nosa i ostaviti ravnu crtu iza sebe.

Što je broj veći, dulji je pomak:

```
forward 200
```



Slika 7.2: Duga crta od 200 koraka

Što je broj manji, kraći je pomak:

```
forward 30
```



Slika 7.3: Kratka crta od 30 koraka

7.2 Naredba back

Naredba back radi suprotno: pomiče Tinu **unatrag**, ali **bez okretanja**. Tina i dalje gleda u istom smjeru, samo hoda unatrag.

```
back 100
```



Slika 7.4: Tina se spustila 100 koraka ispod polazišta, i dalje gleda gore

Tina se pomaknula 100 koraka unatrag i nacrtala crtu ispod polazišta, a nos joj i dalje gleda gore.

7.3 Korisno svojstvo: poništavanje

Ako napišeš ovo:

```
forward 100  
back 100
```



Slika 7.5: Jedna crta, a Tina se vratila na polazište

Tina će krenuti naprijed 100 koraka, pa se vratiti istim putem. Crta će biti iscrtana, ali Tina će završiti **točno tamo gdje je počela**.

To je korisno kad želiš nacrtati nešto, pa se vratiti na isto mjesto za sljedeću crtu.

7.4 Što ako broj ne stane na platno?

Platno je veliko 1000 x 1000 koraka, a Tina kreće iz sredine, pa do ruba ima 500 koraka. Ako joj kažeš forward 700, ona neće nestati, nego će se pojaviti na **suprotnoj strani** platna! Izađe gore, uđe dolje i hoda još 200 koraka. To se zove **omotavanje** (na engleskom *wrap*) i uključeno je samo od sebe. Tina jednostavno izađe s jedne strane i uđe s druge, kao u starim videoigrama.

Da je vratiš u sredinu, napiši naredbu home. Tina se vrati u sredinu i opet gleda gore. Pazi samo: ako je olovka spuštена, Tina usput nacrtá crtu do sredine.

7.5 Mali zadatak

Probaj sam:

1. Pomakni Tinu naprijed za 150 koraka.
2. Pa unatrag za 50 koraka.
3. Pa opet naprijed za 100 koraka.

Gdje je Tina na kraju? Koliko je dugačka nacrtana crta?

```
forward 150  
back 50  
forward 100
```



Slika 7.6: Okomita crta od 200 koraka s Tinom na vrhu

Tina je 200 koraka iznad polazišta ($150 - 50 + 100 = 200$). Vidljiva crta dugačka je 200 koraka, iako je Tina ukupno prehodala 300 koraka: dio puta prešla je dvaput, pa se crte preklapaju.

Negativni brojevi

Tina razumije i negativne brojeve. forward -50 znači isto što i back 50. Većina programera koristi back jer je to jasnije, ali dobro je znati da i ovo radi.

Sljedeće: *Lijevo i desno* →

Poglavlje 8

Lijevo i desno

Da bi Tina nacrtala bilo što osim ravne crte, mora se **okrenuti**. Imamo dvije naredbe za to, iste kao blokovi **right** i **left** iz prve knjige:

- **right**: okreni se desno
- **left**: okreni se lijevo

Obje očekuju **broj stupnjeva** iza naredbe.

8.1 Što je stupanj?

Stupanj (na engleskom **degree**) je mjera za kut. Cijeli krug je **360 stupnjeva**. Pola kruga je **180**. Četvrtina kruga je **90**.

Sjećaš se ovih okreta iz prve knjige? Često ćeš ih koristiti:

Naredba	Što se dogodi
right 90	Okret za četvrtinu kruga (kut kao u kvadratu)
right 180	Okret unatrag (Tina sad gleda u suprotnom smjeru)
right 360	Pun krug (Tina gleda u istom smjeru kao prije)
right 45	Pola četvrtine kruga (koso)

8.2 Naredba right

```
right 90
```



Slika 8.1: Tina u sredini platna okrenuta udesno, bez crte

Tina se okreće za 90 stupnjeva u smjeru kazaljke na satu (udesno). **Ne hoda**, samo se okreće. Ne ostavlja crtu jer ne mijenja položaj.

8.3 Naredba left

```
left 90
```



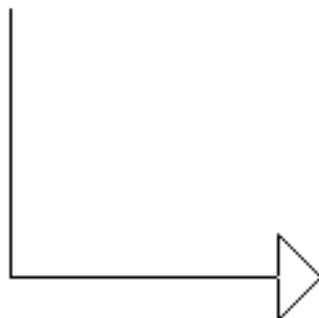
Slika 8.2: Tina u sredini platna okrenuta ulijevo, bez crte

Tina se okreće za 90 stupnjeva suprotno od kazaljke na satu (ulijevo). Također ne hoda i ne crta.

8.4 Spajamo kretanje i okretanje

Da nacrtáš slovo L:

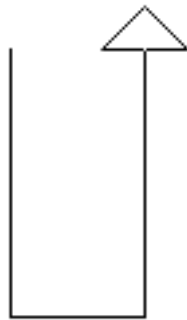
```
forward 100  
back 100  
right 90  
forward 100
```



Slika 8.3: Slovo L: okomita i vodoravna crta iz iste točke

Da nacrtáš slovo **U**:

```
forward 100  
back 100  
right 90  
forward 50  
left 90  
forward 100
```



Slika 8.4: Slovo U

8.5 Pravilo: prvo okreni, pa kreni

Kad god želiš da Tina nacrtá zavoј, pravilo je:

1. **Stani** (Tina se već zaustavlja na kraju svake naredbe)
2. **Okreni se** za neki kut
3. **Kreni** u novom smjeru

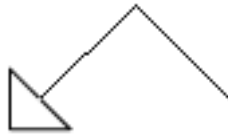
Tina ne može crtati ravnu crtu i istovremeno se okretati. Svaki novi smjer počinje s right ili left.

8.6 Mali test: krov

Pogodi što će ovaj kod nacrtati:

```
left 45  
forward 50  
left 90  
forward 50
```

Pokreni i provjeri. Trebalo bi izgledati kao **gornja polovica romba** ili “krov kuće”.



Slika 8.5: Dvije kose crte kao krov kuće

Nakon left 45 Tina gleda koso gore i ulijevo, a nakon left 90 koso dolje i ulijevo. Zato su obje crte kose i zajedno čine krov.

Provjeri u glavi prije pokretanja

Dobra navika je da prije pritiska na **Pokreni** zamisliš što će se dogoditi. Tako učiš “misliti kao kornjača”, a to je vrlo korisna programerska vještina. Ako zapneš, sjeti se **trika tijela** iz prve knjige: ustani i okreni se kao Tina.

Sljedeće: [Skraćenice naredbi](#) →

Poglavlje 9

Skraćenice naredbi

Pisati forward 100 svaki put može biti dosadno. Programeri vole kratko, pa Logo ima **skraćenice** za sve glavne naredbe za kretanje.

9.1 Tablica skraćénica

Puna naredba	Skraćenica	Što radi
forward	fd	Naprijed
back	bk	Natrag
right	rt	Okret desno
left	lt	Okret lijevo

Skraćenice rade **potpuno isto** kao i pune naredbe. Ovo:

```
forward 100  
right 90
```

...i ovo:

```
fd 100  
rt 90
```

...rade istu stvar. Tina ne vidi razliku.

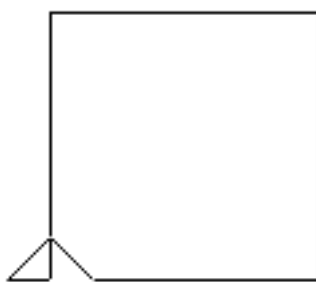
9.2 Primjer: kvadrat sa skraćenicama

Sjećaš se kvadrata iz poglavlja [Zašto pisati kod?](#) Tamo smo ga već napisali skraćenicama:

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

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

Punim nazivima trebao bi četiri puta napisati forward 100 i right 90. Sa skraćenicama je mnogo manje pisanja, a rezultat je isti.



Slika 9.1: Kvadrat sa stranicom 100

9.3 Kad koristimo što

Pune nazive lakše je zapamtiti i drugima pročitati, pa ih ti slobodno piši dok ne zapamtiš skraćénice.

Mi ćemo u ostatku ove knjige koristiti **skraćénice** (fd, bk, rt, lt), jer štede prostor i puno se brže pišu.

9.4 Što sa složenijim naredbama?

I druge naredbe imaju skraćénice. Većinu njih već znaš kao blokove iz prve knjige, a kodom ćeš ih pisati u dijelu **Boje i olovka**:

Puna naredba	Skraćénica	Što radi
penup	pu	Podigni olovku (ne crtaj dok hodaš)
pendown	pd	Spusti olovku (crtaj)
setpencolor	setpc	Promijeni boju olovke
setpensize	nema	Promijeni debljinu olovke
clearscreen	cs	Obriši platno i vrati Tinu u sredinu

Ne moraš zapamtiti sve odjednom

Ne brini ako se ne sjećaš svih skraćenica. Najvažnije su fd, bk, rt, lt. Ostale ćeš naučiti kako budeš išao kroz knjigu.

9.5 Mješavina je dopuštena

Možeš slobodno miješati pune naredbe i skraćenice u istom programu:

```
forward 100  
rt 90  
fd 100  
left 90
```

Ovo radi, ali ne izgleda lijepo. Bolje je odlučiti se za jedan način i držati ga se kroz cijeli program.

Sljedeće: [Probaj sam! →](#)

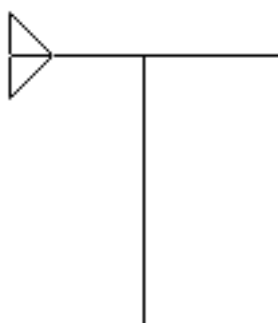
Poglavlje 10

Probaj sam!

Sad kad znaš sve naredbe za kretanje, vrijeme je za vježbu. Probaj riješiti ove zadatke prije nego pogledaš rješenje.

10.1 Zadatak 1: Slovo T

Nacrtaj veliko slovo **T**: vodoravna crta gore i okomita crta od sredine prema dolje.



Slika 10.1: Slovo T

Rješenje

```
fd 100  
rt 90  
fd 50  
bk 100
```

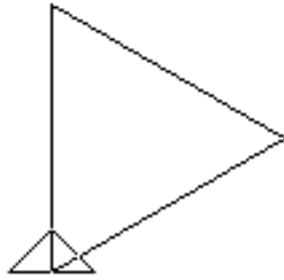
Tina prvo nacrtava okomitu crtu (uspravni dio slova T), zatim se okreće desno i nacrtava polovicu vodoravne crte. Naredba `bk 100` vrati je preko te polovice i nacrtava drugu polovicu, a Tina cijelo vrijeme gleda udesno.

10.2 Zadatak 2: Trokut

Nacrtaj **jednakostranični trokut** sa stranicama dugim 100.

Pomoć: sjećaš se iz prve knjige? U jednakostraničnom trokutu sva tri kuta imaju 60

stupnjeva, ali Tina se okreće za **vanjski kut**: $180 - 60 = 120$ stupnjeva.



Slika 10.2: Jednakostranični trokut

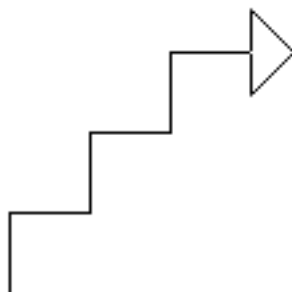
Rješenje

```
fd 100
rt 120
fd 100
rt 120
fd 100
rt 120
```

Tri stranice, tri okreta po 120 stupnjeva. Tina na kraju gleda u istom smjeru kao na početku.

10.3 Zadatak 3: Stepenice

Nacrtaj **tri stepenice**. Svaka stepenica ima vodoravni i okomiti dio od 30 koraka.



Slika 10.3: Tri stepenice

Rješenje

```

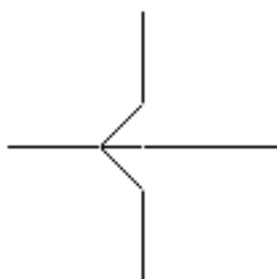
fd 30
rt 90
fd 30
lt 90
fd 30
rt 90
fd 30
lt 90
fd 30
rt 90
fd 30

```

Naizmjenično se okrećemo desno i lijevo, a nakon svakog okreta idemo 30 koraka naprijed. Tako nastaju stepenice.

10.4 Zadatak 4: Plus

Znak plus crtao si i u prvoj knjizi, blokovima. Sad ga napiši kodom: nacrtaj znak **+** s četiri jednaka kraka od po 50 koraka. Svi kraci kreću iz iste točke u sredini.



Slika 10.4: Znak plus s četiri kraka

Pomoć: nakon svakog kraka vrati se u središte i okreni se za 90 stupnjeva prije sljedećeg.

Rješenje

```
fd 50  
bk 50  
rt 90  
fd 50  
bk 50  
rt 90  
fd 50  
bk 50  
rt 90  
fd 50  
bk 50
```

Svaki krak je fd 50 pa bk 50 (povratak u središte), a zatim okret od 90 stupnjeva za sljedeći krak.

10.5 Razmišljaj o rješenjima

Primijetio si nešto u ovim zadacima? **Stalno se ponavljamo!** Trokut ima tri puta fd 100 rt 120, stepenice opet i opet fd 30 rt 90 fd 30 lt 90, a plus četiri puta fd 50 bk 50.

To nije slučajno, jer se puno crteža sastoji od **istog poteza ponovljenog više puta**. Zato u Logu postoji posebna naredba za ponavljanje, koju već znaš kao blok **repeat**. Ona će naš kod skratiti i do deset puta.

Pisat ćeš je u dijelu **Ponavljanje**, odmah nakon oblika!

Sljedeće: [Oblici](#) →

III. dio

Oblici

Poglavlje 11

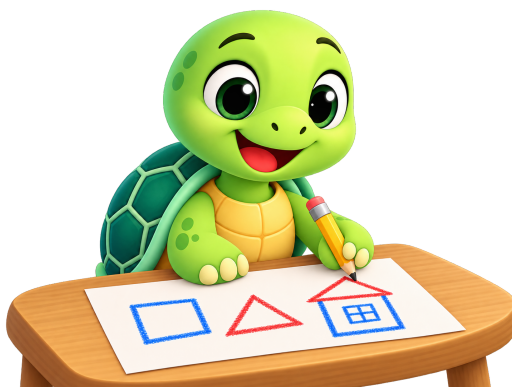
Oblici

Sad kad znaš pomicati Tinu naprijed-natrag i okretati je lijevo-desno, vrijeme je da nacrtamo prave **oblike**. Iste oblike (crtu, kvadrat, trokut i kuću) već si slagao od blokova u prvoj knjizi, a sad ćeš ih napisati kodom.

U ovom dijelu ćemo:

- Pažljivo razložiti kako nastaje **ravna crta**
- Nacrtati **kvadrat** korak po korak
- Naučiti kako se računa kut za **trokut**
- Spojiti više oblika u jednu **kuću**

Sve ćemo i dalje pisati “ručno”, svaku stranicu posebno. U sljedećem dijelu vidjet ćeš kako se blok **repeat** iz prve knjige piše kodom.



Sljedeće: [Ravna crta](#) →

Poglavlje 12

Ravna crta

Najjednostavniji oblik je **ravna crta**. Već znamo kako se ona crta:

```
fd 100
```



Slika 12.1: Okomita crta od 100 koraka

Jedna naredba, jedna crta od 100 koraka. Gotovo!

12.1 Razne duljine

Crte mogu biti različitih duljina:

```
fd 50
```



Slika 12.2: Kratka crta

Kratka crta.

fd 300



Slika 12.3: Dugačka crta

Dugačka crta od 300 koraka. Ni ona ne stiže do ruba, jer je platno veliko: široko je i visoko 1000 koraka, a Tina kreće iz sredine, pa do ruba ima 500 koraka. Da je crta dulja od toga, nastavila bi se na suprotnoj strani platna, baš kao u poglavlju **Naprijed i natrag**.

12.2 Više crta u različitim smjerovima

Ako želiš dvije odvojene crte koje **nisu spojene**, moraš podići olovku između njih. Inače će Tina nacrtati spojnu crtu.

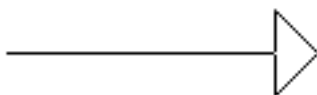
U prvoj knjizi to je radio blok **penup**, a kodom ćemo ga pisati u dijelu **Boje i olovka**. Za sada, sve crte će biti spojene.

12.3 Vodoravna i okomita crta

Na početku svakog pokretanja Tina stoji u sredini platna i gleda **prema gore**. Zato naredba `fd 100` crta **okomitu crtu** prema gore.

Da nacrtáš **vodoravnu crtu**, prvo okreni Tinu:

```
rt 90  
fd 100
```

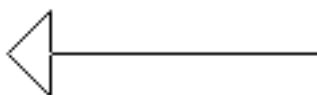


Slika 12.4: Vodoravna crta udesno

Sad imaš vodoravnu crtu udesno.

A za vodoravnu crtu ulijevo:

```
lt 90  
fd 100
```

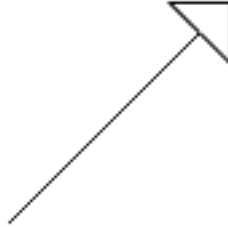


Slika 12.5: Vodoravna crta ulijevo

12.4 Kosa crta

Kosa crta nastaje kad se Tina prije crtanja okrene za kut koji nije 0, 90, 180, 270 ni 360:

```
rt 45  
fd 100
```



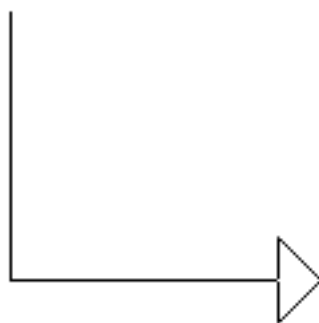
Slika 12.6: Kosa crta gore-desno

Tina se okreće za 45 stupnjeva (pola od 90) i crta dijagonalno, gore-desno.

12.5 Spajanje crta: prvi oblik

Spojimo dvije crte u **slovo L**:

```
fd 100  
bk 100  
rt 90  
fd 100
```



Slika 12.7: Slovo L

Crta gore i natrag, okret, pa crta udesno. Slovo L u dva poteza.

Da nacrtáš slovo **V**, treba ti malo više truda, jer dvije crte trebaju krenuti iz iste točke pa ići u različitim smjerovima:


```
fd 100
bk 100
rt 30
fd 100
```



Slika 12.8: Slovo V: dvije crte iz iste točke

Tina je nacrtala okomitu crtu, vratila se na početak, malo se okrenula i nacrtala drugi krak slova V.

Mali kut, velika razlika

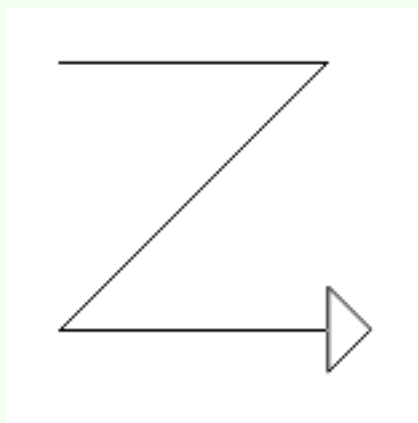
Mijenjaj broj kod naredbe `rt` da vidiš kako mali brojevi (10, 20) daju gotovo paralelne crte, a veliki brojevi (60, 90) daju široko otvorene kutove.

12.6 Vježba

Nacrtaj **slovo Z** koristeći samo `fd`, `bk`, `rt` i `lt`. Zamisli veliko slovo Z: vodoravna crta na vrhu, dijagonala i vodoravna crta na dnu.

Rješenje

```
rt 90
fd 100
rt 135
fd 141
lt 135
fd 100
```



Slika 12.9: Slovo Z

Prvo okrenemo Tinu udesno i nacrtamo vodoravnu crtu od 100. Zatim se okrenemo za 135 stupnjeva i crtamo dijagonalu (141 jer je to približno duljina dijagonale kvadrata sa stranicom 100). Na kraju se okrenemo ulijevo za 135, pa Tina opet gleda udesno i crta donju vodoravnu crtu.

Sljedeće: *Kvadrat korak po korak* →

Poglavlje 13

Kvadrat korak po korak

Kvadrat je najpoznatiji oblik koji se često crta u Logu. Sastoji se od **četiri jednake stranice** i **četiri prava kuta** (kut od 90 stupnjeva).

13.1 Razmišljaj kao Tina

Da nacrtáš kvadrat, Tina treba ovo napraviti:

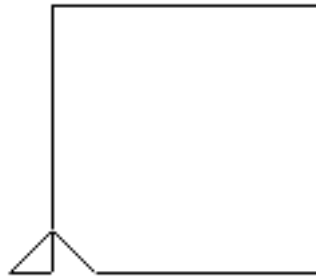
1. Krenuti naprijed (prva stranica)
2. Okrenuti se desno za 90 stupnjeva
3. Krenuti naprijed (druga stranica)
4. Okrenuti se desno za 90 stupnjeva
5. Krenuti naprijed (treća stranica)
6. Okrenuti se desno za 90 stupnjeva
7. Krenuti naprijed (četvrta stranica)
8. Okrenuti se desno za 90 stupnjeva (da na kraju gleda u istom smjeru kao na početku)

13.2 Kvadrat u kodu

Pretvorimo to u Logo kod:

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

Pokreni i vidjet ćeš lijep kvadrat sa stranicama dugim 100 koraka.



Slika 13.1: Kvadrat sa stranicom 100

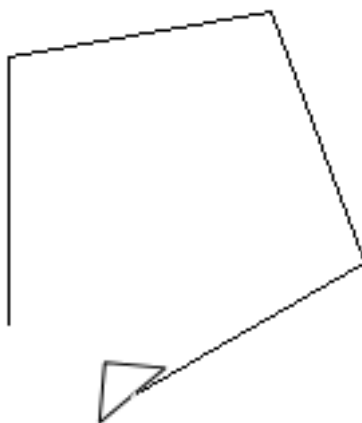
13.3 Zašto baš 90 stupnjeva?

Kvadrat ima **četiri prava kuta**. Pravi kut je 90 stupnjeva. Tina se na svakom uglu okrene za točno toliko da gleda duž sljedeće stranice, koja je okomita na prethodnu.

Što bi se dogodilo da koristimo rt 80? Okreti zajedno ne bi dali 360 ($4 \times 80 = 320$), pa se oblik **ne bi zatvorio**: zadnja stranica ne bi stigla do početka!

Probaj sam:

```
fd 100
rt 80
fd 100
rt 80
fd 100
rt 80
fd 100
rt 80
```



Slika 13.2: Otvoren oblik: okreti od 80 stupnjeva ne zatvaraju kvadrat

Vidiš? Oblik ostaje **otvoren**, crte se ne spajaju. Probaj još s drugačijim kutovima i vidi što dobiješ.

13.4 Pravilo zatvorenih oblika

Postoji jedno pravilo koje vrijedi za svaki **pravilan mnogokut**, to jest za oblik kojem su sve stranice jednake i svi kutovi jednaki:

Zbroj svih okreta mora biti **360 stupnjeva**.

Kvadrat ima 4 okreta x 90 stupnjeva = **360**. Trokut će imati 3 okreta x 120 stupnjeva = **360**. Šesterokut će imati 6 okreta x 60 stupnjeva = **360**.

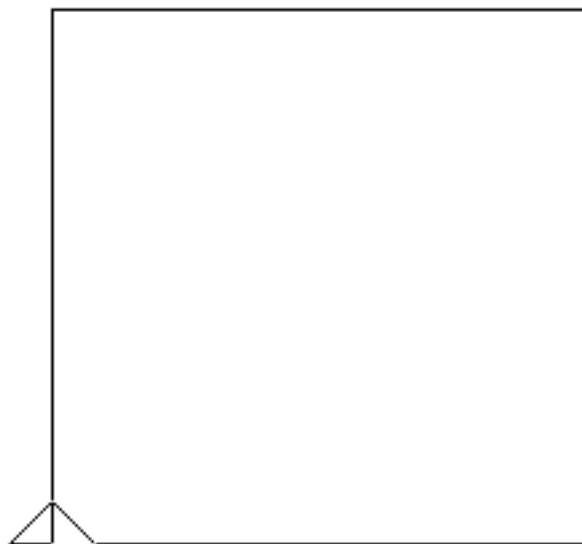
U prvoj knjizi to smo zvali **čarobni broj 360**: dok obiđe takav oblik, Tina se okrene točno jedan puni krug. To pravilo bit će nam jako korisno i kasnije. (Zvijezde su iznimka: dok crta zvijezdu, Tina se okrene dva ili više punih krugova. Vidjet ćeš to u poglavlju **Zvijezde**.)

13.5 Kvadrat različitih veličina

Da nacrtáš veći kvadrat, samo promijeni broj iza fd:

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

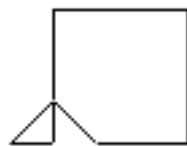
```
fd 200  
rt 90  
fd 200  
rt 90
```



Slika 13.3: Veliki kvadrat

Manji kvadrat:

```
fd 50  
rt 90  
fd 50  
rt 90  
fd 50  
rt 90  
fd 50  
rt 90
```

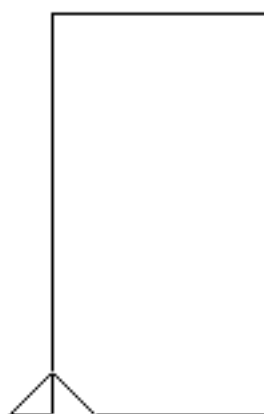


Slika 13.4: Mali kvadrat

13.6 Pravokutnik

A što ako želiš **pravokutnik** umjesto kvadrata? Ima i on četiri prava kuta, ali stranice nisu sve jednake. **Dvije su dulje, a dvije kraće:**

```
fd 150
rt 90
fd 80
rt 90
fd 150
rt 90
fd 80
rt 90
```



Slika 13.5: Pravokutnik 150 x 80

Stranice se izmjenjuju: 150, 80, 150 i 80 koraka.

Brojanje stranica

Pripazi, često se događa da napišeš `fd 150 rt 90` tri puta i `fd 80 rt 90` jednom. Tada dobiješ čudan otvoreni oblik. Pravokutnik treba **točno dvije** dugačke i **točno dvije** kratke stranice, naizmjenično.

13.7 Razmišljanje na kraju

Pogledaj kvadrat iz prvog primjera:

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

Vidiš li kako se isti dio koda ponavlja? **Četiri puta** smo napisali isto: `fd 100` pa `rt 90`. To je puno tipkanja!

U sljedećem dijelu napisat ćeš blok **repeat** kao naredbu `repeat`, pa sve ovo stane u jedan redak:

```
repeat 4 [ fd 100 rt 90 ]
```

Ali prije toga, naučimo još jedan oblik: trokut.

Sljedeće: [Trokut](#) →

Poglavlje 14

Trokut

Trokut ima **tri stranice** i **tri kuta**. Najjednostavnija vrsta trokuta je **jednakostranični trokut**: sva tri kuta su jednaka i sve tri stranice su jednake.

14.1 Koliko stupnjeva za trokut?

Možda si u školi već čuo da tri unutarnja kuta svakog trokuta zajedno uvijek daju **180 stupnjeva**. Ako su sva tri jednaka:

$$180 / 3 = \mathbf{60} \text{ stupnjeva}$$

Znači, svaki **unutarnji** kut jednakostraničnog trokuta ima **60 stupnjeva**.

14.2 Ali Tina koristi vanjski kut!

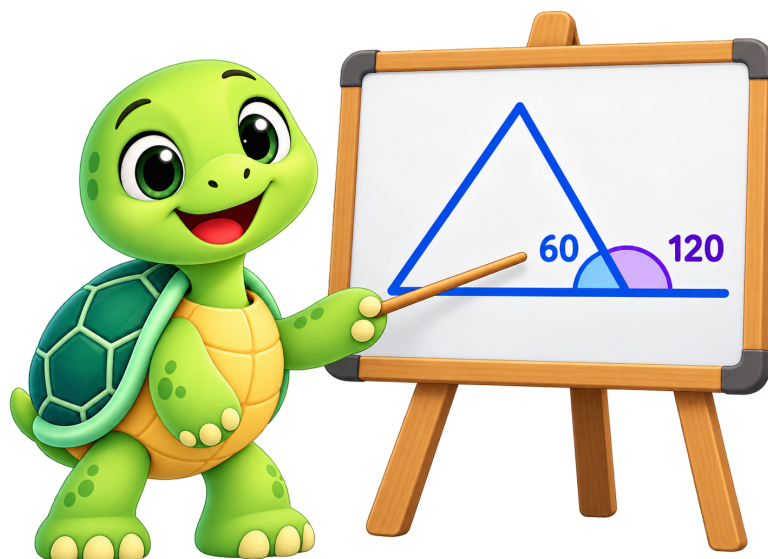
Pažljivo, Tina se ne okreće za **unutarnji** nego za **vanjski** kut. Sjećaš se iz prve knjige? Tina hoda **oko** oblika, a ne stoji unutar njega. Vanjski kut je dopuna unutarnjeg do 180 stupnjeva.

$$\text{Vanjski kut} = 180 - \text{unutarnji kut}$$

Za trokut:

$$\text{Vanjski kut} = 180 - 60 = \mathbf{120 \text{ stupnjeva}}$$

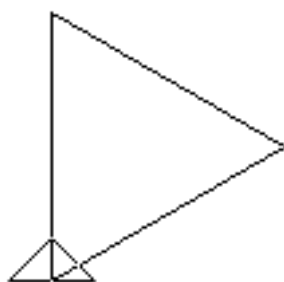
To znači da se Tina na svakom uglu trokuta okreće za **120 stupnjeva**, ne za 60. Isti broj daje i **čarobni broj 360** iz prve knjige: $360 / 3 = \mathbf{120}$.



14.3 Trokut u kodu

```
fd 100
rt 120
fd 100
rt 120
fd 100
rt 120
```

Pokreni ovo i dobit ćeš lijep jednakostranični trokut sa stranicama dugim 100.



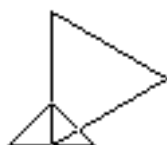
Slika 14.1: Jednakostranični trokut sa stranicom 100

Provjera: 3 okreta x 120 stupnjeva = **360 stupnjeva**. Pravilo zatvorenih oblika potvrđeno.

14.4 Eksperimentiraj s veličinom

Mali trokut:

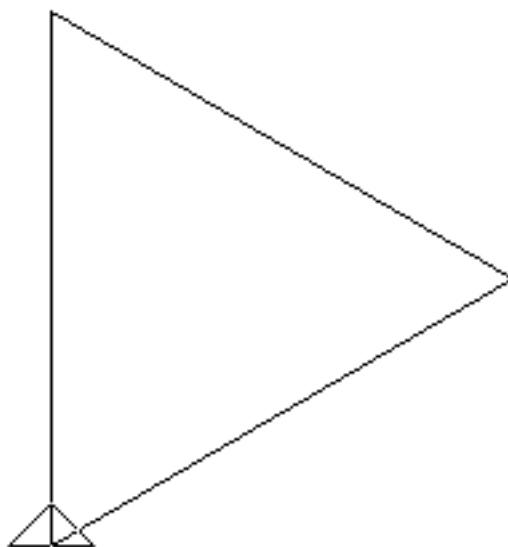
```
fd 50  
rt 120  
fd 50  
rt 120  
fd 50  
rt 120
```



Slika 14.2: Mali trokut

Veliki trokut:

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



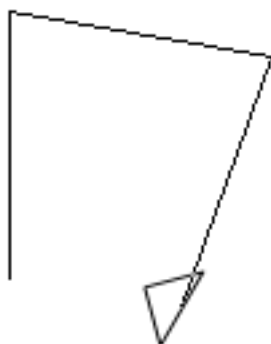
Slika 14.3: Veliki trokut

Stranica može biti bilo koji broj, a oblik ostaje **jednakostranični trokut**.

14.5 Što ako je kut drugačiji?

Probaj rt 100 umjesto rt 120:

```
fd 100  
rt 100  
fd 100  
rt 100  
fd 100  
rt 100
```



Slika 14.4: Otvoren oblik: kutovi od 100 stupnjeva ne zatvaraju trokut

Trokut neće biti zatvoren! Okreti zajedno daju samo $3 \times 100 = 300$ stupnjeva, pa Tina završi negdje u zraku, a ne na polazišnoj točki.

A rt 90?

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



Slika 14.5: Tri stranice kvadrata

Dobiješ tri stranice kvadrata, bez četvrte. Otvoren oblik.

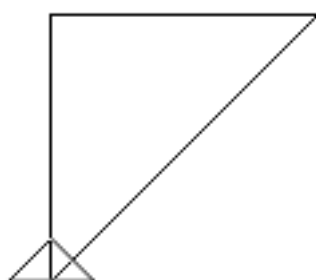
Pravilo: **uvijek izračunaj koliko stupnjeva se Tina treba okrenuti** prije nego napišeš kod.

14.6 Pravokutni trokut

Pravokutni trokut ima **jedan pravi kut**, od 90 stupnjeva, baš kao ugao kvadrata. Zato nije jednakostraničan.

Najjednostavniji je **pravokutni jednakokračni** trokut. On ima dvije jednake kratke stranice i jednu dulju, kosu stranicu.

```
fd 100
rt 90
fd 100
rt 135
fd 141
rt 135
```



Slika 14.6: Pravokutni jednakokračni trokut

Tu se Tina okreće 90, 135 i 135 stupnjeva, a to zajedno daje 360. Kosa stranica dulja je od ostalih dviju: ima otprilike 141 korak.

Za znatiželjne

Stranice uz pravi kut zovu se **katete**, a najdulja stranica, nasuprot pravom kutu, zove se **hipotenuza**. Matematičari znaju da je ova hipotenuza dugačka 100 puta **korijen iz 2**. Umjesto da sam izračunaš 141, možeš napisati $fd\ 100 * \sqrt{2}$. Naredba `sqrt` računa korijen, pa Logo sam izračuna duljinu i Tina prijeđe točno onoliko koraka koliko treba.

Pazi na redoslijed: računaska naredba kao što je `sqrt` uzme **cijeli račun** koji stoji iza nje. Zato `fd sqrt 2 * 100` znači korijen iz 200, pa bi Tina prešla samo 14 koraka. Više o računanju naučit ćeš u poglavlju **Računanje**.

14.7 Što sada?

Sad znaš nacrtati kvadrat i trokut. Spojimo ih u jedan crtež: kuću!

Sljedeće: [Kuća →](#)

Poglavlje 15

Kuća

Kuća je **kvadrat s trokutom** na vrhu (krov). Da je nacrtáš, treba ti samo ono što već znaš i malo planiranja.

Kuću si već slagao od blokova u prvoj knjizi. Ovaj put krov crtamo malo drugačije: kao cijeli trokut, s okretima ulijevo.

15.1 Plan

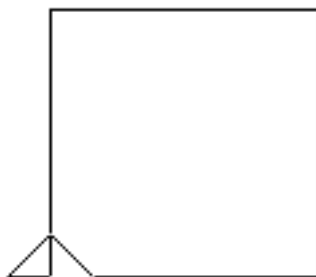
Razmislimo prije nego krenemo pisati kod:

1. Nacrtati kvadrat (zidovi kuće)
2. Dovedi Tinu u **gornji lijevi ugao** kvadrata, okrenutu **udesno**
3. Nacrtati trokut nadovezan na vrh kvadrata (krov)

Pažljivo: kad Tina završi crtanje kvadrata, ona se vraća na **isti položaj i u isti smjer** kao na početku. Treba je “podesiti” prije crtanja krova.

15.2 Korak 1: Kvadrat

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

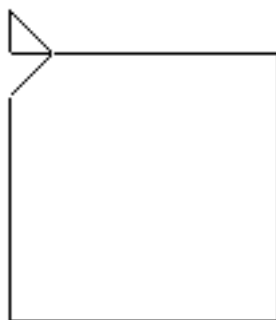
Slika 15.1: Zidovi kuće: kvadrat, a Tina u donjem lijevom uglu gleda gore

Kvadrat sa stranicama 100. Tina sada gleda prema gore i nalazi se u **donjem lijevom uglu**.

15.3 Korak 2: Doći do gornjeg lijevog ugla

Tina treba doći do gornjeg lijevog ugla kvadrata i okrenuti se **udesno**, da može krenuti crtati prvu stranicu trokuta. Ove retke dopiši ispod kvadrata:

```
fd 100  
rt 90
```



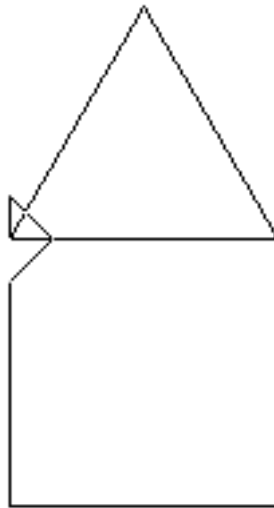
Slika 15.2: Kvadrat, a Tina u gornjem lijevom uglu gleda udesno

Sad je Tina u **gornjem lijevom uglu**, gleda **udesno**.

15.4 Korak 3: Trokut (krov)

Krov je trokut s istom stranicom kao i kvadrat, 100 koraka. Već znamo recept za jednakostranični trokut, samo što se Tina sad okreće **ulijevo**. Dopiši i ove retke:

```
fd 100
lt 120
fd 100
lt 120
fd 100
lt 120
```



Slika 15.3: Krov nacrtan okretima ulijevo sjedi na kvadratu

Zašto ulijevo? Tina prvo hoda udesno po vrhu kvadrata. Okret ulijevo vodi je **prema gore**, pa se trokut nacrtava **iznad** kvadrata, baš kako želimo. Da smo i u trokutu okretali udesno (rt 120), Tina bi skrenula prema dolje i krov bi se nacrtao unutar kvadrata!

15.5 Cijela kuća u kodu

Evo cijelog programa na jednom mjestu:

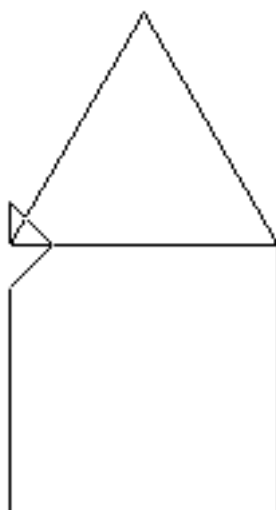
```
fd 100
rt 90
fd 100
rt 90
fd 100
rt 90
fd 100
```

```

rt 90
fd 100
rt 90
fd 100
lt 120
fd 100
lt 120
fd 100
lt 120

```

Pokreni i divi se! Imaš pravu kuću s krovom.



Slika 15.4: Kuća: kvadrat s trokutastim krovom

15.6 Što se događa korak po korak

Razložimo to:

Redak	Što se dogodi
1-8	Kvadrat (zidovi)
9	Tina ide gore lijevim zidom kvadrata
10	Okret udesno za 90, sad gleda udesno
11-16	Trokut (krov)

Pogledaj sliku: Tina na kraju stoji u gornjem lijevom uglu kvadrata, na početku krova, i

gleda udesno. Nije se vratila na polazište!

15.7 Vrata i prozori

Možeš dodati i **vrata**, mali pravokutnik na donjem dijelu kuće. Ali za to ti trebaju naredbe **pu** i **pd** (podigni i spusti olovku), jer se do mjesta gdje će biti vrata moraš pomaknuti **bez crtanja**. U prvoj knjizi to su bili blokovi **penup** i **pendown**.

To dolazi u dijelu **Boje i olovka**. Za sada, evo skice koraka koje bi napravio:

1. Nacrtaj kuću (već znaš)
2. **Podigni olovku** (pu)
3. Pomakni se do mjesta gdje su vrata
4. **Spusti olovku** (pd)
5. Nacrtaj mali pravokutnik (vrata)

Vratit ćemo se na to kad upoznamo **pu** i **pd**.

15.8 Razmišljanje na kraju

Pogledaj koliko si redaka koda napisao za kuću: **šesnaest**. A znaš što? Veći dio tog koda se **ponavlja**:

- Kvadrat: 4 puta **fd 100 rt 90**
- Trokut: 3 puta **fd 100 lt 120**

U sljedećem dijelu naučit ćeš pisati naredbu **repeat**, pa će kuća izgledati ovako:

```
repeat 4 [ fd 100 rt 90 ]  
fd 100  
rt 90  
repeat 3 [ fd 100 lt 120 ]
```

Od šesnaest redaka ostala su samo četiri.

Sljedeće: [Ponavljanje](#) →

IV. dio

Ponavljanje

Poglavlje 16

Ponavljanje



Ako si pažljivo prošao dio **Oblici**, primijetio si da pišeš iste retke koda **opet i opet**. Kvadrat je četiri puta fd 100 rt 90. Trokut je tri puta fd 100 rt 120. Šesterokut bi bio šest puta fd 100 rt 60.

Programeri imaju jedno pravilo:

Ako pišeš isto više od dva puta, postoji bolji način.

Taj bolji način se zove **petlja**. U prvoj knjizi to je bio zeleni blok **repeat**, a u kodu je to naredba **repeat**.

U ovom dijelu ćemo:

- Upoznati naredbu repeat
- Naučiti crtati **mnogokute** (5, 6, 8, 12 stranica) bez puno pisanja
- Crtati **zvijezde** s različitim brojem krakova
- Vidjeti kako se **dvije petlje** mogu složiti jedna u drugu

Sjedeće: [Naredba repeat](#) →

Poglavlje 17

Naredba repeat

Naredba repeat (na hrvatskom **ponovi**) kaže Tini: “napravi ovo nekoliko puta zaredom”.

To je isti blok **repeat** iz prve knjige. Ono što je bilo unutar bloka, sad pišeš između [i].

17.1 Sintaksa

```
repeat BROJ [  
    naredbe  
]
```

- **repeat**: sama naredba
- **BROJ**: koliko puta želiš da se nešto ponovi
- [...]: uglate zagrade unutar kojih su naredbe koje se ponavljaju

Sve naredbe između [i] će se izvršiti **BROJ** puta.

17.2 Najjednostavniji primjer

```
repeat 4 [  
    fd 50  
]
```



Slika 17.1: Ravna crta od 200 koraka

Što se dogodilo? Tina je krenula naprijed... 50 koraka. Pa još 50. Pa još 50. Pa još 50.

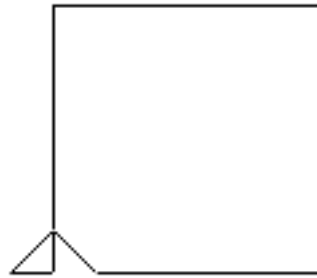
Sve skupa: **200 koraka u jednoj ravnoj crti**. Zato što između koraka nije bilo okreta.

17.3 Kvadrat s repeatom

Sada onaj naš kvadrat:

```
repeat 4 [  
    fd 100  
    rt 90  
]
```

Četiri kratka retka koda umjesto osam. Rezultat: **isti kvadrat** kao prije.



Slika 17.2: Kvadrat nacrtan repeat petljom

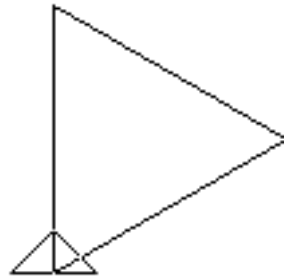
Jedno ponavljanje zove se **prolaz** kroz petlju, a programeri kažu i **iteracija**. Pogledaj što se događa u svakom prolazu:

Prolaz	Što se dogodi
1.	fd 100, rt 90 (prva stranica + okret)
2.	fd 100, rt 90 (druga stranica + okret)
3.	fd 100, rt 90 (treća stranica + okret)
4.	fd 100, rt 90 (četvrta stranica + okret)

Naša petlja ima **4 prolaza**.

17.4 Trokut s repeatom

```
repeat 3 [  
    fd 100  
    rt 120  
]
```



Slika 17.3: Trokut nacrtan repeat petljom

Trokut sa stranicama 100. Njegovi unutarnji kutovi imaju 60 stupnjeva, a Tina se u svakom od tri prolaza okreće za 120.

17.5 Šesterokut

A što ako trebamo **šesterokut**? Sjećaš se iz prve knjige? Za oblik s N stranica okret je **$360 \div N$** . U kodu znak $\div$ pišemo kao /, pa formula za vanjski kut glasi:

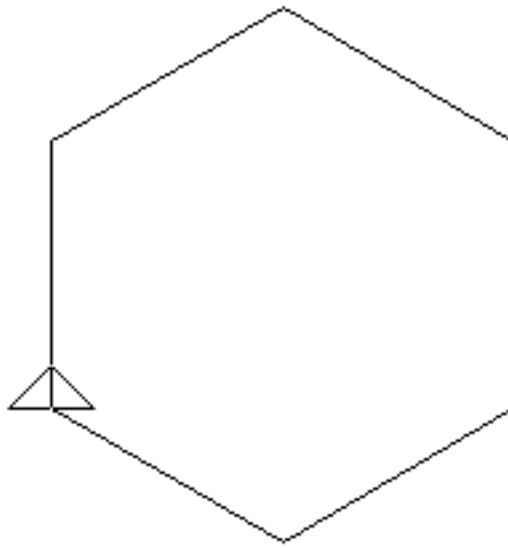
$$\text{Vanjski kut} = 360 / \text{broj stranica}$$

Za šest stranica:

$$360 / 6 = \mathbf{60} \text{ stupnjeva}$$

```
repeat 6 [  
    fd 100  
    rt 60  
]
```

Lijep šesterokut!



Slika 17.4: Šesterokut sa stranicom 100

17.6 Pravilo: koliko ponavljanja, toliko stranica

Za bilo koji **pravilan mnogokut** (oblik s jednakim stranicama i kutovima):

Broj ponavljanja = broj stranica

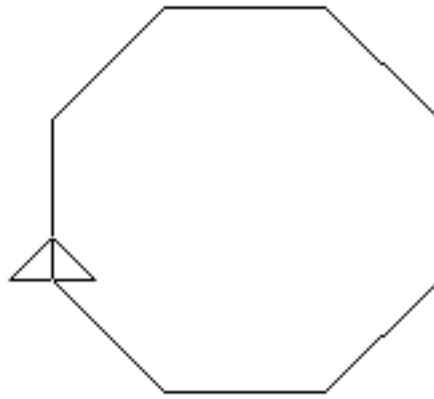
Vanjski kut = $360 / \text{broj stranica}$

Probaj sam:

- Peterokut: 5 stranica, kut = 72
- Sedmerokut: 7 stranica, kut je otprilike 51.4
- Osmerokut: 8 stranica, kut = 45
- Dvanaesterokut: 12 stranica, kut = 30

Pazi: u Logu se decimalni broj piše s **točkom**, a ne sa zarezom kao u školi. Pišeš 51.4, a ne 51,4.

```
repeat 8 [  
  fd 60  
  rt 45  
]
```



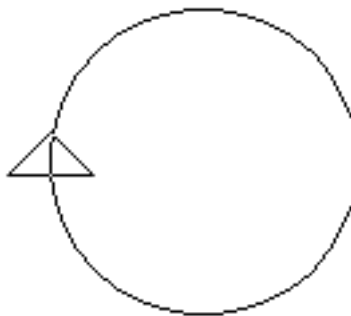
Slika 17.5: Osmerokut

Lijep osmerokut, kao prometni znak STOP.

17.7 Što više stranica, to bliže krugu

Probaj nešto zanimljivo: krug od 36 malih koraka, koji si u prvoj knjizi složio od blokova.

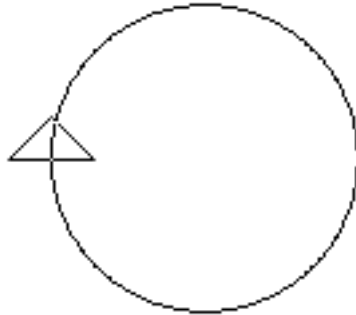
```
repeat 36 [  
  fd 10  
  rt 10  
]
```



Slika 17.6: Krug od 36 kratkih stranica

To je 36 stranica po 10 koraka, s okretom od 10 stupnjeva. Rezultat? Gotovo pravi **krug**! Pravi krug nema ravnih stranica, ali 36 kratkih stranica dovoljno je da oko ne primijeti razliku.

```
repeat 360 [  
  fd 1  
  rt 1  
]
```



Slika 17.7: Savršeno gladak krug

Ovo crta krug koji izgleda **savršeno gladak**: 360 sitnih koraka, svaki s okretom od 1 stupnja. I to je zapravo mnogokut, samo s toliko kratkih stranica da ih oko ne vidi.

Za gotov krug i dalje imaš naredbe iz prve knjige: `circle 50` nacrtá krug oko Tine, a `circle2 50` Tina sama obiđe krug po rubu. Ovdje krug gradimo od malih koraka da vidiš kako nastaje.

Probaj veće brojeve

Možeš naredbi `repeat` dati jako velike brojeve: 100, 500, čak i 1000. Ako pritom Tina izađe s jedne strane platna, pojavit će se na suprotnoj!

17.8 Razmak unutar zagrada

Logo je tolerantan na razmake. Sve ovo radi isto:

```
repeat 4 [ fd 100 rt 90 ]
```

```
repeat 4 [  
  fd 100  
  rt 90  
]
```

```
repeat 4 [fd 100 rt 90]
```

Mi ćemo koristiti **prvu ili drugu varijantu**, jer se lakše čitaju. Druga (s nizom uvučenih naredbi) je posebno korisna za složene petlje.

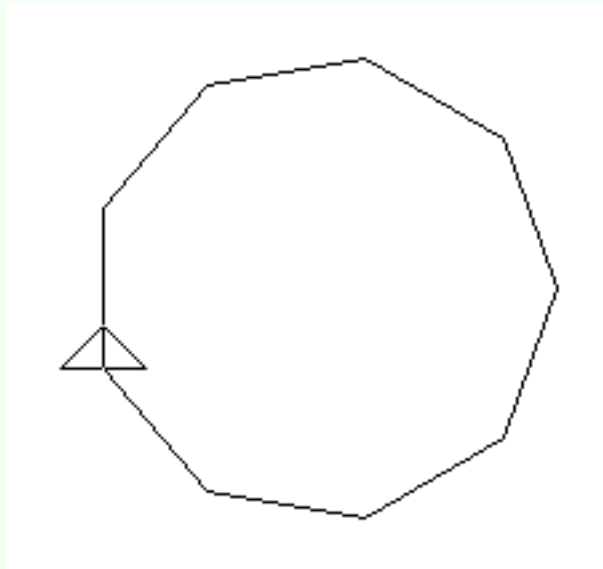
17.9 Vježba

1. Nacrtaj **deveterokut** (9 stranica). Koji je kut?
2. Nacrtaj **kvadrat** koristeći samo lt (lijeve okrete) umjesto rt. U kojem će smjeru ići crtanje?

Rješenja

1. $\text{Kut} = 360 / 9 = 40$ stupnjeva.

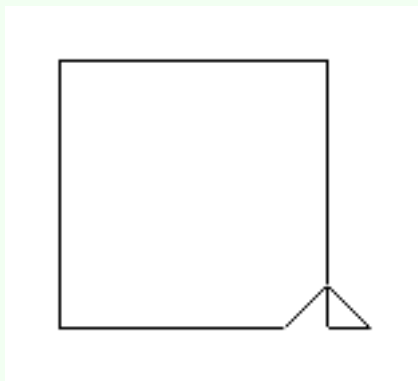
```
repeat 9 [ fd 60 rt 40 ]
```



Slika 17.8: Deveterokut

2. Kvadrat s lt crta se u smjeru suprotnom od kazaljke na satu, lijevo od početne točke.

```
repeat 4 [ fd 100 lt 90 ]
```



Slika 17.9: Kvadrat nacrtan lijevim okretima, u suprotnom smjeru

Sljedeće: [Mnogokuti](#) →

Poglavlje 18

Mnogokuti

Mnogokut (grčkom riječju **poligon**) je svaki zatvoreni oblik s ravnim stranicama. Ako su sve stranice i kutovi jednaki, to je **pravilan mnogokut**.

18.1 Ponavljamo formulu

Već smo naučili formulu:

$$\text{Vanjski kut} = 360 / \text{broj stranica}$$

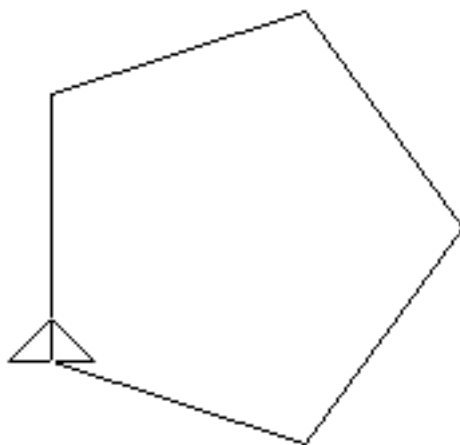
Tablica najčešćih:

Broj stranica	Naziv	Vanjski kut
3	Trokut	120
4	Kvadrat	90
5	Peterokut	72
6	Šesterokut	60
7	Sedmerokut	~51.4
8	Osmerokut	45
10	Deseterokut	36
12	Dvanaesterokut	30

18.2 Primjeri

18.2.1 Peterokut (kao u nogometnoj lopti)

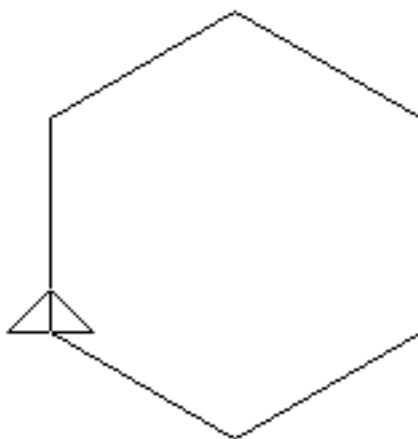
```
repeat 5 [  
    fd 100  
    rt 72  
]
```

Slika 18.1: Peterokut

18.2.2 Šesterokut (kao saće)

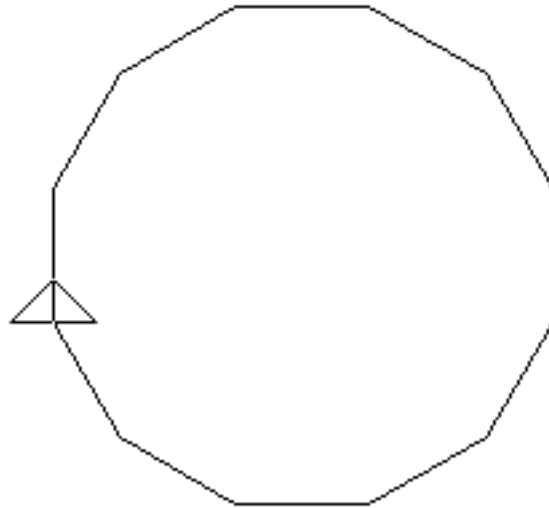
```
repeat 6 [  
  fd 80  
  rt 60  
]
```



Slika 18.2: Šesterokut

18.2.3 Dvanaesterokut (gotovo krug)

```
repeat 12 [  
  fd 50  
  rt 30  
]
```

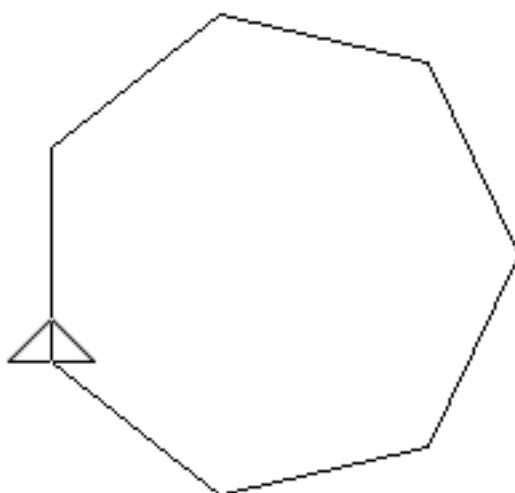


Slika 18.3: Dvanaesterokut

18.3 Sedmerokut: kut s decimalama

Sedmerokut zahtijeva kut od $360 / 7 = 51.428571...$ stupnjeva. Logo razumije i decimalne brojeve:

```
repeat 7 [  
  fd 80  
  rt 51.43  
]
```



Slika 18.4: Sedmerokut

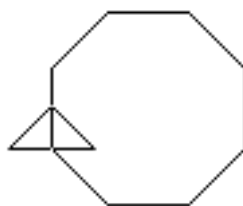
Sedmerokut neće biti **savršen** jer 51.43 nije točno $360 / 7$, ali oku će izgledati dobro.

Ali Logo zna i sam računati, pa možeš napisati `rt 360 / 7` i sedmerokut će se točno zatvoriti. Više o računanju naučit ćeš u poglavlju **Računanje**.

18.4 Veliki i mali mnogokuti

Probaj isti broj stranica s različitim duljinama stranica:

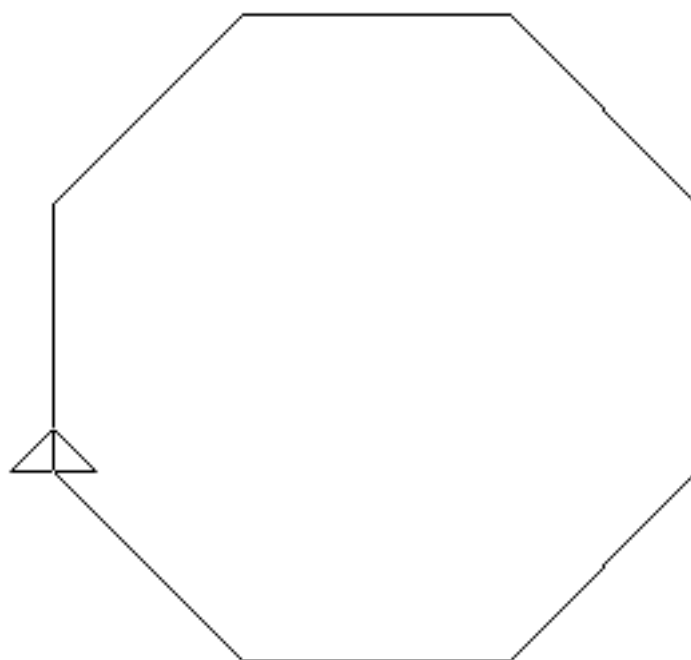
```
repeat 8 [ fd 30 rt 45 ]
```



Slika 18.5: Mali osmerokut

Mali osmerokut.

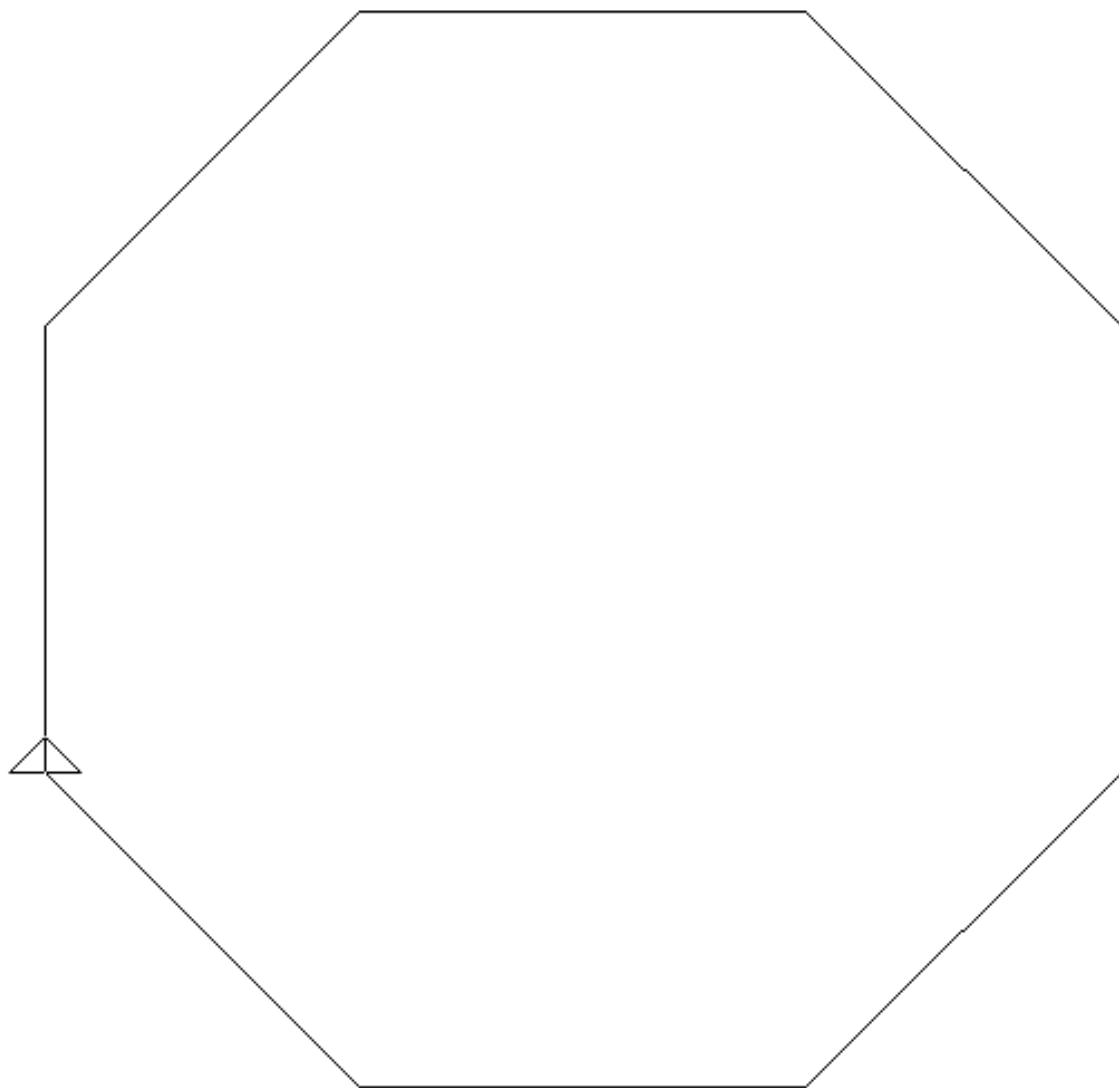
```
repeat 8 [ fd 100 rt 45 ]
```



Slika 18.6: Veliki osmerokut

Veliki osmerokut.

```
repeat 8 [ fd 200 rt 45 ]
```



Slika 18.7: Ogroman osmerokut

Ogroman osmerokut: širok je skoro 500 koraka, a to je gotovo pola platna.

Veličinu mnogokuta najviše mijenja **duljina stranice**. Šesterokut s fd 200 je veći od dvanaesterokuta s fd 50, iako ima manje stranica. Ali i broj stranica nešto znači: uz istu duljinu stranice, mnogokut s više stranica je veći.

18.5 Provjera: zatvoren oblik

Sjećaš se pravila zatvorenih oblika? Za svaki **pravilan mnogokut** vrijedi:

Zbroj svih okreta mora biti 360 stupnjeva.

Provjerimo na našim mnogokutima:

- Trokut: $3 \times 120 = 360$ ☒
- Kvadrat: $4 \times 90 = 360$ ☒
- Peterokut: $5 \times 72 = 360$ ☒
- Dvanaesterokut: $12 \times 30 = 360$ ☒

Ako napišeš nešto poput `repeat 5 [ fd 100 rt 50 ]`, Tina će se ukupno okrenuti samo $5 \times 50 = 250$ stupnjeva, pa oblik **neće biti zatvoren**.

A što ako okreti zajedno daju više od 360, recimo 720? Tada Tina obiđe sredinu dvaput i može nastati **zvijezda**. To je tema sljedećeg poglavlja.

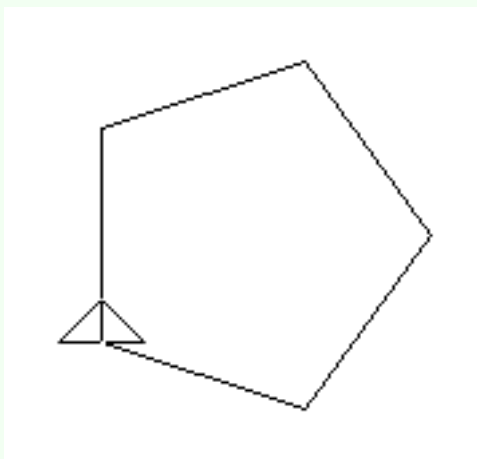
18.6 Vježba

1. Nacrtaj peterokut sa stranicom 80 koraka.
2. Što se dogodi ako `rt 72` zamijeniš s `lt 72`? Napravi ga i sam.
3. Probaj nacrtati šesnaesterokut. Koji kut treba?

Rješenja

1. Peterokut sa stranicom 80:

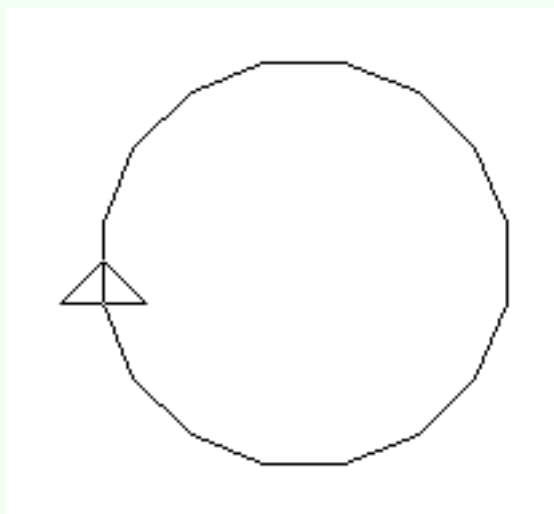
```
repeat 5 [ fd 80 rt 72 ]
```



Slika 18.8: Peterokut sa stranicom 80

2. S `repeat 5 [ fd 80 lt 72 ]` peterokut se nacrtava u suprotnom smjeru (suprotno od kazaljke na satu), pa se nalazi **lijevo** od početne točke, kao zrcalna slika prvoga. I dalje je zatvoren, jer je $5 \times 72 = 360$.
3. $360 / 16 = 22.5$ stupnjeva.

```
repeat 16 [ fd 30 rt 22.5 ]
```



Slika 18.9: Šesnaesterokut

Sljedeće: [Zvijezde](#) →

Poglavlje 19

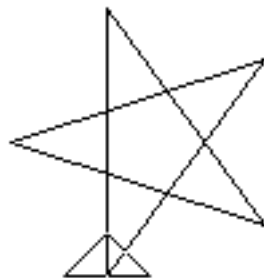
Zvijezde

Zvijezde su slične mnogokutima, jer imaju jednake stranice i jednake kutove. Ali kut nije izračunat iz formule **$360 / \text{broj stranica}$** . Kod zvijezda kut je **veći**, pa Tina prelazi preko svojih crta.

19.1 Petokraka zvijezda

Zvijezda sa 5 krakova crta se ovako:

```
repeat 5 [  
  fd 100  
  rt 144  
]
```



Slika 19.1: Petokraka zvijezda

Pet stranica, kut od **144 stupnja**.

Provjera: $5 \times 144 = 720$, što je dva puna kruga. Tina je obišla sredinu dvaput, pa se stranice sijeku u sredini.

19.2 Pravilo zvijezde

Sjećaš se **pravila zvijezde** iz prve knjige? Za zvijezdu s N krakova uzmeš repeat N i okret **$360 \times 2 \div N$** . Broj 2 znači da se Tina, dok crta zvijezdu, okrene **dva puna kruga**.

U kodu znak $\times$ pišemo kao $*$, a znak $\div$ kao $/$. Logo sam izračuna kut, pa pravilo pišeš baš ovako:

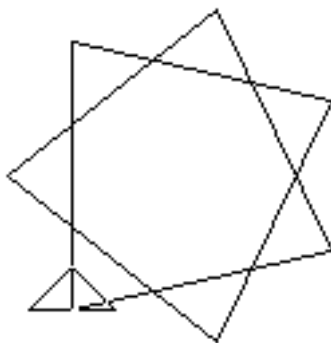
```
repeat N [ fd 100 rt 360 * 2 / N ]
```

Umjesto N upiši broj krakova. Za petokraku zvijezdu to je $rt\ 360 * 2 / 5$, a Logo izračuna **144**. Zato baš 144!

Krakovi	Račun	Kut
5	$360 * 2 / 5$	144
7	$360 * 2 / 7$	otprilike 102.86
9	$360 * 2 / 9$	80

Zvijezda sa 7 krakova:

```
repeat 7 [
  fd 100
  rt 360 * 2 / 7
]
```



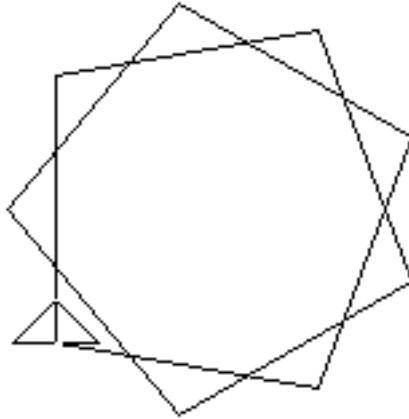
Slika 19.2: Sedmokraka zvijezda po pravilu zvijezde

Provjera: $7 \times 102.86 =$ otprilike 720, opet dva puna kruga.

I zvijezda s 9 krakova:

```
repeat 9 [
  fd 100
```

```
rt 360 * 2 / 9
]
```



Slika 19.3: Devetokraka zvijezda po pravilu zvijezde

Provjera: $9 \times 80 = 720$, i opet dva puna kruga.

19.3 Za znatiželjne: oštija zvijezda

Ovaj dio možeš i preskočiti, jer ti je pravilo zvijezde dovoljno. Postoji i druga formula. Ona daje zvijezdu s **tanjim, oštrijim krakovima**, ali radi samo za **neparan** broj krakova (5, 7, 9...):

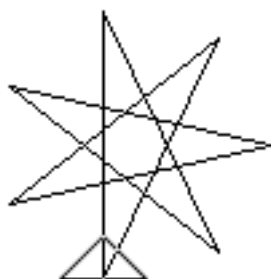
$$\text{Kut zvijezde} = 180 - (180 / \text{broj krakova})$$

Za 5 krakova: $180 - 180 / 5 = \mathbf{144}$, ista zvijezda kao prije ☒

Ali za zvijezdu sa 7 krakova dobiješ drugi kut:

$$180 - (180 / 7) = 180 - 25.71 = \mathbf{154.29}$$

```
repeat 7 [
  fd 100
  rt 154.29
]
```

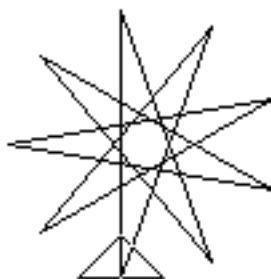


Slika 19.4: Sedmokraka zvijezda s oštrim krakovima

Provjera: $7 \times 154.29 =$ otprilike **1080**, a to su **tri** puna kruga. Tina se okreće jače i obiđe sredinu triput, pa su krakovi tanji.

Za 9 krakova formula daje $180 - 180 / 9 = \mathbf{160}$, pa Tina obiđe sredinu čak četiri puta ($9 \times 160 = 1440$):

```
repeat 9 [  
  fd 100  
  rt 160  
]
```



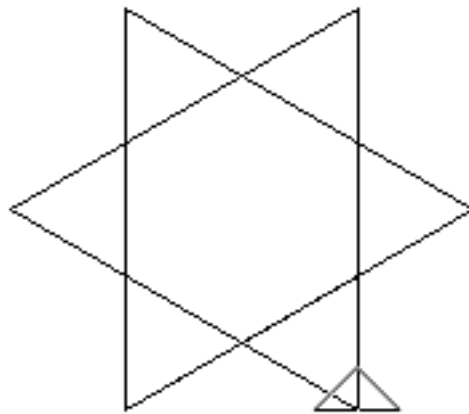
Slika 19.5: Devetokraka zvijezda s tankim krakovima

Devetokraka zvijezda s vrlo “tankim” krakovima.

19.4 Šestokraka zvijezda

Po pravilu zvijezde kut za 6 krakova bio bi $360 \times 2 / 6$, a to je 120. Ali rt 120 je okret za trokut, pa bi Tina samo dvaput nacrtala isti trokut! Zato šestokraku zvijezdu crtamo kao **dva trokuta**, od kojih je jedan obrnut:

```
repeat 3 [ fd 150 rt 120 ]
rt 60
fd 100
lt 60
bk 50
repeat 3 [ fd 150 lt 120 ]
```



Slika 19.6: Šestokraka zvijezda od dva trokuta

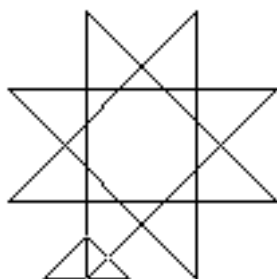
To je već prilično komplicirano, ali lijepo izgleda!

Probaj i ovo: `repeat 6 [ fd 50 lt 60 fd 50 rt 120 ]`. Tina jednom petljom nacrtala samo **rub** šestokrake zvijezde, bez crta u sredini.

19.5 Osmokraka zvijezda

I s 8 krakova pravilo zvijezde zapne: $360 * 2 / 8$ je 90, a `rt 90` crta kvadrat, pa bi Tina dvaput obišla isti kvadrat. Zato uzmemo **tri** puna kruga umjesto dva: $360 * 3 / 8$ je **135**.

```
repeat 8 [
  fd 100
  rt 135
]
```



Slika 19.7: Osmokraka zvijezda

Osmokraka zvijezda (kut 135). Provjera: $8 \times 135 = 1080$, tri puna kruga, pa se oblik zatvori.

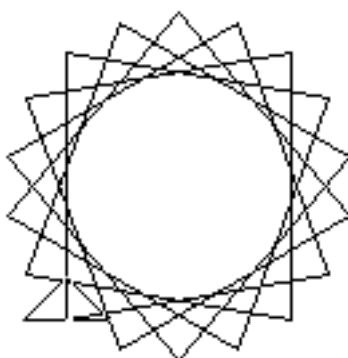
Formula za oštrij zvijezdu ovdje ne pomaže, jer je 8 paran broj: $180 - 180 / 8 = 157.5$, a $8 \times 157.5 = 1260$. To nije cijeli broj krugova, pa se oblik ne zatvori.

19.6 “Cvjetni” uzorak

Što ako uzmeš neki “čudan” kut, na primjer 100? Probaj:

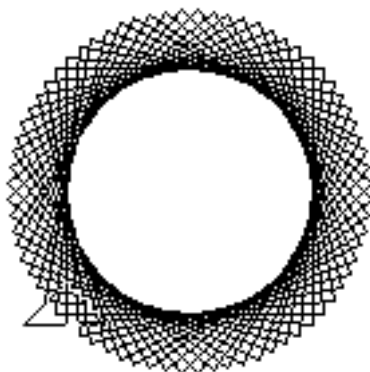
```
repeat 36 [
  fd 100
  rt 100
]
```

Tina će se okretati $36 \times 100 = 3600$ stupnjeva, a to je **deset punih krugova**. Rezultat: prekrasan cvjetni uzorak gdje stranice dijagonalno presijecaju jedna drugu.



Slika 19.8: Cvjetni uzorak od 36 crta

```
repeat 72 [
  fd 100
  rt 95
]
```



Slika 19.9: Cvjetni uzorak od 72 crte

Drugi cvjetni uzorak. Mali pomaci u kutu (95 umjesto 100) daju potpuno drugačije slike.

Igraj se!

Ovo je najbolji dio Loga: eksperimentiranje. Mijenjaj brojeve repeat-a i rt-a i pogledaj što se događa. Često ćeš dobiti slike ljepše od svega što si namjeravao.

19.7 Kad se oblik zatvori?

Evo pravila koje vrijedi i za mnogokute i za zvijezde:

Oblik se zatvori kad Tina ponavlja iste korake, a **broj ponavljanja puta kut** je cijeli broj punih krugova: 360, 720, 1080...

- Točno **360** (jedan krug) daje **mnogokut**.
- **720** ili više (dva ili više krugova) daje **zvijezdu** ili cvjetni uzorak. Iznimka je kad Tina samo ponovi isti oblik, kao repeat 8 [fd 100 rt 90], koji dvaput nacrtava isti kvadrat.
- Ako umnožak nije cijeli broj krugova, Tina završi negdje drugdje i oblik ostane **otvoren**.
- Sam kut ne smije biti pun krug. S repeat 4 [fd 100 rt 360] Tina se nakon svakog okreta opet gleda u istom smjeru, pa nacrtava samo ravnu crtu.

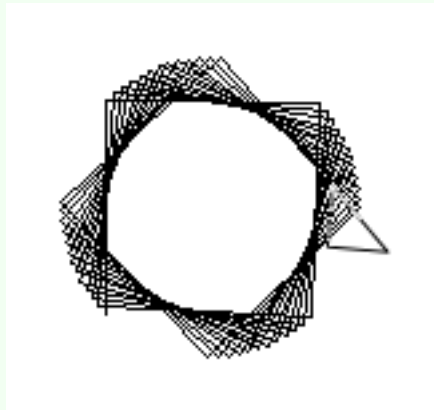
19.8 Vježba

Probaj nacrtati ovih nekoliko zvijezda:

1. Petokraka zvijezda sa stranicom 80
2. Sedmokraka zvijezda
3. Cvjetni uzorak po vlastitom izboru (slobodno eksperimentiraj)

Rješenja

1. repeat 5 [fd 80 rt 144]
2. Po pravilu zvijezde: repeat 7 [fd 100 rt $360 * 2 / 7$]. Oštija zvijezda: repeat 7 [fd 100 rt $180 - 180 / 7$] ili jednostavnije repeat 7 [fd 100 rt 154.29].
3. Beskrajno mnogo mogućnosti! Na primjer repeat 50 [fd 80 rt 91].



Slika 19.10: Zvezdani uzorak od 50 crta

Sljedeće: [Ugniježđeni repeat](#) →

Poglavlje 20

Ugniježđeni repeat

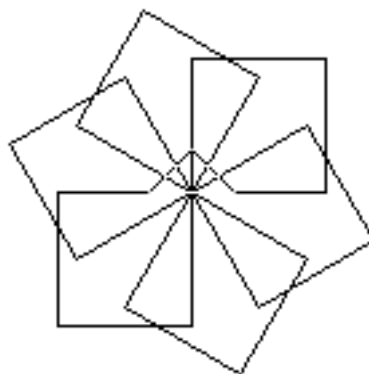
Što ako želimo **petlju unutar petlje**? Logo to dopušta, a to otvara vrata vrlo zanimljivim crtežima.

U prvoj knjizi si blok **repeat** već stavio unutar drugog bloka **repeat** i dobio cvijet od 12 kvadrata. Kad jedan repeat stoji unutar drugog, to zovemo **ugniježđeni repeat** (na engleskom **nested loop**). Vanjska petlja određuje “koliko puta”, a unutarnja određuje **što** se ponavlja.

20.1 Najjednostavniji primjer

Nacrtajmo nekoliko kvadrata zaredom, svaki pomalo zakrenut:

```
repeat 6 [  
  repeat 4 [ fd 50 rt 90 ]  
  rt 60  
]
```



Slika 20.1: Šest kvadrata raspoređenih u krugu

Što se događa:

1. **Vanjska petlja** se izvršava 6 puta
2. **Unutar nje**, svaki put se prvo nacrtaju cijeli kvadrat (repeat 4 [fd 50 rt 90])

3. Zatim se Tina okrene za 60 stupnjeva
4. Pa nacrtaj sljedeći kvadrat, ali sad zakrenut za 60 stupnjeva

Ukupno: **šest kvadrata** raspoređenih u krugu, svaki zakrenut za 60 stupnjeva više od prethodnog.

20.2 Razložimo ga

Pogledajmo unutarnju petlju zasebno:

```
repeat 4 [ fd 50 rt 90 ]
```

Ova petlja crta jedan kvadrat sa stranicom 50.

A vanjska petlja:

```
repeat 6 [  
    nacrtaj kvadrat      (ovdje dođe unutarnja petlja)  
    rt 60  
]
```

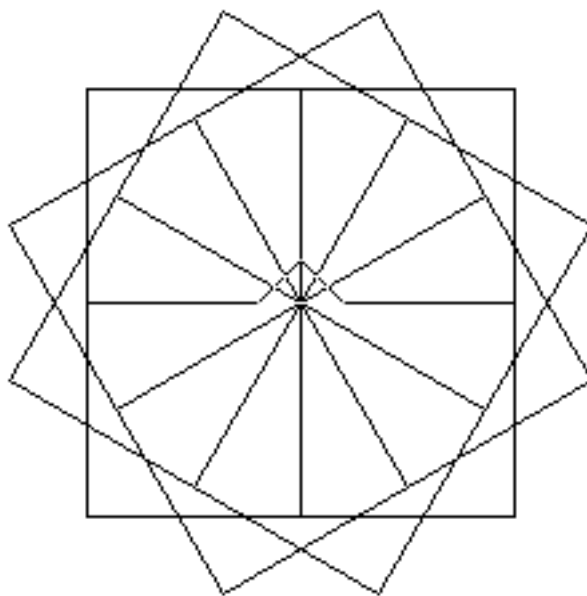
Šest puta: nacrtaj nešto pa se okreni za 60. Budući da je $6 \times 60 = 360$, Tina će se vratiti u isti smjer kao na početku.

20.3 Cvijet od kvadrata

Probaj sljedeće:

```
repeat 12 [  
    repeat 4 [ fd 80 rt 90 ]  
    rt 30  
]
```

12 kvadrata, svaki zakrenut za 30 stupnjeva ($12 \times 30 = 360$). Rezultat izgleda kao **cvijet** s 12 latica u obliku kvadrata.



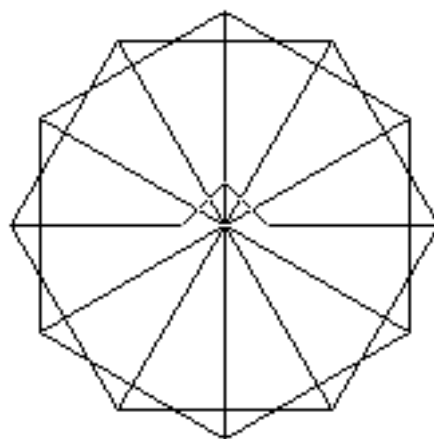
Slika 20.2: Cvijet od 12 kvadrata

Ovo je isti cvijet koji si u prvoj knjizi složio od blokova.

20.4 Cvijet od trokuta

Slično, ali s trokutima:

```
repeat 12 [
  repeat 3 [ fd 80 rt 120 ]
  rt 30
]
```



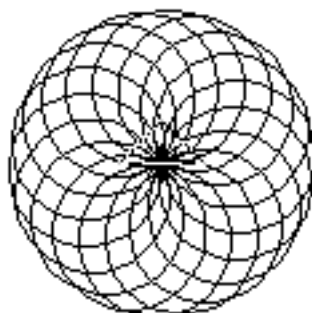
Slika 20.3: Cvijet od 12 trokuta

Druga vrsta cvijeta.

20.5 Cvijet od krugova

Ako unutarnja petlja crta krug:

```
repeat 18 [
  repeat 36 [ fd 5 rt 10 ]
  rt 20
]
```



Slika 20.4: Cvijet od 18 krugova

Vanjska petlja: 18 puta. Unutarnja: krug (36 sitnih koraka). Pomak između krugova: 20 stupnjeva.

Ovaj program crta **mного** crtica: 18 krugova po 36 koraka, ukupno 648. Računalo je brzo, pa se crtež pojavi gotovo odmah.

20.6 Pravilo: vanjska x unutarnja = ukupno ponavljanja

Ako vanjska petlja ima 6 prolaza, a unutarnja 4, onda se naredbe unutarnje petlje izvršavaju **6 x 4 = 24 puta**.

S većim brojevima to brzo raste:

```
repeat 100 [  
    repeat 100 [ fd 1 rt 1 ]  
]
```

Ovo je $100 \times 100 = 10000$ koraka! Računalo ih nacrtalo gotovo odmah. Tina pritom više od 27 puta obiđe isti krug od 360 koraka, pa na platnu vidiš samo jedan krug. Tek kad bi obje petlje imale po 10000 ponavljanja, računalo bi radilo dugo. Takav program možeš zaustaviti gumbom .

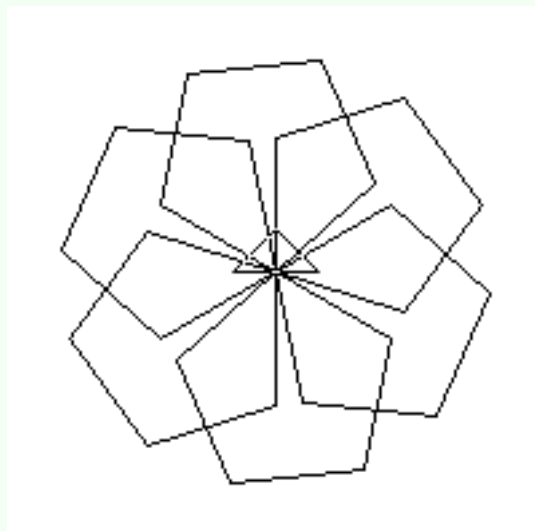
20.7 Vrijeme za vlastite eksperimente

Ovo je najljepši dio Loga: kombiniranje. Probaj:

1. Šest peterokuta u krug
2. Dvanaest osmerokuta s pomakom od 30 stupnjeva
3. Krug krugova
4. Trokut sastavljen od trokuta

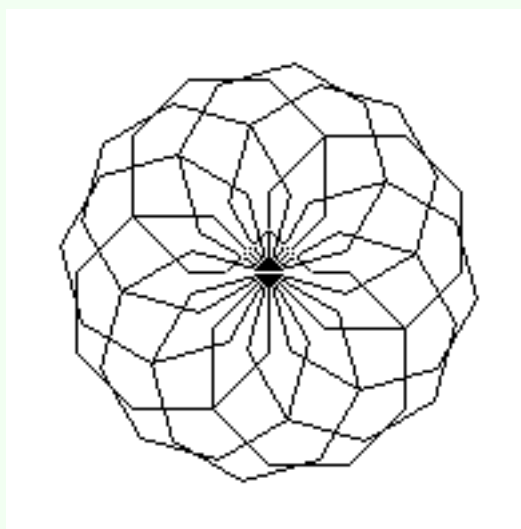
Primjeri

1. `repeat 6 [ repeat 5 [ fd 50 rt 72 ] rt 60 ]`



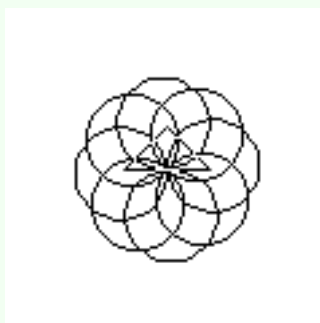
Slika 20.5: Šest peterokuta u krugu

2. repeat 12 [repeat 8 [fd 30 rt 45] rt 30]



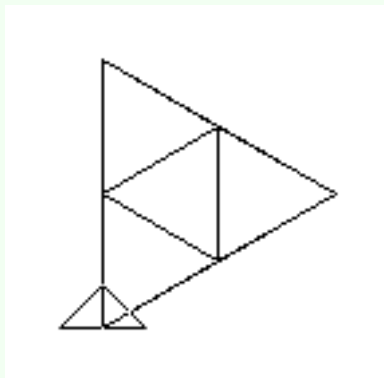
Slika 20.6: Dvanaest osmerokuta

3. repeat 8 [repeat 36 [fd 3 rt 10] rt 45]



Slika 20.7: Krug krugova

```
4. repeat 3 [ repeat 3 [ fd 50 rt 120 ] fd 100 rt 120 ]
```



Slika 20.8: Trokut sastavljen od trokuta

20.8 Što sad?

Sad znaš kombinirati petlje. Ali jedan problem ostaje: petlja uvijek crta **istu stvar**. Kvadrat sa stranicom 50, kvadrat sa stranicom 100... svaki put pišeš novi repeat.

To rješavaju **varijable** i **procedure**, koje već znaš iz prve knjige kao kutijice i vlastite blokove. Uskoro ćeš ih pisati kodom.

Sljedeće: [Boje i olovka](#) →

V. dio

Boje i olovka

Poglavlje 21

Boje i olovka

Sve dosad smo u ovoj knjizi crtali **crnom** olovkom koja je **uvijek bila spuštена**. U prvoj knjizi si Tininu olovku već podizao, bojao i podebljavao blokovima iz crvene ladice **Crtanje**. Sad ćeš iste tri stvari napisati kodom. Naučit ćeš:

- **Podizati i spuštati** olovku, da možeš crtati odvojene oblike
- Mijenjati **boju** crte
- Mijenjati **debljinu** crte

S ova tri alata crteži postaju mnogo bogatiji.



Sljedeće: [Olovka gore i dolje →](#)

Poglavlje 22

Olovka gore i dolje

Tina na svojim leđima nosi olovku. Sve dosad je bila **spuštena**: kad god Tina hoda, olovka crta. Ali što ako želimo da Tina hoda **bez crtanja**?

Tada olovku **podignemo**. Dvije naredbe:

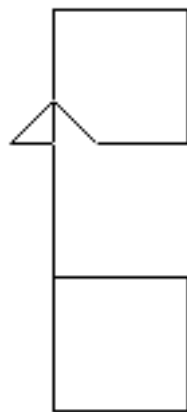
- **penup** (kratko: pu): podigni olovku
- **pendown** (kratko: pd): spusti olovku natrag

To su blokovi **penup** i **pendown** iz crvene ladice **Crtanje**, sad napisani riječima.

22.1 Primjer: dva odvojena kvadrata

Bez podizanja olovke dva kvadrata bila bi spojena crtom:

```
repeat 4 [ fd 50 rt 90 ]  
fd 100  
repeat 4 [ fd 50 rt 90 ]
```

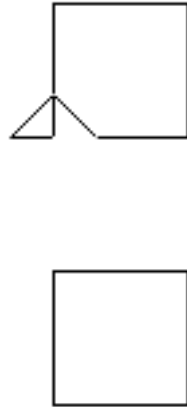


Slika 22.1: Dva kvadrata spojena neželjenom crtom

Vidiš onu crtdu koja spaja dva kvadrata? Ona je tu jer Tina crta dok se pomiče.

Sad probaj ovo, isto kao s blokovima u prvoj knjizi:

```
repeat 4 [ fd 50 rt 90 ]  
pu  
fd 100  
pd  
repeat 4 [ fd 50 rt 90 ]
```



Slika 22.2: Dva odvojena kvadrata

Dva kvadrata, **bez spojne crte**. Tina je između njih hodala s podignutom olovkom.

22.2 Pravilo

- pu: **od sada nadalje**, Tina hoda bez crtanja
- pd: **od sada nadalje**, Tina opet crta

Olovka ostaje u onom stanju u kojem si je posljednji put postavio.

22.3 Vrata na kući

Sjećaš se kuće iz dijela **Oblici**? Sad joj možemo dodati **vrata i prozor**:

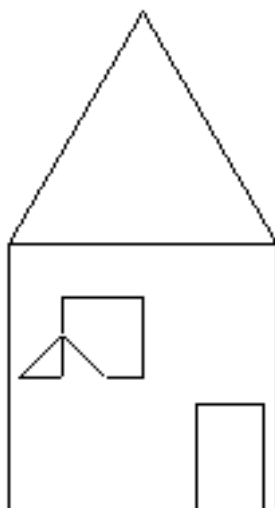
```
; Crtamo kuću s krovom  
repeat 4 [ fd 100 rt 90 ]  
fd 100  
rt 90  
repeat 3 [ fd 100 lt 120 ]
```

```
; Idemo do mjesta za vrata, bez crtanja
pu
rt 90
fd 100
lt 90
fd 70
lt 90
pd

; Crtamo vrata, mali pravokutnik
repeat 2 [
  fd 40 rt 90
  fd 25 rt 90
]

; Idemo do mjesta za prozor
pu
lt 90
fd 50
rt 90
fd 50
pd

; Crtamo prozor, mali kvadrat
repeat 4 [ fd 30 rt 90 ]
```



Slika 22.3: Kuća s krovom, vratima i prozorom

Nakon krova Tina stoji u gornjem lijevom uglu i gleda udesno. Zato se s podignutom olovkom prvo okrene prema dolje, siđe niz lijevi zid i tek onda krene udesno, do mjesta za vrata.

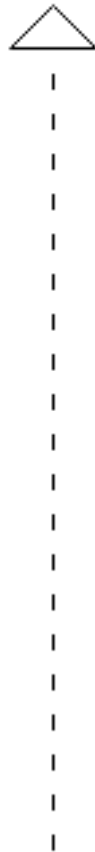
Redci koji počinju sa ; su **komentari**: tekst koji Logo preskače, a nama služi za bilješke. O njima više u poglavlju [Prva procedura](#).

22.4 Primjer: isprekidana crta

Olovka može i samo “kratko taknuti” papir. Umjesto točaka nacrtat ćemo **mного malih odvojenih crtica**. Takvu isprekidanu crtu već si u prvoj knjizi složio od blokova:

```
repeat 20 [  
  pd  
  fd 5  
  pu  
  fd 10  
]
```

20 puta: spusti olovku, nacrtaj kratku crticu, podigni olovku, hodaj malo dalje. Dobit ćeš **isprekidanu crtu** kao na geografskoj karti.



Slika 22.4: Isprekidana crta od 20 crtica

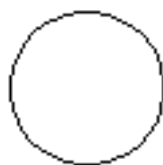
22.5 Sakrij i pokaži Tinu

Iako nije baš o olovci, postoje i naredbe za **skrivanje** i **pokazivanje** Tine:

- **hideturtle** (kratko: ht): sakrij Tinu
- **showturtle** (kratko: st): pokaži je opet

Tina crta jako brzo: crtež se pojavi gotovo odmah čim pritisneš  **Pokreni**. Zato je ne moraš skrivati dok crta. Naredbu ht zato najčešće pišemo **na kraj** programa: Tina nestane, a crtež je čišći.

```
repeat 36 [ fd 5 rt 10 ]  
ht
```



Slika 22.5: Krug od 36 malih koraka, bez Tine

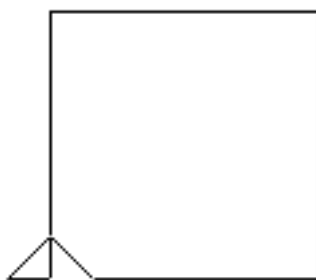
Želiš li Tinu opet vidjeti, dopiši st. Pojavit će se tamo gdje je stala, na početku kruga.

22.6 Brisanje platna usred programa

Kad pritisneš **Pokreni**, platno se samo od sebe očisti, a Tina krene iz sredine prema gore. Zato na početak programa ne trebaš pisati ništa posebno.

Naredba **clearscreen** (kratko: **cs**) treba ti kad želiš obrisati platno **usred** programa. Ona obriše sve što je dotad nacrtano i vrati Tinu u sredinu, okrenutu prema gore:

```
repeat 36 [ fd 5 rt 10 ]  
cs  
repeat 4 [ fd 100 rt 90 ]
```



Slika 22.6: Samo kvadrat, jer je cs obrisao krug

Pazi, **cs** briše samo crtež. Boju i debljinu olovke, o kojima učiš u sljedeća dva poglavlja, ne vraća na početne, a podignuta olovka ostaje podignuta.

Sljedeće: [Boja crte →](#)

Poglavlje 23

Boja crte

Crna crta je u redu, ali šaren crtež je puno ljepši. Naredba **setpencolor** (kratko: **setpc**) mijenja boju olovke. U prvoj knjizi boju si birao s palete na bloku **setpencolor**, a u kodu je pišeš riječju.

23.1 Sintaksa s nazivom boje

Najlakši način je napisati **engleski naziv boje** s navodnikom ispred:

```
setpencolor "red"
fd 100
```



Slika 23.1: Crvena crta

Crta će biti **crvena**.

Pažnja: ispred imena boje stoji **navodnik** ", ali **iza nema navodnika**. To je posebnost Loga:

- **Pravilno:** setpencolor "blue"
- **Pogrešno:** setpencolor "blue"

Za pogrešan oblik konzola će javiti **setpencolor doesn't like blue" as input**, to jest "setpencolor ne voli blue" kao ulaz". Navodnik na kraju postao je dio riječi, a boje s takvim imenom nema.

23.2 Najčešće boje

Logo razumije ova engleska imena, a i mnoga druga:

Naziv	Boja
"red	crvena
"green	zelena
"blue	plava
"yellow	žuta
"orange	narandžasta
"purple	ljubičasta
"pink	ružičasta
"brown	smeđa
"black	crna
"white	bijela
"cyan	cijan
"magenta	ružičasto-ljubičasta

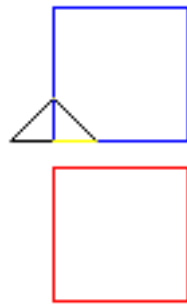
23.3 Mijenjanje boje usred crteža

Boju možeš promijeniti **bilo kad**, pa i usred crteža:

```
setpencolor "red
repeat 4 [ fd 50 rt 90 ]

setpencolor "blue
pu
fd 60
pd
repeat 4 [ fd 50 rt 90 ]
```

Crveni kvadrat dolje, plavi gore.

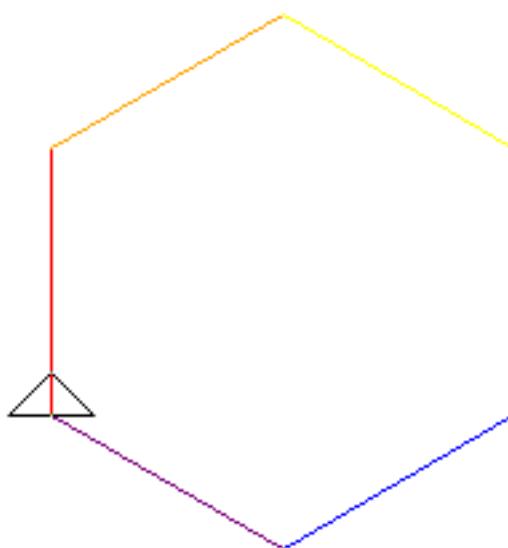


Slika 23.2: Crveni i plavi kvadrat

23.4 Šareni mnogokut

Boju možeš mijenjati i nakon **svake stranice**, kao zrake šarene zvijezde iz prve knjige:

```
setpencolor "red
fd 100
rt 60
setpencolor "orange
fd 100
rt 60
setpencolor "yellow
fd 100
rt 60
setpencolor "green
fd 100
rt 60
setpencolor "blue
fd 100
rt 60
setpencolor "purple
fd 100
rt 60
```



Slika 23.3: Šesterokut u bojama duge

Šesterokut u **bojama duge**.

23.5 Boja kao tri broja (RGB)

Logo razumije i boje pisane kao **tri broja** za crveno, zeleno i plavo. Svaki broj je između 0 i 255:

```
setpencolor [255 0 0]  
fd 100
```



Slika 23.4: Crvena crta iz RGB brojeva

To je crvena (255 crvenog, 0 zelenog, 0 plavog).

```
setpencolor [0 255 0]  
fd 100
```



Slika 23.5: Zelena crta iz RGB brojeva

To je jarko zelena. Riječ "green" daje tamniju zelenu, [0 128 0].

```
setpencolor [128 0 128]  
fd 100
```



Slika 23.6: Ljubičasta crta iz RGB brojeva

To je ljubičasta (mješavina crvenog i plavog), ista kao "purple.

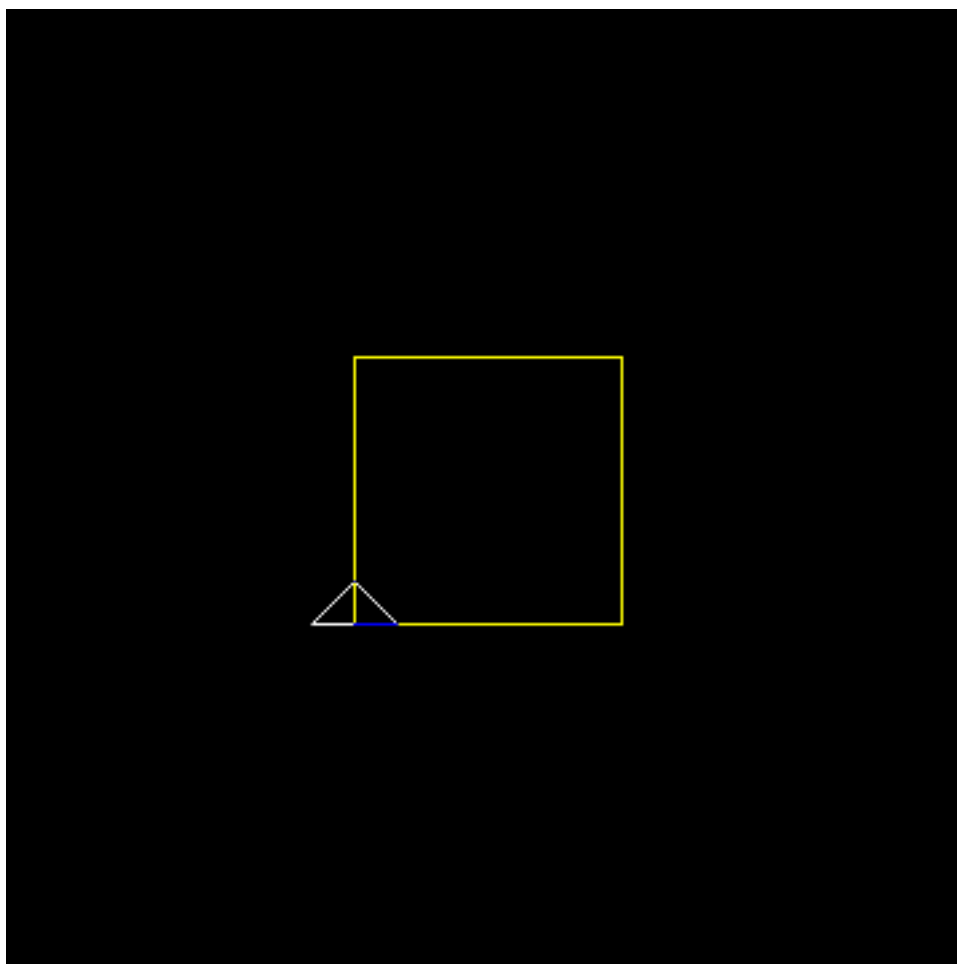
Ovaj način je korisniji jer možeš dobiti **bilo koju** boju, ne samo one s imenima.

23.6 Boja pozadine

Naredba **setscreencolor** (kratko **setsc**) mijenja boju **cijelog platna**, ne samo olovke:

```
setscreencolor "black  
setpencolor "yellow  
repeat 4 [ fd 100 rt 90 ]
```

Žuti kvadrat na crnoj podlozi.



Slika 23.7: Žuti kvadrat na crnoj podlozi

Pazi: `setscreencolor` oboji cijelo platno i pritom **obriše** sve što je dotad nacrtano. Zato ga piši **na početku** programa, prije crtanja.

Olovka koja se ne vidi

Ako postaviš boju olovke jednaku boji pozadine, Tina će i dalje crtati, ali nećeš vidjeti ništa. Platno je na početku bijelo, pa `setpencolor "white` daje nevidljive crte.

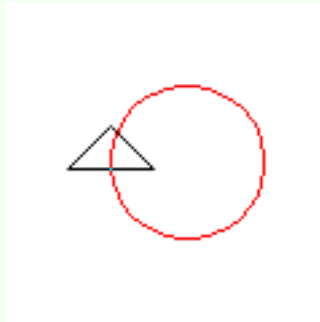
23.7 Vježba

1. Nacrtaj **crveni krug** (36 stranica).
2. Nacrtaj **tri šarena kvadrata** jedan pokraj drugoga (crveni, bijeli, plavi). Pazi na bijelu boju! Što ćeš vidjeti?
3. Nacrtaj **dugu od šest usporednih crta**, svaku u drugoj boji.

Rješenja

1.

```
setpencolor "red  
repeat 36 [ fd 5 rt 10 ]
```



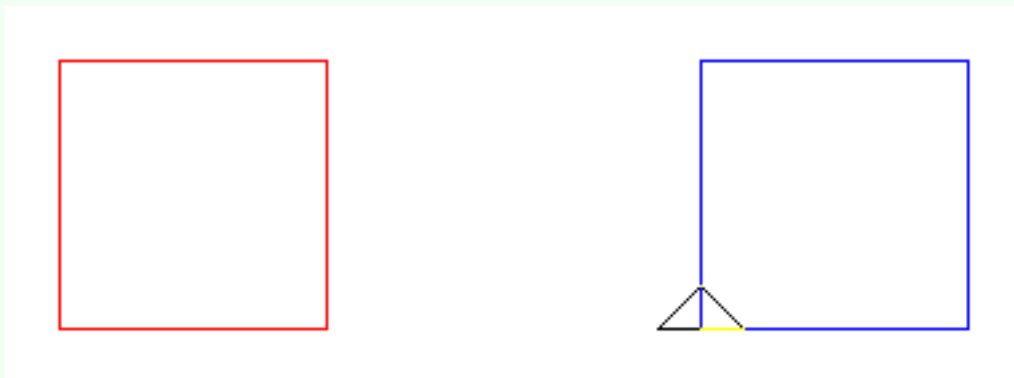
Slika 23.8: Crveni krug

2.

```

setpencolor "red
repeat 4 [ fd 100 rt 90 ]
pu
rt 90
fd 120
lt 90
pd
setpencolor "white
repeat 4 [ fd 100 rt 90 ]
pu
rt 90
fd 120
lt 90
pd
setpencolor "blue
repeat 4 [ fd 100 rt 90 ]

```

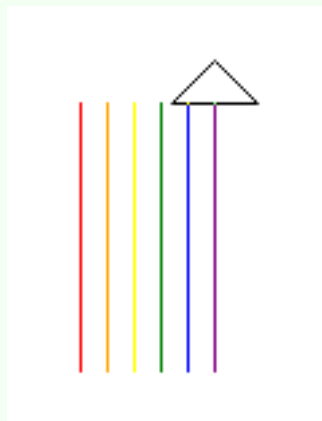


Slika 23.9: Crveni i plavi kvadrat, a bijeli se ne vidi na bijeloj podlozi

Primijeti: bijeli kvadrat u sredini se ne vidi, jer je podloga bijela! Zato između kvadrata idemo fd 120, a ne fd 100. Da se kvadrati dodiruju, lijeva stranica bijelog kvadrata prešla bi preko desne stranice crvenog i obojila je u bijelo. Nova crta uvijek prekrije staru. Želiš li vidjeti i bijeli kvadrat, na sam početak dopiši `setscreencolor "black`.

3.

```
setpencolor "red  
fd 100  
pu bk 100 rt 90 fd 10 lt 90 pd  
setpencolor "orange  
fd 100  
pu bk 100 rt 90 fd 10 lt 90 pd  
setpencolor "yellow  
fd 100  
pu bk 100 rt 90 fd 10 lt 90 pd  
setpencolor "green  
fd 100  
pu bk 100 rt 90 fd 10 lt 90 pd  
setpencolor "blue  
fd 100  
pu bk 100 rt 90 fd 10 lt 90 pd  
setpencolor "purple  
fd 100
```



Slika 23.10: Duga od šest usporednih crta

Sljedeće: [Debljina crte](#) →

Poglavlje 24

Debljina crte

Crta može biti tanka kao kosa ili debela kao prst. Naredba **setpenseize** mijenja **debljinu olovke**. To je blok **setpenseize** iz prve knjige, samo što broj sad pišeš iza naredbe.

24.1 Sintaksa

```
setpenseize 5  
fd 100
```



Slika 24.1: Crta debela 5 piksela

Crta debela 5 piksela.

Što je broj veći, crta je deblja:

```
setpenseize 1  
fd 100
```



Slika 24.2: Vrlo tanka crta

Vrlo tanka crta. Zadana debljina olovke je 1, baš kao u prvoj knjizi, pa je ova crta jednaka onoj bez `setpensize`.

```
setpensize 20  
fd 100
```



Slika 24.3: Vrlo debela crta

Vrlo debela crta, kao od kista.

24.2 Tri kvadrata različitih debljina

```
setpensize 1  
repeat 4 [ fd 50 rt 90 ]  
  
pu fd 60 pd
```

```
setpensize 5  
repeat 4 [ fd 50 rt 90 ]  
  
pu fd 60 pd  
  
setpensize 15  
repeat 4 [ fd 50 rt 90 ]
```

Prvi kvadrat je iscrtan tankom crtom, drugi srednjom, treći debelom.



Slika 24.4: Tri kvadrata različitih debljina crte

24.3 Debljina ostaje dok je ne promijeniš

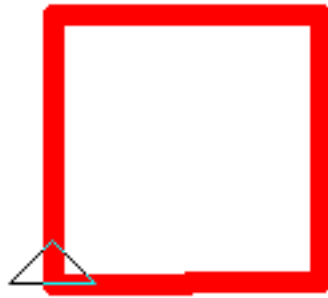
Kao i boja, debljina ostaje **dok je ne promijeniš**. Ako napišeš `setpensize 10`, sve sljedeće crte bit će debele 10, sve dok ne napišeš novi `setpensize`. Ni `cs` je ne vraća na 1.

24.4 Debela boja izgleda drugačije

Probaj kombinirati debljinu i boju:

```
setpensize 8  
setpencolor "red  
repeat 4 [ fd 100 rt 90 ]
```

Crveni kvadrat s debelim rubom. Unutrašnjost ostaje bijela: debela olovka crta samo debeli **obrub**, a ne ispunjen kvadrat.



Slika 24.5: Crveni kvadrat s debelom crtom

Debela olovka radi i s krugom:

```
setpensize 12
setpencolor "blue
repeat 36 [ fd 5 rt 10 ]
```

Plavi krug s debelim obrubom.



Slika 24.6: Plavi krug s debelim rubom

24.5 Kombinacija s mnogo crta

Debela olovka i mnogo crta koje se preklapaju daju zanimljive efekte:

```
setpensize 4
setpencolor "purple
repeat 36 [
  repeat 4 [ fd 60 rt 90 ]
```

```
rt 10
]
```

Cvijet od ljubičastih kvadrata. Svi se preklapaju u sredini, pa je sredina gusto ispunjena. Boje se ne zbrajaju: nova crta samo prekrije staru, pa je i sredina iste ljubičaste boje.



Slika 24.7: Cvijet od ljubičastih kvadrata s debelom olovkom

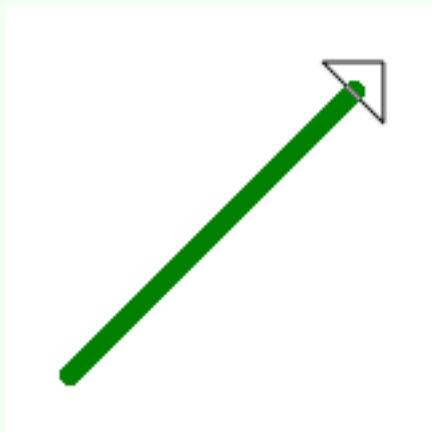
24.6 Vježba

1. Nacrtaj **debelu zelenu** dijagonalnu crtu.
2. Nacrtaj jedan **tanki** mali kvadrat unutar **debelog** velikog kvadrata.
3. Nacrtaj sunce: debeli žuti krug sa zrakama (8 crta koje izlaze iz središta).

Rješenja

1.

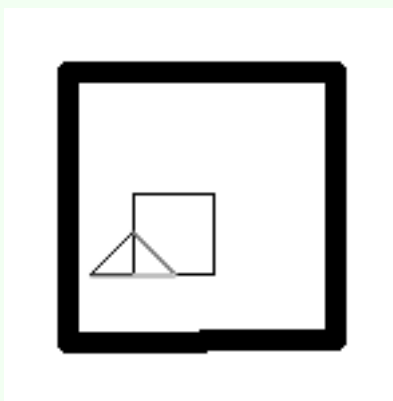
```
setpensize 8
setpencolor "green"
rt 45
fd 150
```



Slika 24.8: Debela zelena dijagonalna crta

2.

```
setpensize 8  
repeat 4 [ fd 100 rt 90 ]  
pu rt 45 fd 35 lt 45 pd  
setpensize 1  
repeat 4 [ fd 30 rt 90 ]
```



Slika 24.9: Tanki kvadrat unutar debelog kvadrata

3.

```
setpensize 6
setpencolor "yellow
repeat 36 [ fd 5 rt 10 ]
pu rt 90 fd 29 lt 90 pd
setpensize 3
setpencolor "orange
repeat 8 [
    fd 60
    bk 60
    rt 45
]
```



Slika 24.10: Sunce: žuti krug s narančastim zrakama

Sljedeće: [Varijable i računanje](#) →

VI. dio

Varijable i računanje

Poglavlje 25

Variable i računanje

Sad krećemo u **pravo programiranje**. Sve dosad smo pisali “konkretne” brojeve: fd 100, repeat 4, rt 90. Sad ćemo brojevima dati **imena**.

Sjećaš se **čarobnih kutijica** iz prve knjige? Blok **set strana to** stavio je broj u kutijicu, a okrugli blok **strana** ga je iz nje uzeo. Programeri ih zovu **varijable**, a sad ćeš ih pisati kodom.

U ovom dijelu ćemo:

- Podsjetiti se **što je varijabla** i zašto je korisna
- Upoznati naredbu **make** koja puni kutijicu
- Naučiti kako **računati** s varijablama (zbrajanje, oduzimanje, množenje, dijeljenje)
- Spremiti cijeli **popis** u jednu kutijicu, kao popise iz prve knjige

Variable će kasnije biti ključne za **procedure s parametrima**, gdje ista naredba radi različito, ovisno o broju koji joj daš.



Sljedeće: [Što je varijabla →](#)

Poglavlje 26

Što je varijabla

Sjećaš se poglavlja **Jedan broj, mnogo blokova** iz prve knjige? Tri jednaka kvadrata htio si povećati. Bez varijable morao bi pronaći i promijeniti **tri** broja, a da zaboraviš samo jedan, crtež bi bio kriv. S kutijicom **strana** mijenjao si **samo jedan** broj, onaj u bloku **set strana to**.

Isti trik radi i u kodu: broj napišeš samo **jednom**, u kutijicu, a svi kvadrati ga uzimaju odande.

26.1 Što je varijabla zapravo

Varijabla je ona ista **kutijica s imenom** iz prve knjige. U kutijicu spremiš broj (ili riječ), pa kasnije koristiš ime kutijice umjesto da svaki put pišeš broj.

Recimo, napravimo varijablu `strana` i u nju spremimo broj 75:

```
make "strana 75
```

`make "strana 75` je blok **set strana to 75** iz prve knjige. Ovaj redak sam ne crta ništa, samo puni kutijicu.

Sada možemo Tini reći da koristi `:strana` umjesto broja 75:

```
make "strana 75  
repeat 4 [ fd :strana rt 90 ]
```

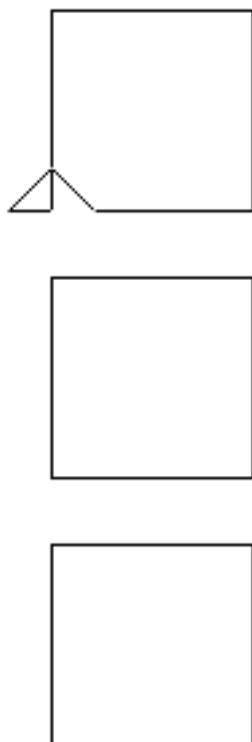
`:strana` je okrugli blok **strana**. Logo na tom mjestu pogleda u kutijicu, nađe **75** i izvrši `fd 75`. Taj si kod u prvoj knjizi mogao pročitati ispod blokova.

26.2 Cijeli primjer

Sad i tri kvadrata u nizu, svi iz iste kutijice:

```
make "strana 75
```

```
repeat 4 [ fd :strana rt 90 ]
pu fd 100 pd
repeat 4 [ fd :strana rt 90 ]
pu fd 100 pd
repeat 4 [ fd :strana rt 90 ]
```



Slika 26.1: Tri kvadrata u nizu, svi sa stranicom 75

Tri kvadrata, svi koriste varijablu `:strana`. Ako želiš veće kvadrate, mijenjaš **samo prvi redak**: `make "strana 120`.

26.3 Pažnja na sintaksu

Varijabla u Logu izgleda čudno. U prvoj knjizi to su bila dva bloka, **set strana to** i okrugli **strana**. U kodu ih razlikuješ po znaku ispred imena:

- **Kad kutijicu puniš**: `"ime` (navodnik ispred), kao blok **set strana to**
- **Kad iz kutijice čitaš**: `:ime` (dvotočka ispred), kao okrugli blok **strana**

Pogledaj još jednom:

```
make "strana 75      ; punimo: "strana
fd :strana           ; čitamo: :strana
```

Zašto razlika? Kad pišeš "strana, govoriš Logu **“ovo je ime, ne čitaj sadržaj”**. Kad pišeš :strana, govoriš **“daj mi sadržaj te kutijice”**.

Najčešća greška

Mladi programeri često pomiješaju ova dva znaka. Ako napišeš fd "strana, konzola će crvenim slovima javiti **fd doesn't like strana as input**, što znači "fd ne voli strana kao ulaz": naredba je dobila riječ umjesto broja. Ako napišeš make :strana 75, javit će **strana has no value** ("strana nema vrijednost"), jer Logo pokušava pročitati kutijicu koja još nema vrijednost.

26.4 Imena varijabli

Ime varijable može biti bilo koja riječ, duga ili kratka. Dobro je dati varijabli **smisljeno ime**:

- :strana je bolje od :x
- :debljina je bolje od :d
- :brojZraka je bolje od :n

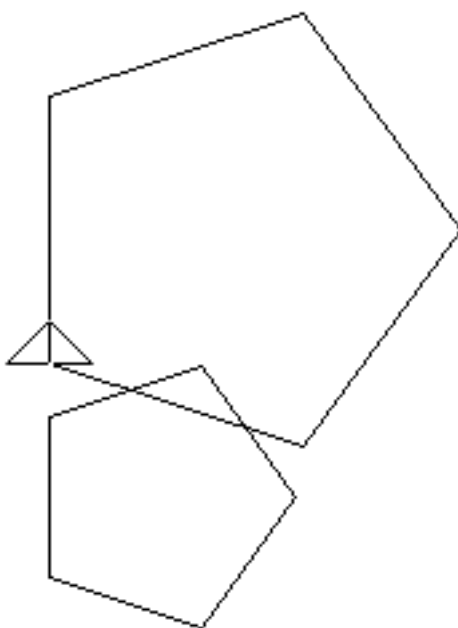
U sasvim kratkim primjerima i mi ćemo ponekad napisati kratko ime, kao :x ili :n. U pravim programima ipak biraj smisljena imena.

Pravila:

- Ime je **jedna riječ**, bez razmaka (broj zraka neće raditi, zato piši brojZraka ili broj_zraka)
- Ime ne smije sadržavati **znakove računanja** (+, -, *, /, =, <, >). Na primjer, :moja-boja Logo čita kao :moja minus boja
- Kao u prvoj knjizi, najbolje je da ime počinje slovom i da nema kvačica: strana, boja, duljina

26.5 Mnogokut s varijablom

```
make "mala 60  
make "velika 100  
  
repeat 5 [ fd :mala rt 72 ]  
pu fd 80 pd  
repeat 5 [ fd :velika rt 72 ]
```



Slika 26.2: Mali i veliki peterokut

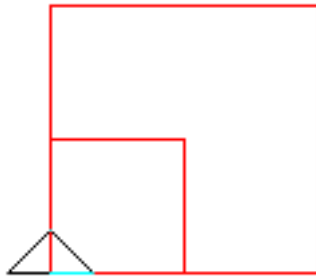
Dva peterokuta: prvi mali (60), drugi veliki (100). Malo se preklapaju, jer je razmak od 80 koraka kratak. Sve je određeno s **dvije varijable**, a ne brojevima razbacanim po cijelom kodu.

26.6 Varijabla može držati i tekst

Kao **kutijica za boju** iz prve knjige, varijabla može držati i ime boje:

```
make "moja_boja "red  
  
setpencolor :moja_boja
```

```
repeat 4 [ fd 100 rt 90 ]  
  
setpencolor :moja_boja  
repeat 4 [ fd 50 rt 90 ]
```



Slika 26.3: Dva crvena kvadrata

Sad oba kvadrata koriste istu boju, a tu boju definiraš **samo na jednom mjestu**.

Sljedeće: [Naredba make →](#)

Poglavlje 27

Naredba make

Naredba **make** stvara varijablu i sprema u nju vrijednost. U prvoj knjizi to je bio blok **set ... to**.

27.1 Sintaksa

```
make "ime vrijednost
```

- **make**: sama naredba
- **"ime**: ime varijable (s navodnikom ispred)
- **vrijednost**: što spremiti (broj, riječ ili popis)

27.2 Primjeri

Spremamo broj:

```
make "duljina 100
```

Spremamo drugi broj:

```
make "kut 60
```

Spremamo ime boje:

```
make "boja "blue
```

(Pažljivo, ovdje su **dva** navodnika: jedan za ime varijable, a drugi za vrijednost koja je riječ.)

27.3 Promjena vrijednosti

Naredba make može i **promijeniti** postojeću varijablu:

```
make "x 50
fd :x          ; Tina ide 50
make "x 100
fd :x          ; Tina ide 100
```

Drugi make "x ne stvara novu kutijicu, nego staru vrijednost u postojećoj kutijici **zami-jeni** novom.

Novu vrijednost možeš izračunati i iz stare. Blok **change strana by 10** iz prve knjige u kodu se piše ovako:

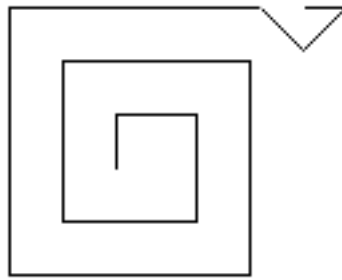
```
make "strana 20
make "strana :strana + 10
fd :strana      ; Tina ide 30
```

Drugi redak čitaj ovako: uzmi broj iz kutijice (:strana), dodaj mu 10 i spremi zbroj natrag u kutijicu "strana.

27.4 Varijable u petljama

Takav make možeš staviti i **unutar petlje**. Sličnu spiralu si u prvoj knjizi već složio od blokova, a sad je pišeš sam:

```
make "duljina 20
repeat 10 [
  fd :duljina
  rt 90
  make "duljina :duljina + 10
]
```

Slika 27.1: Kvadratna spirala s rastućim stranicama

Ovo je ono najljepše: varijable, petlje i računanje zajedno. Pogledaj što se događa:

1. Na početku je u kutijici duljina broj 20
2. Prvi prolaz: fd 20, okret, u kutijici je sad 30
3. Drugi prolaz: fd 30, okret, u kutijici je sad 40
4. Treći prolaz: fd 40, okret, u kutijici je sad 50
5. ...i tako 10 puta

Rezultat? **Spirala!** Svaka stranica je dulja od prethodne.

27.5 Brojač u petlji

Možeš varijablu koristiti i kao **brojač**:

```
make "brojac 0
repeat 5 [
  repeat 4 [ fd 30 rt 90 ]
  pu fd 50 pd
  make "brojac :brojac + 1
]
```

Varijabla :brojac raste od 0 do 5 dok crtamo pet kvadrata u nizu.



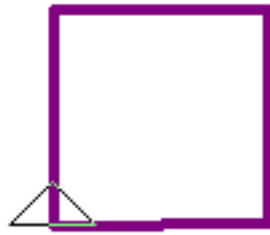
Slika 27.2: Pet kvadrata u nizu

27.6 Više varijabli istovremeno

Možeš imati **više varijabli** u istom programu:

```
make "strana 80
make "boja "purple
make "debljina 4

setpencolor :boja
setpensize :debljina
repeat 4 [ fd :strana rt 90 ]
```



Slika 27.3: Debeli ljubičasti kvadrat definiran varijablama

Tri varijable određuju kako će kvadrat izgledati. Promijeniš li samo jednu (na primjer :debljina), mijenjaš samo jedno svojstvo, a sve ostalo ostaje isto.

27.7 Što ako upotrijebim varijablu koja ne postoji?

```
fd :nepostojeca
```

Ovaj program namjerno ima grešku. Konzola će crvenim slovima javiti **nepostojeca has no value**, što znači “nepostojeca nema vrijednost”: kutijicu s tim imenom nisi nikad napunio. Uvijek treba prvo make, pa onda :.

```
nepostojeca has no value
```

Slika 27.4: Konzola s crvenom porukom nepostojeca has no value

Varijable se gube na kraju programa

Varijable koje stvoriš u jednom pokretanju programa **ne pamte se** za sljedeći put. Kad opet pritisneš  **Pokreni**, Logo krene ispočetka: čisto platno, Tina u sredini i prazne kutijice. Zato make uvijek mora biti u kodu, iznad retka koji kutijicu čita.

Sljedeće: [Računanje](#) →

Poglavlje 28

Računanje

Logo zna **računati**. Možeš zbrajati, oduzimati, množiti i dijeliti brojeve, a rezultate odmah dati naredbama.

28.1 Operatori

Operator	Znači
+	zbrajanje
-	oduzimanje
*	množenje
/	dijeljenje

U prvoj knjizi pisali smo $\times$ i $\div$. U kodu množenje je $*$, a dijeljenje $/$.

28.2 Najjednostavniji primjer

Pomakni Tinu za **60 + 40 = 100** koraka:

```
fd 60 + 40
```



Slika 28.1: Ravna crta od 100 koraka

Tina će napraviti 100 koraka. Logo prvo izračuna $60 + 40$, dobije 100, pa izvrši `fd 100`.

28.3 Računanje s varijablama

Računanje s varijablama je još korisnije:

```
make "strana 50
fd :strana * 2
```

Tina će napraviti $:strana * 2 = 50 * 2 = \mathbf{100}$ koraka.

A ovo:

```
make "strana 50
fd :strana + 30
```

Tina ide $50 + 30 = \mathbf{80}$ koraka.

28.4 Računanje kuta za mnogokut

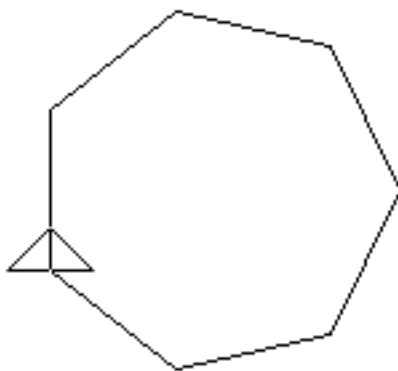
Sjećaš se formule za mnogokut? U prvoj knjizi pisali smo $360 \div N$, a u kodu je to:

Vanjski kut = $360 / \text{broj stranica}$

Sad to možemo zapisati izravno:

```
make "stranice 7
repeat :stranice [
  fd 60
  rt 360 / :stranice
]
```

Sedmerokut **bez** ručnog računanja $360 / 7 = 51.428....$ Logo to napravi sam.



Slika 28.2: Sedmerokut

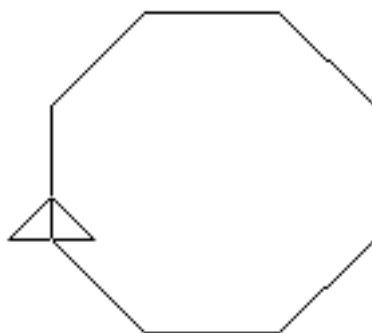
Promijeni make "stranice 7 u make "stranice 9 i dobiješ deveterokut. Promijeni u 12 i dobiješ dvanaesterokut. **Jedan broj određuje sve.**

28.5 Dvije varijable za mnogokut

```
make "stranice 8
make "duljina 50

repeat :stranice [
  fd :duljina
  rt 360 / :stranice
]
```

Dvije varijable određuju i broj stranica i duljinu. Promjena bilo koje vrijednosti odmah se vidi u crtežu.



Slika 28.3: Osmerokut

28.6 Kombinirani računi

Možeš kombinirati više operacija:

```
fd (10 + 5) * 2          ; 30
fd 100 - 20 * 3          ; 100 - 60 = 40
make "x 50
fd :x * 2 + 10           ; 50 * 2 + 10 = 110
```

Pravila prioriteta su uobičajena (kao u matematici): **prvo množenje i dijeljenje**, zatim **zbiranje i oduzimanje**. Zgrade () mogu promijeniti redoslijed.

28.7 Slučajan broj

Logo zna i **birati slučajan broj**. U prvoj knjizi Tina je nasumično birala stvar s popisa, blokom **in list get random** (u kodu je to `pick`, o kojem više u sljedećem poglavlju). Naredba **random** bira nasumičan **broj**: `random 100` daje cijeli broj od 0 do 99. Najveći broj koji može ispasti uvijek je za jedan manji od zadanog.

```
fd random 100
```



Slika 28.4: Crta slučajne duljine, jedan mogući rezultat

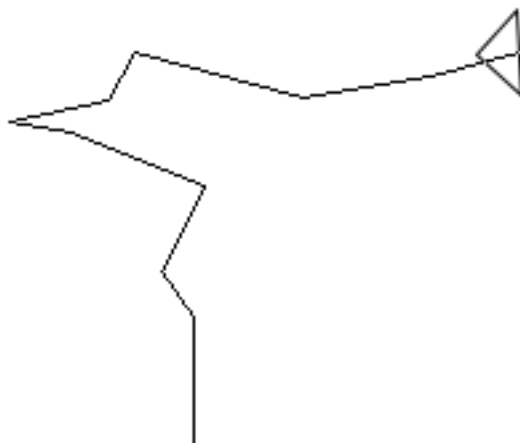
Tina će napraviti **slučajan broj** koraka, negdje između 0 i 99. Pokreni program nekoliko puta: crta je svaki put drugačije duljine.

```
rt random 360
```

Slučajan okret.

Slučajni brojevi su super za “umjetnost”: svaki put kad pokreneš program, dobiješ drugačiji crtež:

```
repeat 10 [
  fd (random 50) + 20
  rt random 360
]
```



Slika 28.5: Jedan mogući rezultat, svaki put je drugačiji

Deset kratkih crta u nasumičnim smjerovima, svaka različite duljine (između 20 i 69).

Zagrade oko random

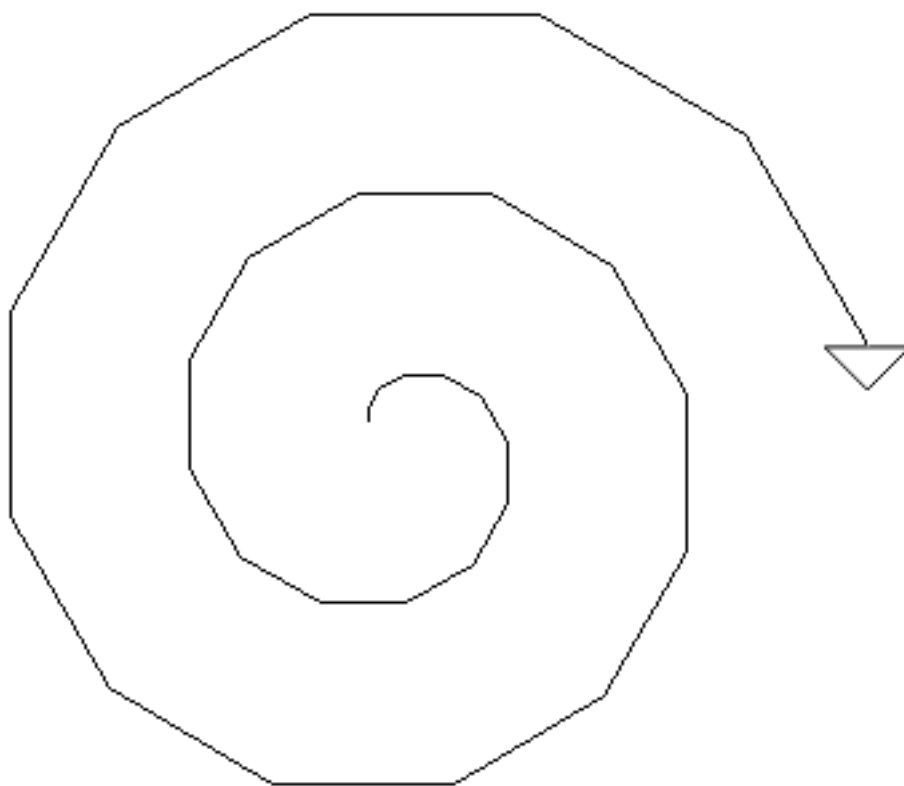
Naredba random kao ulaz uzme **cijeli račun** iza sebe. Bez zagrada random 50 + 20 Logo čita kao random 70: broj od 0 do 69, a ne od 20 do 69. Zagradama kažeš: prvo izaberi slučajan broj manji od 50, pa mu dodaj 20. Isto vrijedi i za druge naredbe koje računaju, na primjer sqrt.

28.8 Spirala s rastućom stranicom

Spojimo sve naučeno u **spiralu**:

```
make "duljina 5
repeat 30 [
  fd :duljina
  rt 30
  make "duljina :duljina + 3
]
```


30 stranica, a nakon svake okret od 30 stupnjeva. Prva stranica ima 5 koraka, a svaka sljedeća 3 više, pa zadnja ima $5 + 29 \cdot 3 = 92$ koraka. Rezultat: lijepa **spiral**a.



Slika 28.6: Spirala s rastućom stranicom

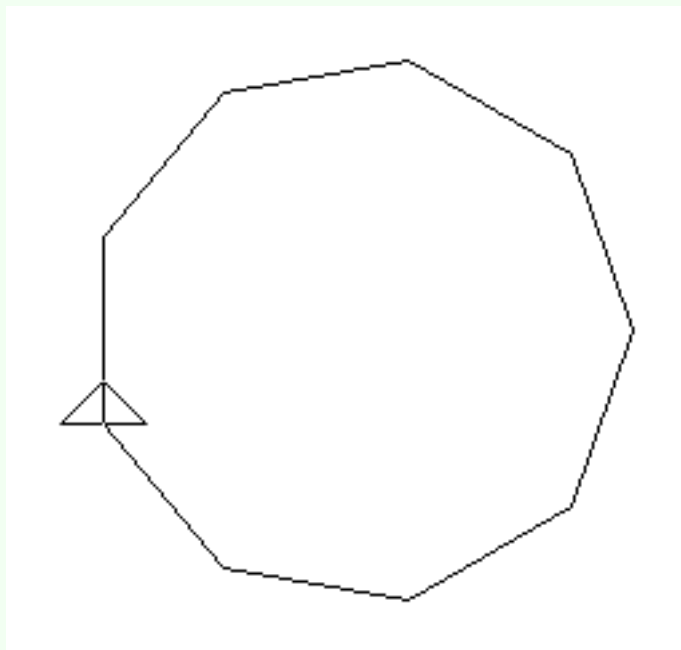
28.9 Vježba

1. Nacrtaj **deveterokut sa stranicom 70** koristeći make i $360 /$:stranice.
2. Nacrtaj “kvadrat” kojem je svaka stranica **slučajne** duljine.
3. Nacrtaj spiralnu od trokuta, u kojoj je svaki trokut malo veći od prethodnog.

Rješenja

1.

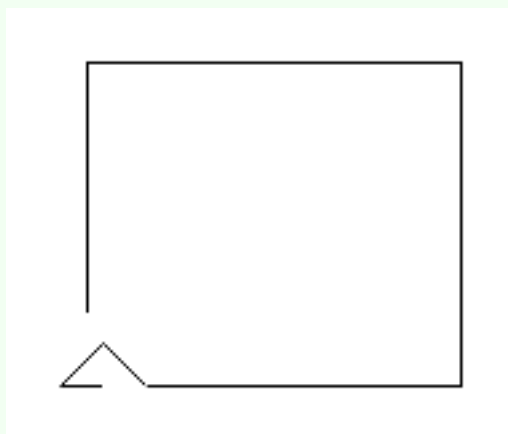
```
make "stranice 9
make "duljina 70
repeat :stranice [
  fd :duljina
  rt 360 / :stranice
]
```



Slika 28.7: Deveterokut sa stranicom 70

2.

```
repeat 4 [
  fd 50 + random 100
  rt 90
]
```

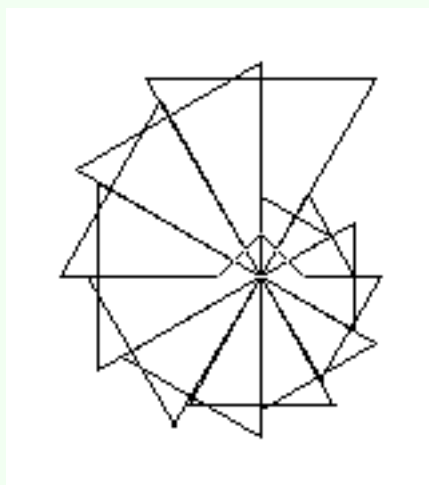


Slika 28.8: Krivi kvadrat sa slučajnim stranicama, jedan mogući rezultat

Kvadrat je “krivi” jer su mu stranice različitih duljina, i to je u redu. Zgrade ne trebaju, jer random 100 stoji na kraju.

3.

```
make "duljina 30
repeat 12 [
  repeat 3 [ fd :duljina rt 120 ]
  rt 30
  make "duljina :duljina + 5
]
```



Slika 28.9: Spirala od sve većih trokuta

Sljedeće: [Popis u kutijici →](#)

Poglavlje 29

Popis u kutijici

U prvoj knjizi Tina je radila i s **popisima**. Blokom **create list with** složio si popis boja i spremio ga u kutijicu, blokom **in list get** uzimao si stvari s popisa, a blokom **for each** Tina je prošla kroz cijeli popis, stvar po stvar. Sve to možeš napisati i kodom.



29.1 Napravi popis

Popis boja duge spremaš ovako:

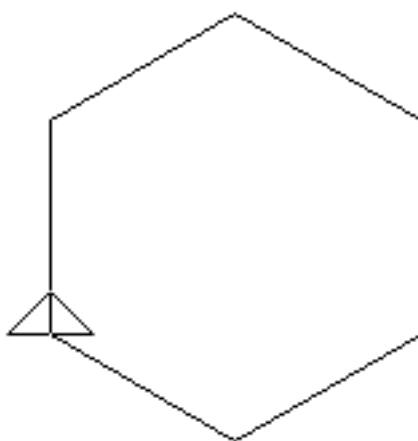
```
make "boje [red orange yellow green blue purple]
```

To je blok **set boje to** s blokom **create list with** i šest boja. Popis pišeš u **uglatim zagradama** [], a stvari odvajaš razmakom. Pazi: imena boja unutar zagrada pišeš **bez navodnika**, jer zagrade Logu već kažu da su unutra riječi. S navodnicima, ["red "blue], konzola bi kasnije javila **setpencolor doesn't like "red as input**: navodnik bi postao dio riječi, baš kao kod setpencolor "blue".

29.2 Koliko je stvari na popisu?

Blok **length of** u kodu je count. Ubaci ga u repeat, pa Tina ponavlja onoliko puta koliko je boja na popisu:

```
make "boje [red orange yellow green blue purple]
repeat count :boje [ fd 80 rt 360 / count :boje ]
```



Slika 29.1: Šesterokut, jer je na popisu šest boja

Na popisu je 6 boja, pa count :boje daje 6 i Tina nacrtat šesterokut. Dodaš li na popis sedmu boju, dobit ćeš sedmerokut, a redak s repeat ne moraš ni taknuti.

29.3 Prva, zadnja, bilo koja

Blok **in list get** imao je izbornik s izborima **first**, **last**, **#** i **random**. U kodu je svaki izbor posebna riječ:

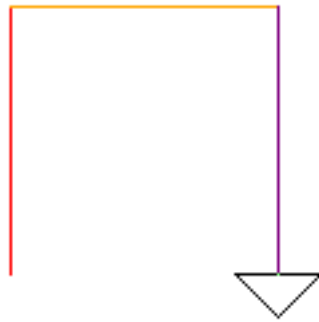
Blok u prvoj knjizi	Kod	Za popis :boje daje
in list boje get first	first :boje	red
in list boje get last	last :boje	purple
in list boje get # 2	item 2 :boje	orange
in list boje get random	pick :boje	nasumičnu boju

Stvar koju uzmeš odmah daješ naredbi, na primjer naredbi setpencolor:

```

make "boje [red orange yellow green blue purple]
setpencolor first :boje
fd 100 rt 90
setpencolor item 2 :boje
fd 100 rt 90
setpencolor last :boje
fd 100

```



Slika 29.2: Tri stranice kvadrata: crvena, narančasta i ljubičasta

Prva stranica je crvena (prva boja), druga narančasta (boja broj 2), a treća ljubičasta (zadnja boja).

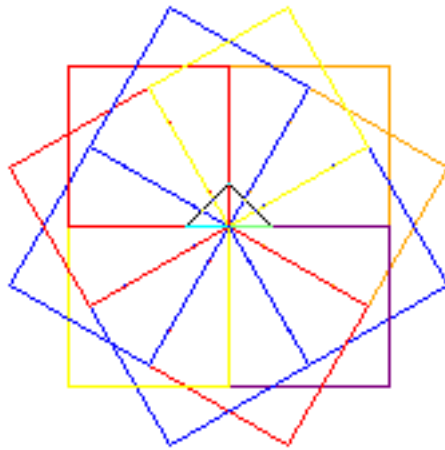
29.4 Nasumično s popisa

Najzabavniji izbor bio je **random**. U kodu se zove `pick`: Tina zatvori oči i izvuče jednu stvar s popisa. Ovo je nasumični cvijet iz prve knjige:

```

make "boje [red orange yellow green blue purple]
repeat 12 [
  setpencolor pick :boje
  repeat 4 [ fd 60 rt 90 ]
  rt 30
]

```



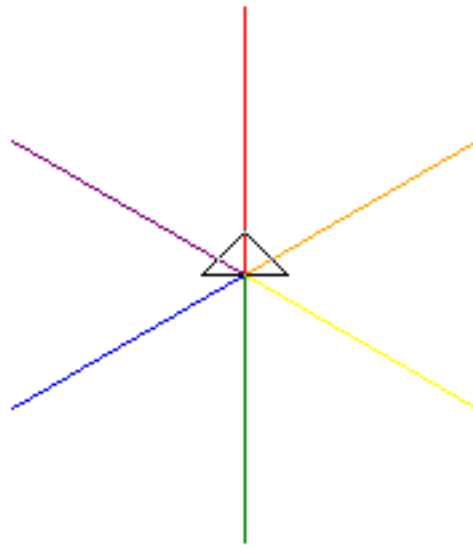
Slika 29.3: Cvijet od dvanaest kvadrata u nasumičnim bojama, jedan mogući rezultat

Pokreni ga nekoliko puta: boje su svaki put drugačije. Sjećaš se naredbe `random` iz poglavlja [Računanje](#)? `random 6` izvuče broj, a `pick :boje` izvuče stvar s popisa.

29.5 Za svaku boju

Blok **for each** u kodu je `foreach`. Ovo je šarena zvijezda iz prve knjige, u kraćoj verziji s popisom:

```
make "boje [red orange yellow green blue purple]
foreach :boje [ [boja]
  setpencolor :boja
  fd 100
  bk 100
  rt 60
]
```



Slika 29.4: Šest zraka iz sredine, svaka u drugoj boji duge

Iza popisa dolaze uglate zagrade s naredbama, a na njihovu početku, u svojim zagradama, stoji [boja]. To je **kutijica** s bloka for each: ono malo ime **j** koje si u prvoj knjizi preimenovao u **boja**. Tina uzme prvu stvar s popisa, stavi je u kutijicu boja i izvrši naredbe. Onda uzme drugu, pa treću, sve do kraja popisa. Unutra kutijicu čitaš kao i svaku drugu: :boja.

29.6 Popis brojeva

Na popisu mogu biti i brojevi. Kutijicu ovaj put zovemo d, kao duljina:

```
make "duljine [50 100 150]
foreach :duljine [ [d]
  fd :d
  rt 90
]
```




Slika 29.5: Tri crte, od 50, 100 i 150 koraka, s okretima od 90

Isti crtež dobiješ i s `item`, samo pišeš više:

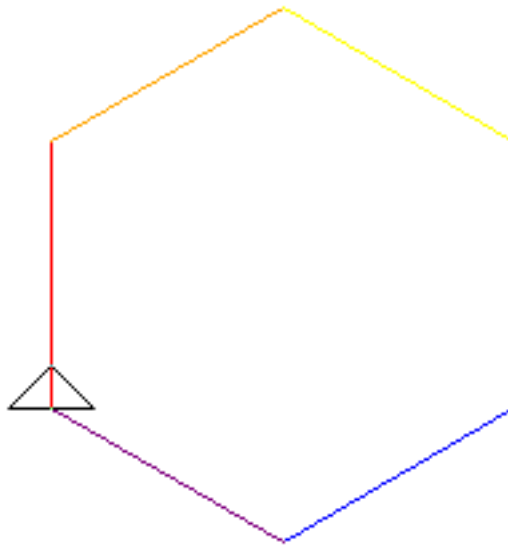
```
make "duljine [50 100 150]
fd item 1 :duljine rt 90
fd item 2 :duljine rt 90
fd item 3 :duljine rt 90
```

`foreach` umjesto tebe ide redom: prva stvar, druga, treća.

29.7 Boje redom s `repcount`

Petlja `repeat` ima svoj mali brojač, `repcount`: u prvom prolazu je 1, u drugom 2, i tako dalje. S njim `item` uzima boje redom:

```
make "boje [red orange yellow green blue purple]
repeat 6 [
  setpencolor item repcount :boje
  fd 100
  rt 60
]
```



Slika 29.6: Šesterokut u bojama duge

To je šareni šesterokut iz poglavlja **Boja crte**, ali u šest redaka umjesto osamnaest.

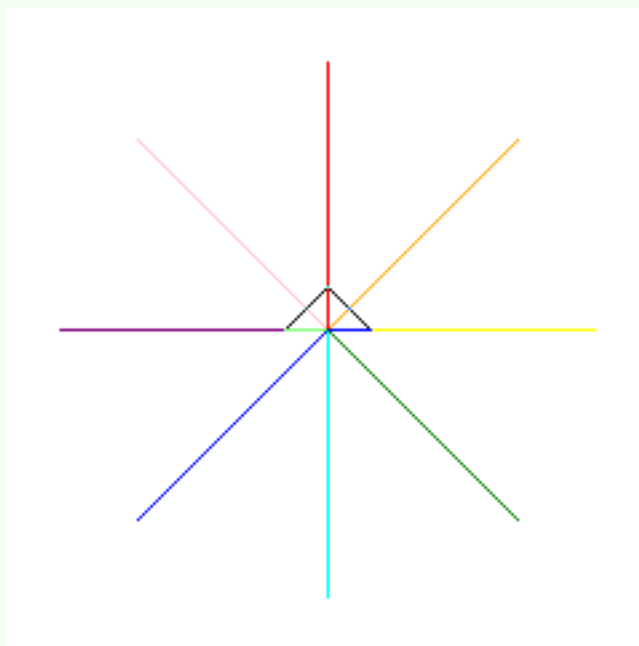
29.8 Vježba

1. Dodaj na popis boja još cyan i pink, pa nacrtaj sunce s 8 zraka. Kut neka Tina izračuna sama, pomoću count.
2. Napravi popis od tri debljine olovke, [2 6 12]. Nacrtaj deset crta u krug tako da Tina prije svake crte nasumično izabere debljinu.
3. Nacrtaj pet kvadrata iz istog ugla, sa stranicama s popisa [20 40 60 80 100].

Rješenja

1.

```
make "boje [red orange yellow green cyan blue purple pink]
foreach :boje [ [boja]
  setpencolor :boja
  fd 100
  bk 100
  rt 360 / count :boje
]
```

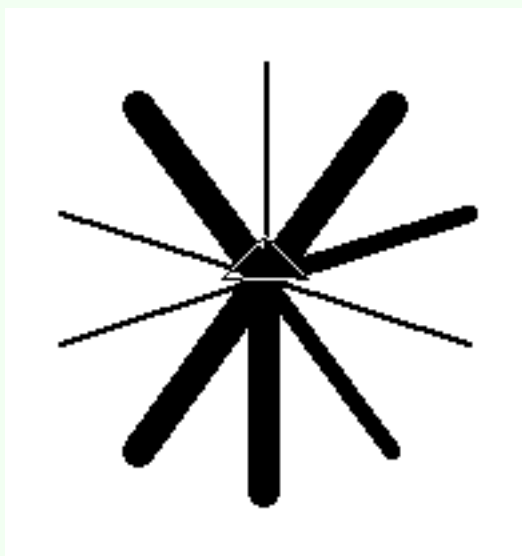


Slika 29.7: Sunce od osam zraka u osam boja

Na popisu je 8 boja, pa je okret $360 / 8 = 45$.

2.

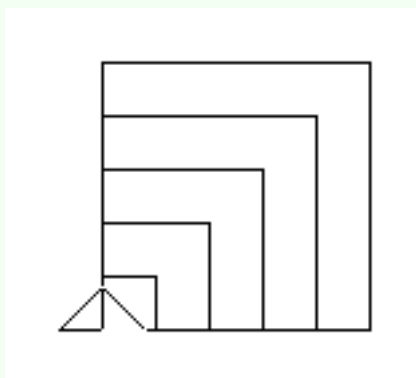
```
make "debljine [2 6 12]
repeat 10 [
  setpense pick :debljine
  fd 80
  bk 80
  rt 36
]
```



Slika 29.8: Deset zraka nasumične debljine, jedan mogući rezultat

3.

```
make "duljine [20 40 60 80 100]
foreach :duljine [ [d]
  repeat 4 [ fd :d rt 90 ]
]
```



Slika 29.9: Pet kvadrata iz istog ugla, sve veći

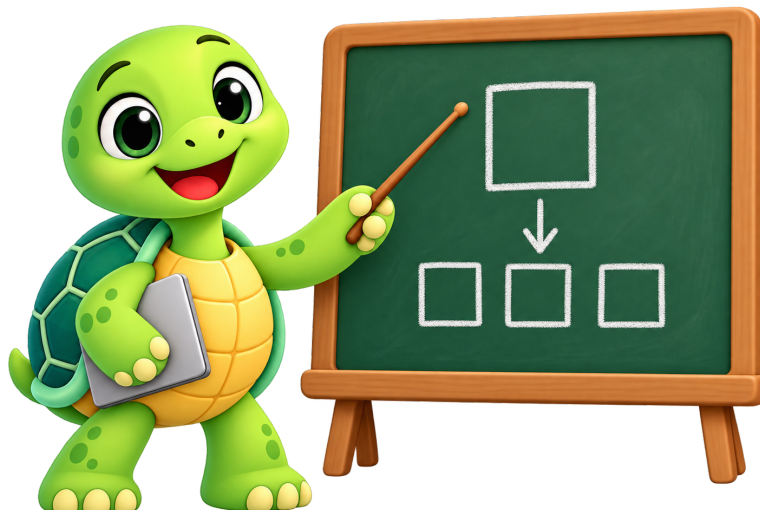
Sljedeće: [Procedure](#) →

VII. dio

Procedure

Poglavlje 30

Procedure



Sad dolazimo do najmoćnijeg alata u Logu: **procedure**.

U prvoj knjizi si Tinu naučio novu riječ **kvadrat** vlastitim blokom **to kvadrat**. Programeri takav blok zovu **procedura** (u prvoj knjizi rekli smo i funkcija). U kodu je pišeš s to i end.

Procedura je **vlastita naredba** koju sam izmisliš. Daješ joj ime i govoriš Logu što treba napraviti kad je netko pozove. Onda je koristiš kao bilo koju drugu naredbu, kao fd, rt ili repeat.

Zamisli da napraviš naredbu kvadrat koja crta kvadrat. Sad bilo gdje napišeš kvadrat i dobit ćeš kvadrat. **Ne moraš ponovno pisati** repeat 4 [fd 100 rt 90].

U ovom dijelu ćemo:

- Vidjeti **što je procedura** i zašto je korisna
- Napisati našu **prvu proceduru** s to i end
- Naučiti procedurama dodavati **parametre** (na primjer kvadrat 50), baš kao što si u prvoj knjizi bloku dodao ulaz

Sljedeće: [Što je procedura →](#)

Poglavlje 31

Što je procedura

Pogledaj ovu naredbu:

```
fd 100
```

To je naredba **fd**. Tina je razumije i radi nešto. Tko ju je napisao? Ljudi koji su napravili Logo. Ona je **ugrađena** u jezik.

A što ako želiš svoju vlastitu naredbu? Recimo, naredbu **kvadrat** koja crta kvadrat?

Takvu naredbu već si napravio u prvoj knjizi, ljubičastim blokom **to kvadrat**. U kodu se ona zove **procedura**. Ti je definiraš i daješ joj ime, a ona postaje dio jezika, kao da je oduvijek bila tu.

31.1 Što znači definirati proceduru

Definirati znači **objasniti Logu što naredba treba raditi**. Pišeš je ovako:

```
to ime_procedure
  naredbe
end
```

- **to**: počinje definicija (na bloku iz prve knjige pisalo je **to do something**, “da napraviš nešto”)
- **ime_procedure**: kako se procedura zove
- **naredbe**: što Tina treba napraviti
- **end**: kraj definicije

To je veliki blok **to kvadrat** iz prve knjige: to je njegov vrh, a end njegovo dno. Sve između je ono što si slagao unutar bloka.

31.2 Naša prva procedura

Definiramo proceduru kvadrat:

```
to kvadrat
  repeat 4 [ fd 100 rt 90 ]
end
```

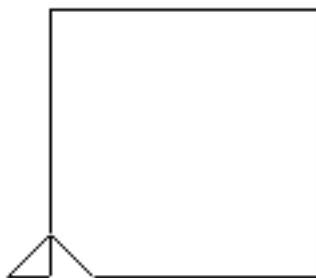
Pritisni ► **Pokreni**. Što se dogodilo? **Ništa!** Baš kao u prvoj knjizi: Tina je naučila novu riječ, ali joj još nisi rekao da je napravi.

Zato ispod definicije (iza retka end) dopiši kvadrat:

```
to kvadrat
  repeat 4 [ fd 100 rt 90 ]
end

kvadrat
```

Pokreni i Tina crta kvadrat! Točno kao da si napisao repeat 4 [fd 100 rt 90]. Sad je kvadrat **prava naredba**, a Tina zna što ta riječ znači.



Slika 31.1: Kvadrat nacrtan pozivom procedure

31.3 Pozivanje procedure

Kad jednom definiraš proceduru, možeš je **pozvati** koliko god puta želiš:

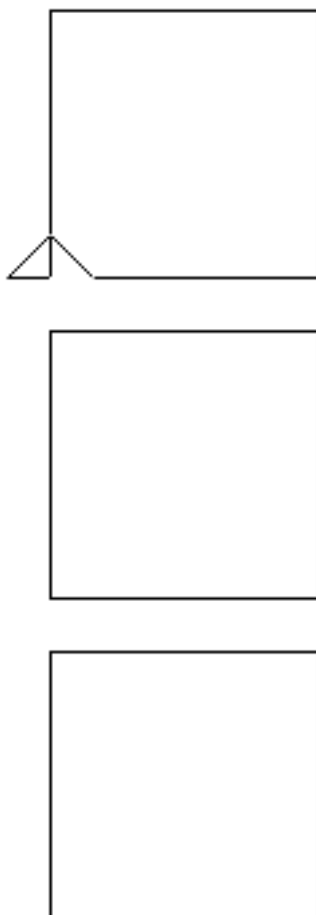
```
to kvadrat
  repeat 4 [ fd 100 rt 90 ]
end

kvadrat
pu fd 120 pd
kvadrat
```



```
pu fd 120 pd  
kvadrat
```

Tri kvadrata u nizu. Pišeš kvadrat umjesto cijele petlje svaki put.



Slika 31.2: Tri kvadrata u nizu

Kao u prvoj knjizi, program ima dva dijela. **Definicija** od `to` do `end` objašnjava novu riječ, a **pozivi** ispod nje kažu Tini “napravi to sad”. Definiciju napišeš jednom, a poziv koliko god puta želiš.

31.4 Zašto procedure?

Tri velika razloga:

31.4.1 Manje pisanja

Umjesto `repeat 4 [ fd 100 rt 90 ]` šest puta, samo napišeš kvadrat šest puta.

31.4.2 Lakše se mijenja

Ako želiš da kvadrati budu veći, u programu iznad mijenjaš **samo definiciju**:

```
to kvadrat
  repeat 4 [ fd 200 rt 90 ]
end
```

Sad svaki kvadrat crta veliki kvadrat, a pozive ispod definicije nisi ni taknuo.

31.4.3 Čitljivije

Što je ovo?

```
repeat 4 [ fd 100 rt 90 ]
pu fd 120 pd
repeat 3 [ fd 80 rt 120 ]
```

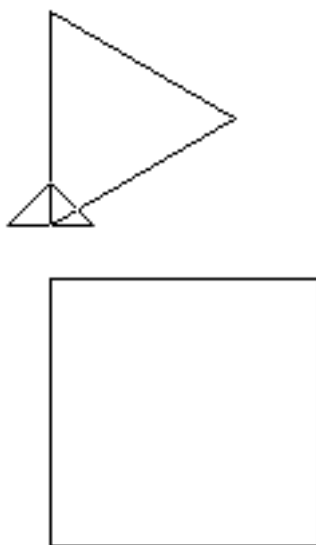
Treba ti nekoliko sekundi da shvatiš. A ovo?

```
to kvadrat
  repeat 4 [ fd 100 rt 90 ]
end

to trokut
  repeat 3 [ fd 80 rt 120 ]
end

kvadrat
pu fd 120 pd
trokut
```

Oba programa crtaju isto. Ali u drugom su definicije na vrhu, a zadnja tri retka čitaju se kao rečenica: kvadrat, pomak, trokut. Odmah je jasno što Tina radi.



Slika 31.3: Kvadrat i trokut

31.5 Procedure pozivaju procedure

Procedura može u sebi koristiti **drugu proceduru**! Kao u prvoj knjizi, **kuca** koristi **kvadrat** i **trokut**:

```
to kvadrat
  repeat 4 [ fd 50 rt 90 ]
end

to trokut
  repeat 3 [ fd 50 rt 120 ]
end

to kuca
  kvadrat
  fd 50
  rt 30
  trokut
end

kuca
```

Pozoveš **kuca** jednom i dobiješ cijelu kuću: kvadrat nacrtava zidove, a trokut krov.



Slika 31.4: Kuća nacrtana procedurom

31.6 Pravila o imenima

Imena procedura slijede ista pravila kao imena varijabli:

- Neka počinje slovom, tako je najsigurnije
- Smije sadržavati slova, brojeve i donju crtu, kao obojeni_kvadrat
- Ne smije sadržavati razmak ni znakove računanja (+, -, *, /, =, <, >)
- Najbolje je bez kvačica, kao u prvoj knjizi: kuca, a ne kuća

Za procedure vrijedi još jedno pravilo: **ime ne smije biti kao postojeća naredba**. Ako napišeš to `fd`, konzola javi **fd is already defined** (“`fd` već postoji”), a odmah zatim i **I don’t know how to end** (“ne znam kako se radi `end`”), jer Logo više ne zna čemu služi zadnji redak.

31.7 Gdje se procedure pamte?

Svaki put kad pritisneš  **Pokreni**, Logo kreće ispočetka: čisto platno, Tina u sredini i prazna memorija. Zatim čita tvoj kod od prvog retka. Zato definicija procedure mora biti u kodu **iznad poziva**, i to pri svakom pokretanju.

Ako obrišeš definiciju i ostaviš samo kvadrat, konzola javi **I don’t know how to kvadrat** (“ne znam kako se radi kvadrat”).

Naredba `cs` briše samo crtež, a procedure ostaju do kraja tog pokretanja. Nakon pokretanja proceduru možeš isprobati i u **retku za naredbe** ispod konzole: upiši `kvadrat` i pritisni Enter.

Sljedeće: [Prva procedura →](#)

Poglavlje 32

Prva procedura

Sad ćemo zajedno napisati nekoliko procedura, korak po korak. Svaki put napiši definiciju, a ispod nje (iza retka end) poziv, pa pokreni.

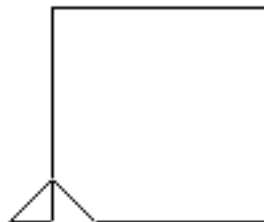
32.1 Procedura: kvadrat

Ovu smo već vidjeli, pa krenimo s njom:

```
to kvadrat
  repeat 4 [ fd 80 rt 90 ]
end

kvadrat
```

Dobit ćeš mali kvadrat sa stranicom 80.



Slika 32.1: Kvadrat sa stranicom 80

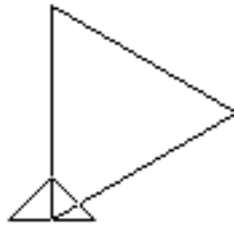
32.2 Procedura: trokut

Po istom uzorku, definicija pa ispod nje poziv:

```
to trokut
  repeat 3 [ fd 80 rt 120 ]
end

trokut
```

Dobit ćeš trokut.

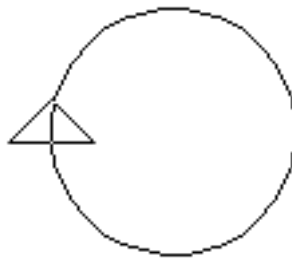


Slika 32.2: Trokut sa stranicom 80

32.3 Procedura: krug

```
to krug  
  repeat 36 [ fd 8 rt 10 ]  
end  
  
krug
```

Dobit ćeš krug. Tina ga crta hodajući po rubu, malo po malo, kao u dijelu **Ponavljanje**.



Slika 32.3: Krug

(Postoji i gotova naredba `circle2 50` iz prve knjige. Ovdje krug radimo sami, da vidiš kako nastaje.)

32.4 Sada sve zajedno

Stavi sve tri procedure u isti program, jednu ispod druge, a pozive na dno:

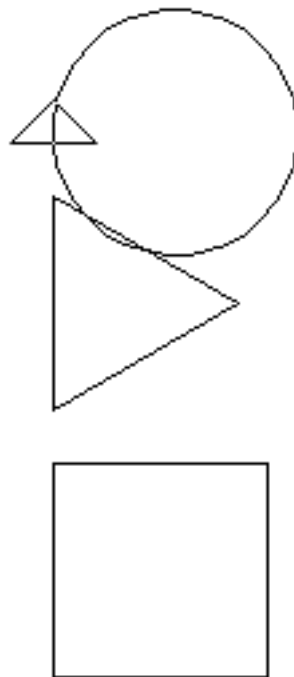
```
to kvadrat
  repeat 4 [ fd 80 rt 90 ]
end

to trokut
  repeat 3 [ fd 80 rt 120 ]
end

to krug
  repeat 36 [ fd 8 rt 10 ]
end

kvadrat
pu fd 100 pd
trokut
pu fd 100 pd
krug
```

Tri oblika u nizu: kvadrat dolje, trokut iznad njega, a krug na vrhu.



Slika 32.4: Kvadrat, trokut i krug u nizu

32.5 Procedura: pomakni

Vidiš li koliko često pišemo pu fd 100 pd? Tina se **često** mora pomaknuti bez crtanja. Od toga možemo napraviti proceduru:

```
to pomakni
  pu
  fd 100
  pd
end
```

Dopiši je ispod ostalih definicija, a u pozivima umjesto pu fd 100 pd napiši pomakni. Cijeli program sad izgleda ovako:

```
to kvadrat
  repeat 4 [ fd 80 rt 90 ]
end

to trokut
  repeat 3 [ fd 80 rt 120 ]
end

to krug
  repeat 36 [ fd 8 rt 10 ]
end

to pomakni
  pu
  fd 100
  pd
end

kvadrat
pomakni
trokut
pomakni
krug
```

Crtež je isti kao prije, ali kod je mnogo čišći.

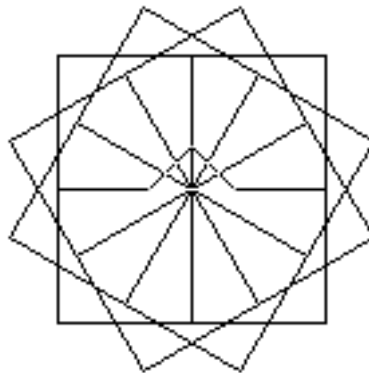
32.6 Procedura: cvijet od kvadrata

Sjećaš se ugniježdene petlje iz dijela **Ponavljanje** koja je crtala cvijet od kvadrata? Pretvorimo je u proceduru:

```
to cvijet
  repeat 12 [
    repeat 4 [ fd 50 rt 90 ]
    rt 30
  ]
end

cvijet
```

Sada je cvijet tvoja vlastita naredba!



Slika 32.5: Cvijet od 12 kvadrata

32.7 Komentar u proceduri

U proceduru možeš pisati i **komentare**: tekst koji Logo preskače, ali tebi pomaže da se sjetiš što kod radi. Komentar počinje znakom `;` i traje do kraja retka:

```
to cvijet
  ; Crta cvijet od 12 kvadrata u krugu
  repeat 12 [
    repeat 4 [ fd 50 rt 90 ] ; jedan kvadrat
    rt 30 ; pomak za sljedeću "laticu"
```

```
    ]  
end  
  
cvijet
```

Kad sutra otvoriš ovaj kod, komentari će ti reći **što kod radi**, a da ne moraš čitati svaki redak.

32.8 Što sada?

Sve naše procedure dosad rade **istu stvar svaki put**. kvadrat uvijek crta stranicu 80, a cvijet uvijek 12 kvadrata. Što ako želimo kvadrat 50 ili kvadrat 200, kvadrata različitih veličina?

Sjećaš se iz prve knjige? Svom bloku si **zupčanikom** dodao **ulaz**. U kodu se ulaz zove **parametar**, a o njemu je sljedeće poglavlje.

Sljedeće: [Procedura s parametrima →](#)

Poglavlje 33

Procedura s parametrima

Tinine ugrađene naredbe primaju **brojeve**: fd 100, rt 90. Tako i tvoje vlastite procedure mogu primiti brojeve! Rupica za takav broj zove se **parametar**.

U prvoj knjizi si svom bloku **zupčanikom** dodao **ulaz** velicina. Parametar je taj isti ulaz. U kodu ga pišeš iza imena procedure, na primjer to kvadrat :strana.

33.1 Sintaksa

```
to ime_procedure :parametar
  naredbe koje koriste :parametar
end
```

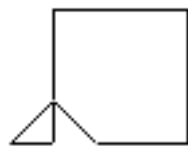
Iza imena procedure dodaješ **:ime_parametra**. Unutar procedure koristiš taj parametar **isto kao varijablu**, s : ispred. Parametar je kutijica koju Logo pri svakom pozivu napuni brojem iz poziva.

33.2 Kvadrat s parametrom

Umjesto da kvadrat uvijek crta sa stranicom 80, neka prima broj. Ispod definicije (iza retka end) dopiši poziv kvadrat 50:

```
to kvadrat :strana
  repeat 4 [ fd :strana rt 90 ]
end

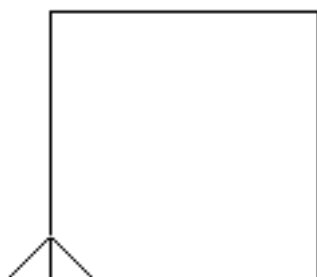
kvadrat 50
```



Slika 33.1: Mali kvadrat

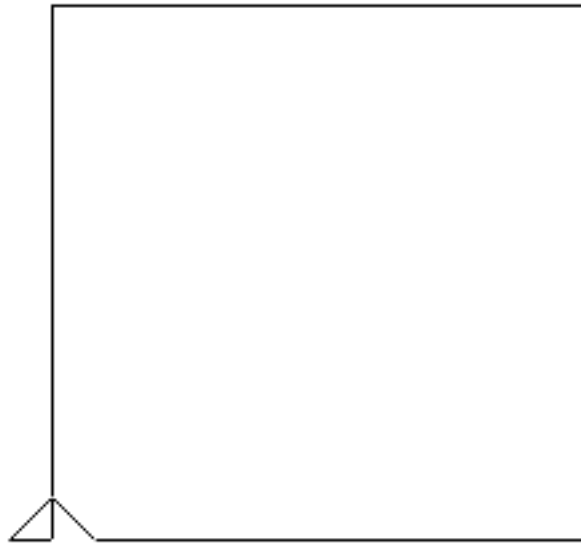
Mali kvadrat sa stranicom 50. Logo je broj 50 stavio u kutijicu :strana, pa je fd :strana zapravo fd 50.

Sad zadnji redak promijeni u kvadrat 100 i pokreni:



Slika 33.2: Srednji kvadrat

Srednji kvadrat. A s kvadrat 200:



Slika 33.3: Veliki kvadrat

Veliki kvadrat.

Jedna procedura, beskonačno mnogo veličina!

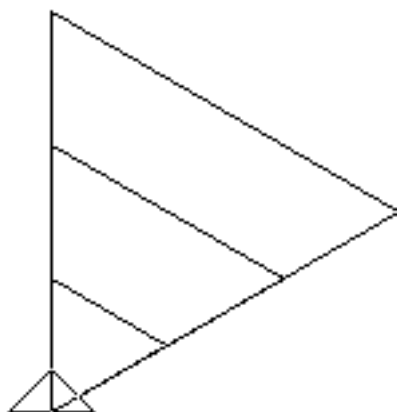
33.3 Trokut s parametrom

Isto možeš i s trokutom. Ispod definicije dopiši tri poziva:

```
to trokut :strana
  repeat 3 [ fd :strana rt 120 ]
end

trokut 50
trokut 100
trokut 150
```

Tri trokuta različitih veličina, sva tri iz iste točke.



Slika 33.4: Tri ugniježđena trokuta iz iste točke

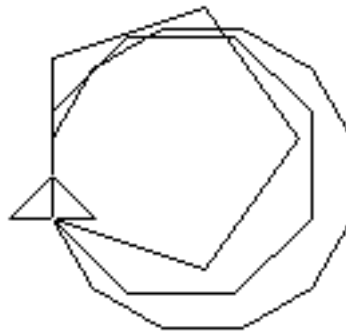
33.4 Više parametara

Procedura može imati **više parametara**. Napravimo proceduru mnogokut koja prima i broj stranica i duljinu stranice:

```
to mnogokut :stranice :duljina
  repeat :stranice [
    fd :duljina
    rt 360 / :stranice
  ]
end

mnogokut 5 60      ; peterokut sa stranicom 60
mnogokut 8 40      ; osmerokut sa stranicom 40
mnogokut 12 30     ; dvanaesterokut sa stranicom 30
```

Brojevi u pozivu idu redom: prvi ide u :stranice, a drugi u :duljina. Jedna procedura crta **bilo koji** pravilan mnogokut. To je vrlo moćno.



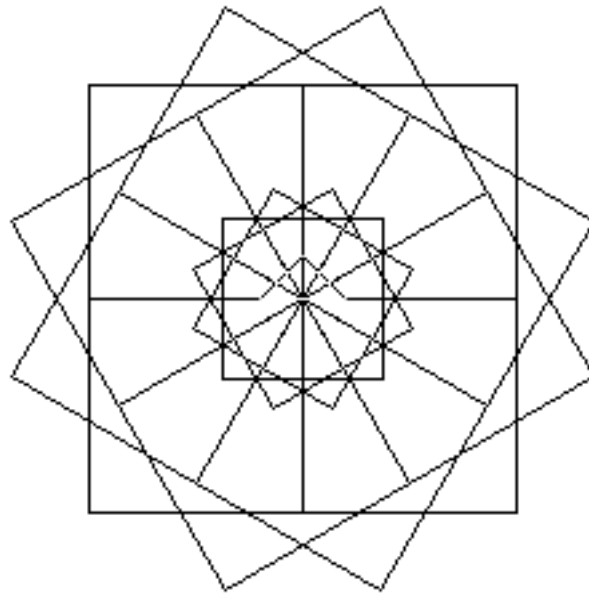
Slika 33.5: Peterokut, osmerokut i dvanaesterokut iz iste točke

33.5 Cvijet s veličinom

Ovaj cvijet s ulazom već poznaješ iz prve knjige. Ovdje kvadrat pišemo odmah unutar cvijeta:

```
to cvijet :velicina
  repeat 12 [
    repeat 4 [ fd :velicina rt 90 ]
    rt 30
  ]
end

cvijet 30      ; mali cvijet
cvijet 80      ; veliki cvijet
```



Slika 33.6: Mali i veliki cvijet iz iste sredine

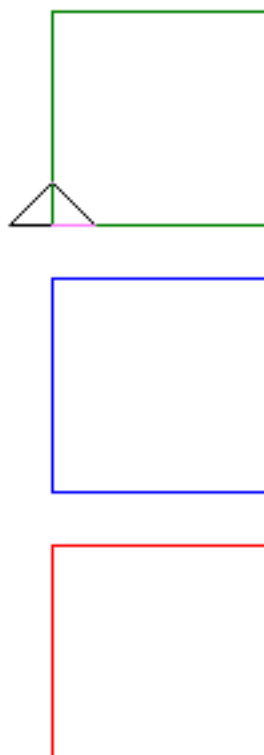
33.6 Boja kao parametar

Parametri ne moraju biti samo brojevi. Mogu biti i **imena boja**. Kao blok s **dva ulaza** iz prve knjige, procedura može primiti broj i boju:

```
to obojeni_kvadrat :strana :boja
  setpencolor :boja
  repeat 4 [ fd :strana rt 90 ]
end

obojeni_kvadrat 80 "red
pu fd 100 pd
obojeni_kvadrat 80 "blue
pu fd 100 pd
obojeni_kvadrat 80 "green
```

Tri kvadrata jednake veličine, tri različite boje. Sve s istom procedurom.



Slika 33.7: Crveni, plavi i zeleni kvadrat u nizu

33.7 Procedura unutar procedure, sad još bolje

Sad kad imamo parametre, procedure mogu lijepo međusobno raditi. Sjećaš se iz prve knjige kako su se kuće u selu naginjale? Nakon krova Tina gleda 30 stupnjeva udesno. Zato na kraj procedure kuca dodajemo lt 30, a bk :sirina je vrati dolje, na početak kuće:

```
to kvadrat :strana
  repeat 4 [ fd :strana rt 90 ]
end

to trokut :strana
  repeat 3 [ fd :strana rt 120 ]
end

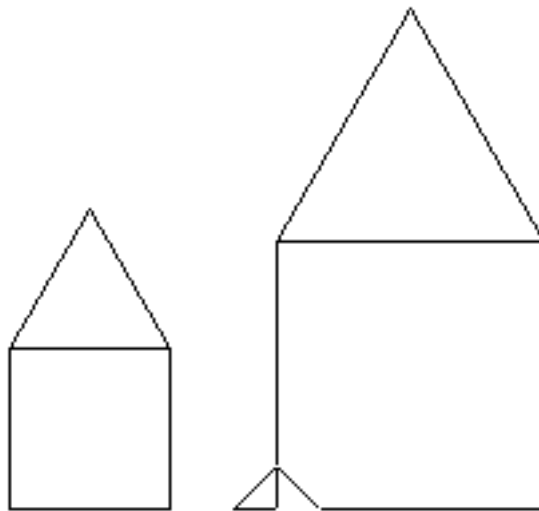
to kuca :sirina
  kvadrat :sirina
  fd :sirina
```

```
    rt 30
    trokut :sirina
    lt 30
    bk :sirina
end

kuca 60
rt 90
pu fd 100 pd
lt 90
kuca 100
```

Procedura kuca prima jedan parametar :sirina i prosljeđuje ga procedurama kvadrat i trokut. Zadnja dva retka procedure vraćaju Tinu u donji lijevi ugao kuće, okrenutu prema gore. Između kuća Tina se okrene udesno, prošeta 100 koraka s podignutom olovkom i opet se okrene prema gore. Zato se i druga kuća crta uspravno.

Mala kuća i velika kuća, jedna pokraj druge, sve s istom procedurom.



Slika 33.8: Mala i velika kuća jedna pokraj druge, obje uspravne

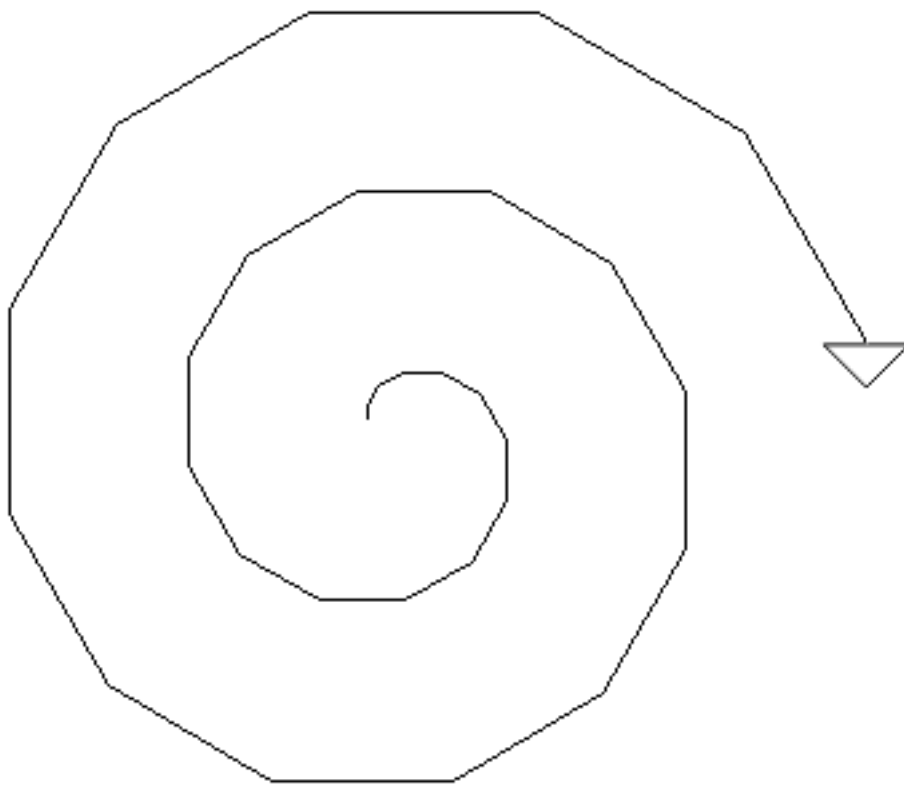
33.8 Procedura koja koristi računanje

Parametre možeš i **koristiti u računima** unutar procedure:

```
to spirala :pocetak :rast :broj
  make "duljina :pocetak
  repeat :broj [
    fd :duljina
    rt 30
    make "duljina :duljina + :rast
  ]
end

spirala 5 3 30
```

Procedura spirala prima početnu duljinu, za koliko svaka crta raste i koliko crta treba nacrtati. Daješ joj brojeve i ona crta spiralu po želji.



Slika 33.9: Spirala nacrtana procedurom

Probaj i druge brojeve u zadnjem retku: spirala 10 5 50 daje veću spiralu, a spirala 1 1 100 gustu spiralu od puno malih koraka. (Ako napišeš oba poziva, druga spirala počinje tamo gdje je prva završila, pa se crteži preklapaju.)

33.9 Vježba

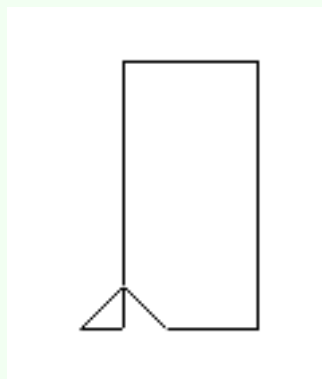
1. Napiši proceduru pravokutnik :duljina :sirina koja crta pravokutnik.
2. Napiši proceduru obojeni_krug :boja :debljina koja crta krug zadane boje i debljine.
3. Napiši proceduru niz_kvadrata :koliko :strana koja crta :koliko kvadrata u nizu, svaki sa zadanom stranicom.

Rješenja

1.

```
to pravokutnik :duljina :sirina
  repeat 2 [
    fd :duljina
    rt 90
    fd :sirina
    rt 90
  ]
end

pravokutnik 100 50
```

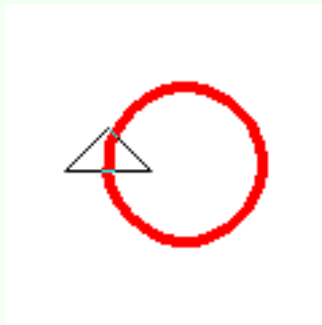


Slika 33.10: Pravokutnik

2.

```
to obojeni_krug :boja :debljina
  setpencolor :boja
  setpensize :debljina
  repeat 36 [ fd 5 rt 10 ]
end

obojeni_krug "red 4
```

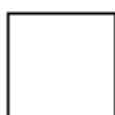


Slika 33.11: Crveni debeli krug

3.

```
to niz_kvadrata :koliko :strana
  repeat :koliko [
    repeat 4 [ fd :strana rt 90 ]
    pu fd :strana + 20 pd
  ]
end

niz_kvadrata 5 40
```



Slika 33.12: Pet kvadrata u nizu

Sljedeće: [Uvjeti](#) →

VIII. dio

Uvjeti

Poglavlje 34

Uvjeti



Sve dosad je Tina radila **uvijek isto**: što god joj kažeš, to izvrši. Ali pravi programi mogu **donositi odluke**.

“Ako je ovaj broj veći od 100, nacrtaj kvadrat. Inače, nacrtaj trokut.”

To se zove **uvjet**. U Logu je to naredba **if** (na hrvatskom **ako**).

Na kraju prve knjige obećali smo ti da ćeš naučiti kako Tina **razmišlja i odlučuje**. Sad je vrijeme za to! Uvjete nisi slagao blokovima, pa je ovo za tebe potpuno nova čarolija.

U ovom dijelu ćemo:

- Upoznati naredbu **if**
- Naučiti raditi **usporedbe** brojeva (veći, manji, jednak)
- Napisati **slučajnu šetnju**, malu igru koja koristi uvjete

S uvjetima tvoji programi više nisu samo “crteži”: ponašaju se različito, ovisno o brojevima koje dobiju.

Sljedeće: [Naredba if](#) →

Poglavlje 35

Naredba if

Naredba **if** kaže Logu: “ako je nešto istina, onda napravi ovo”.

35.1 Sintaksa

```
if uvjet [  
    naredbe  
]
```

- **if**: sama naredba
- **uvjet**: nešto što je ili **istina** ili **laž**
- **[...]**: naredbe koje se izvrše **samo ako je uvjet istinit**

35.2 Najjednostavniji primjer

```
make "x 50  
  
if :x > 30 [  
    fd 100  
]
```



Slika 35.1: Crta od 100 koraka, jer je uvjet istinit

Što se događa:

1. Postavljamo :x na 50
2. Provjeravamo: je li :x > 30? **Da, 50 je veće od 30. Istina!**
3. Izvršavamo: fd 100

A sad:

```
make "x 10

if :x > 30 [
    fd 100
]
```



Slika 35.2: Prazno platno, samo Tina u sredini

1. :x je 10
2. Je li 10 > 30? **Ne!**
3. Naredbe unutar [...] se **preskaču**

Tina ne radi ništa.

35.3 Što unutra može stajati

U [...] može biti **bilo koji** Logo kod: jedna naredba, više naredbi ili čak nova petlja:

```
make "n 5

if :n > 3 [
    repeat :n [
        fd 50
        rt 360 / :n
    ]
]
```

Ako je :n veći od 3, Tina nacrtat mnogokut. Inače ne radi ništa.

35.4 ifelse: ako, inače

Ponekad želiš nešto napraviti ako je uvjet istinit, **a nešto drugo** ako je laž. Za to služi naredba **ifelse**:

```
ifelse uvjet [  
    naredbe ako istina  
] [  
    naredbe ako laž  
]
```

Primjer:

```
make "n 7  
  
ifelse :n > 5 [  
    fd 100  
] [  
    fd 50  
]
```



Slika 35.3: Crta od 100 koraka, jer je 7 veće od 5

Ako je :n > 5, Tina ide 100 koraka, a inače 50. Ovdje je :n jednako 7, pa ide 100. Promijeni prvi redak u make "n 3 i pokreni opet: sad ide samo 50 koraka.

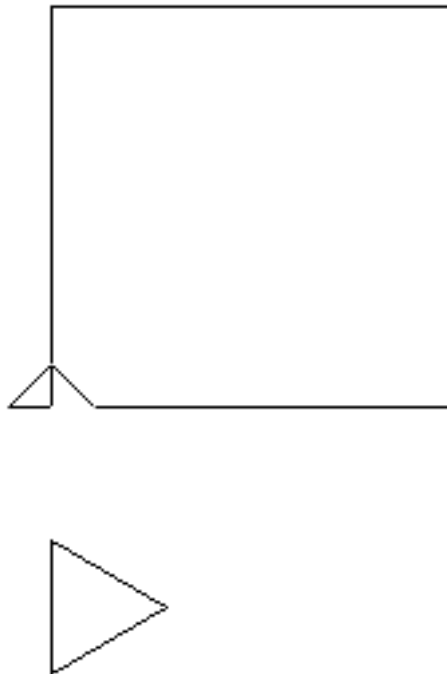
35.5 ifelse u proceduri

Posebno su korisne procedure koje se ponašaju različito, ovisno o parametru:

```
to oblik :velicina
  ifelse :velicina > 100 [
    repeat 4 [ fd :velicina rt 90 ]
  ] [
    repeat 3 [ fd :velicina rt 120 ]
  ]
end

oblik 50          ; mali trokut
pu fd 100 pd
oblik 150         ; veliki kvadrat
```

Procedura oblik crta trokut ako je :velicina ≤ 100 (manja od 100 ili jednaka 100), a kvadrat ako je veća.

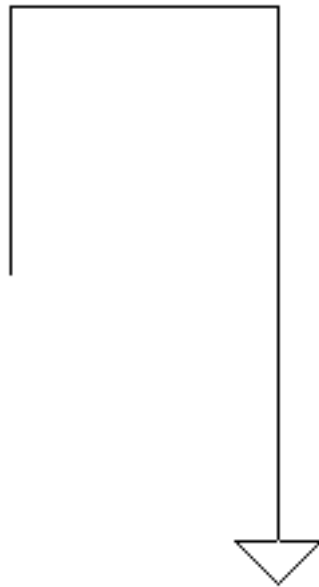


Slika 35.4: Mali trokut i veliki kvadrat

35.6 Kombiniranje sa slučajnim brojem

Spojimo ifelse i random da Tina “donosi odluke”:

```
repeat 10 [
  ifelse (random 2) = 0 [
    fd 50
  ] [
    rt 90
  ]
]
```



Slika 35.5: Jedan mogući rezultat slučajne šetnje

Rezultat: Tina hoda na slučajan način, kao da luta.

random 2 daje ili 0 ili 1. Tina svaki put baca “kovanicu”:

- Ako padne 0, ide naprijed
- Ako padne 1, okreće se

Zagrade oko random 2 su obavezne. Bez njih Logo prvo izračuna $2 = 0$ i javi grešku **random doesn't like false as input** (“random ne voli false kao ulaz”).

Slučajne odluke

Ovo je super za “umjetničke” crteže. Svaki put kad pokreneš program, dobiješ drugačiji rezultat.

35.7 Vježba

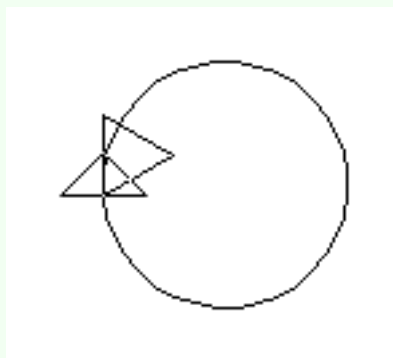
1. Napiši proceduru velicina :n koja crta veliki krug ako je :n > 50, a mali trokut ako je :n <= 50.
2. Probaj: Tina u svakom prolazu petlje ide naprijed za 30, a svaki treći put okreće se za 90 stupnjeva udesno. (Pomoć: koristi brojač.) Što je nacrtala?

Rješenja

1.

```
to velicina :n
  ifelse :n > 50 [
    repeat 36 [ fd 8 rt 10 ]
  ] [
    repeat 3 [ fd 30 rt 120 ]
  ]
end

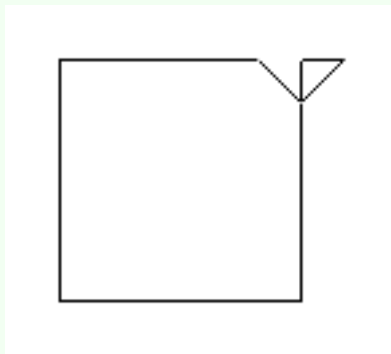
velicina 80
velicina 30
```



Slika 35.6: Veliki krug i mali trokut

2.

```
make "brojac 0
repeat 30 [
  fd 30
  make "brojac :brojac + 1
  if :brojac = 3 [
    rt 90
    make "brojac 0
  ]
]
```



Slika 35.7: Kvadrat stranice 90: Tina skrene nakon svaka tri koraka

Iznenadjenje: kvadrat! Tri koraka po 30 daju stranicu od 90, a Tina se uvijek okreće udesno, pa hoda ukrug po istom kvadratu. Kad bi se okretala naizmjenice desno i lijevo, dobio bi stepenice.

Sljedeće: [Usporedbe](#) →

Poglavlje 36

Usporedbe

U prethodnom poglavlju vidjeli smo `:x > 30` i `(random 2) = 0`. Ti znakovi su **operatori usporedbe**. Oni daju **istinu** ili **laž**, a Logo ih piše na engleskom: `true` i `false`.

36.1 Tablica operatora

Operator	Znači
<code>=</code>	jednako
<code><></code>	nije jednako (različito)
<code><</code>	manje od
<code>></code>	veće od
<code><=</code>	manje ili jednako
<code>>=</code>	veće ili jednako

36.2 Primjeri

```
make "x 10

if :x = 10 [ fd 50 ]      ; istina (x je jednak 10)
if :x = 20 [ fd 50 ]      ; laž (x nije 20)
if :x < 20 [ fd 50 ]      ; istina (10 < 20)
if :x > 5 [ fd 50 ]       ; istina (10 > 5)
if :x <> 11 [ fd 50 ]      ; istina (x nije 11)
if :x >= 10 [ fd 50 ]     ; istina (10 >= 10)
if :x <= 5 [ fd 50 ]      ; laž (10 nije <= 5)
```

Pet uvjeta je istinito, pa Tina pet puta ide 50 koraka. Na platnu je jedna crta od 250 koraka.

36.3 Procedura: parno ili neparno?

Možemo provjeriti je li broj **paran** koristeći **ostatak pri dijeljenju**. Naredba **remainder** računa ostatak. Da ga vidimo, ispisujemo ga naredbom **print** u **konzolu**, dolje desno:

```
print remainder 10 2    ; ispiše 0 (10 je djeljivo s 2)
print remainder 11 2    ; ispiše 1
print remainder 7 3     ; ispiše 1
```

```
0
1
1
```

Slika 36.1: Konzola s brojevima 0, 1 i 1, svaki u svom retku

Bez print Logo ne zna što bi s izračunatim brojem. Napišeš li samo remainder 10 2, konzola crvenim slovima javi **You don't say what to do with 0** ("ne kažeš što da radim s 0").

Broj je paran ako mu je ostatak pri dijeljenju s 2 jednak nuli. U kodu to pišemo (remainder :n 2) = 0. Zagrade su obavezne, jer bi inače Logo prvo izračunao 2 = 0.

```
to provjeri :n
  ifelse (remainder :n 2) = 0 [
    ; paran, pa nacrtaj kvadrat
    repeat 4 [ fd 30 rt 90 ]
  ] [
    ; neparan, pa nacrtaj trokut
    repeat 3 [ fd 30 rt 120 ]
  ]
end

provjeri 4          ; kvadrat (paran)
pu fd 50 pd
provjeri 7          ; trokut (neparan)
pu fd 50 pd
provjeri 10         ; kvadrat
```



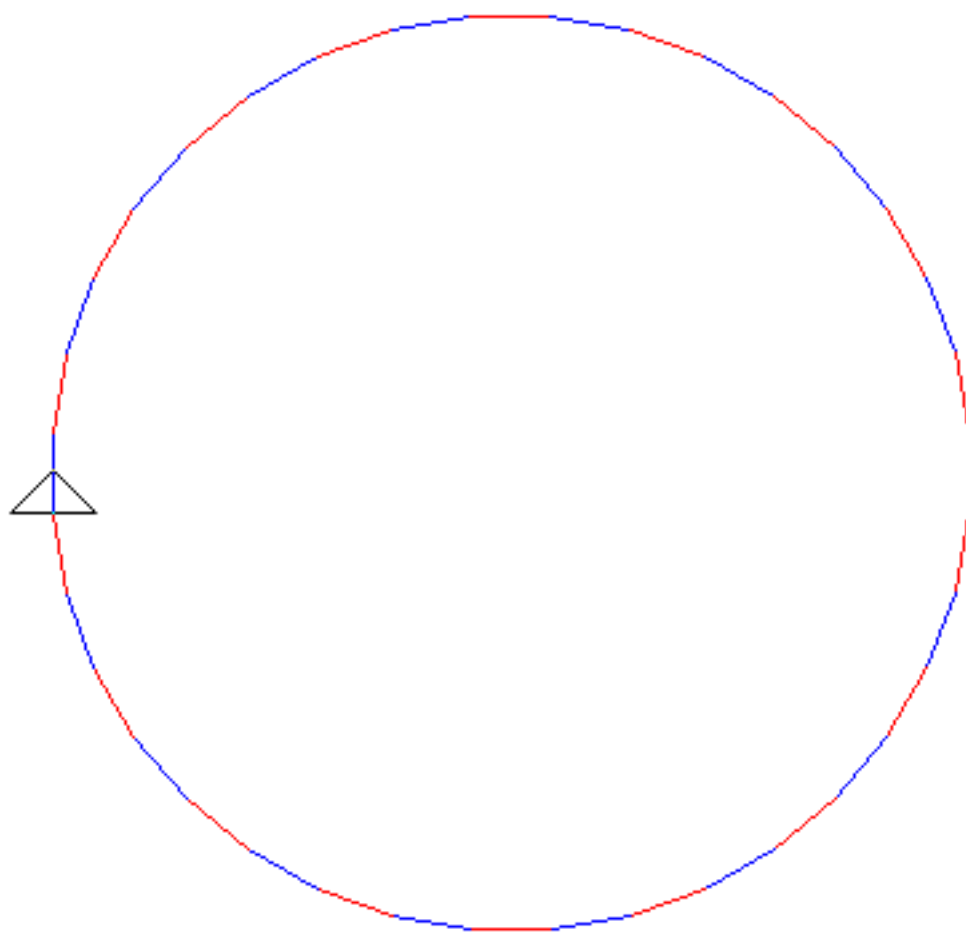
Slika 36.2: Kvadrat, trokut i kvadrat: parni brojevi daju kvadrate

36.4 Krug u dvije boje

Pravimo veliki krug od 36 stranica u kojem **parne** stranice crtamo crveno, a **neparne** plavo:

```
make "i 1
repeat 36 [
  ifelse (remainder :i 2) = 0 [
    setpencolor "red
  ] [
    setpencolor "blue
  ]
  fd 30
  rt 10
  make "i :i + 1
]
```

Brojač `:i` broji stranice: 1, 2, 3... Kad je paran, olovka je crvena, a kad je neparan, plava. Dobiješ krug od 36 crtica koje su naizmjenice crvene i plave.



Slika 36.3: Crveno-plavi kružni uzorak

36.5 Više usporedbi: and i or

Što ako uvjet treba kombinirati **dva** uvjeta?

- **and**: oba uvjeta moraju biti istinita
- **or**: dovoljno je da je jedan istinit

```
make "x 50

if and :x > 10 :x < 100 [
  fd 50
]
```

Ovo radi fd 50 samo ako je :x između 10 i 100 (oba uvjeta istinita).

```
make "y 5

if or :y < 0 :y > 100 [
  fd 50
]
```

Ovo radi fd 50 ako je :y izvan raspona od 0 do 100 (bilo koji od dva uvjeta istinit). S make "y 5 nijedan uvjet nije istinit, pa Tina ne crta ništa. Promijeni prvi redak u make "y 150 i pokreni opet.

Sintaksa and/or

U Logu and i or stoje **prije** uvjeta, ne između. Pišeš and uvjet1 uvjet2, a ne uvjet1 and uvjet2. To može djelovati čudno, jer je Logo tu malo drugačiji od matematike.

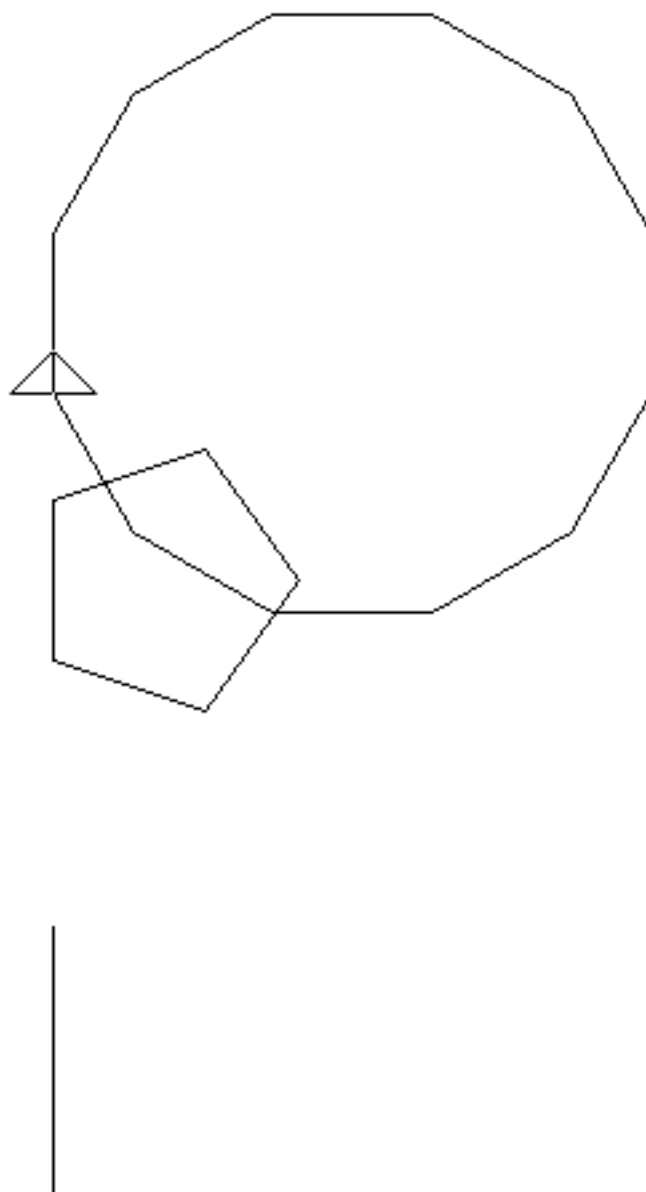
36.6 Procedura: granična vrijednost

Često se procedura treba ponašati **drugačije** kad parametar prijeđe granicu:

```
to mnogokut :n
  ifelse :n < 3 [
    fd 100 ; manje od 3, samo ravna crta
  ] [
    repeat :n [
      fd 60
      rt 360 / :n
    ]
  ]
end

mnogokut 1 ; ravna crta
pu fd 100 pd
mnogokut 5 ; peterokut
pu fd 100 pd
mnogokut 12 ; dvanaesterokut
```

Ako broj stranica bude manji od 3 (a takav mnogokut ne postoji), procedura napravi nešto drugo.



Slika 36.4: Crtā, peterokut i dvanaesterokut

36.7 Provjera: pozitivan broj

Korisno za procedure koje oĉekuju **pozitivan** broj:

```
to nacrtaj :duljina
  if :duljina <= 0 [
    ; loš parametar, ne radi ništa
  ]
  if :duljina > 0 [
```

```
        fd :duljina
    ]
end
```

To radi, ali s ifelse je pametnije i kraće:

```
to nacrtaj :duljina
  ifelse :duljina > 0 [
    fd :duljina
  ] [
    ; loš broj, ispiši poruku
    print [Duljina mora biti veća od nule]
  ]
end

nacrtaj 50
nacrtaj -5      ; ispisat će poruku, neće crtati
```



Duljina mora biti veća od nule

Slika 36.5: Crta od 50 koraka, a u konzoli poruka Duljina mora biti veća od nule

Naredba print ispisuje i tekst: riječi u uglatim zagradama pojave se u konzoli, ali bez zagrada. To je korisno za poruke i za traženje grešaka.

Sljedeće: [Slučajna šetnja](#) →

Poglavlje 37

Slučajna šetnja

Sad kad imaš sve alate (varijable, procedure i uvjete), možemo napisati pravu **malu igru**.

Zove se **Slučajna šetnja**. Ovu igru Tina igra sama: ti pritisneš  **Pokreni** i gledaš kamo će otići. Tina luta po platnu i na svakom potezu **slučajno** odlučuje:

- Hoda li naprijed
- Okreće li se

Svaki nekoliko poteza slučajno izabere i **novu boju**. I sve to ponavlja **50 puta**.

37.1 Pravila igre

1. Tina kreće iz sredine
2. Na svakom potezu baci “kovanicu”: 0 ili 1
3. Ako je 0, ide naprijed za slučajni broj koraka
4. Ako je 1, okreće se za slučajni kut
5. Svaki 10 poteza mijenja boju
6. Igra završava nakon 50 poteza

37.2 Korak 1: Procedura za jedan potez

```
to potez
  ifelse (random 2) = 0 [
    fd 10 + random 30
  ] [
    rt (random 90) - 45
  ]
end
```

random 2 daje 0 ili 1:

- 0: Tina ide naprijed za **10 do 39 koraka** (random 30 daje od 0 do 29, plus 10)
- 1: Tina se okreće za slučajni kut između **-45 i 44 stupnja**. Negativan kut znači

okret ulijevo.

37.3 Korak 2: Procedura za promjenu boje

Imamo nekoliko boja koje ćemo nasumično birati:

```
to slucajna_boja
  make "izbor random 5
  if :izbor = 0 [ setpencolor "red ]
  if :izbor = 1 [ setpencolor "blue ]
  if :izbor = 2 [ setpencolor "green ]
  if :izbor = 3 [ setpencolor "purple ]
  if :izbor = 4 [ setpencolor "orange ]
end
```

random 5 daje broj od 0 do 4. Svaki broj daje drugu boju.

Isto s popisom

Sjećaš se poglavlja **Popis u kutijici**? Isto možeš napisati i u jednom retku: `setpencolor pick [red blue green purple orange]`. Ovdje smo namjerno uzeli `if`, jer je ovo dio o uvjetima.

37.4 Korak 3: Glavna petlja

Sad spajamo sve: 50 poteza, a svakih 10 poteza nova boja.

```
to igra
  cs
  setpensize 3
  make "i 0
  repeat 50 [
    if (remainder :i 10) = 0 [
      slucajna_boja
    ]
    potez
    make "i :i + 1
  ]
end
```


Što radi:

1. cs: obriše platno i vrati Tinu u sredinu. Kad pritisneš , platno je ionako čisto, ali ako igra pozoveš još jednom, cs prvo obriše staru šetnju.
2. setpensize 3: srednja debljina olovke
3. make "i 0: brojač poteza
4. Petlja 50 puta:
 - Ako je :i djeljiv s 10 (0, 10, 20, 30, 40), promijeni boju
 - Napravi jedan potez
 - Povećaj brojač

37.5 Cijeli kod

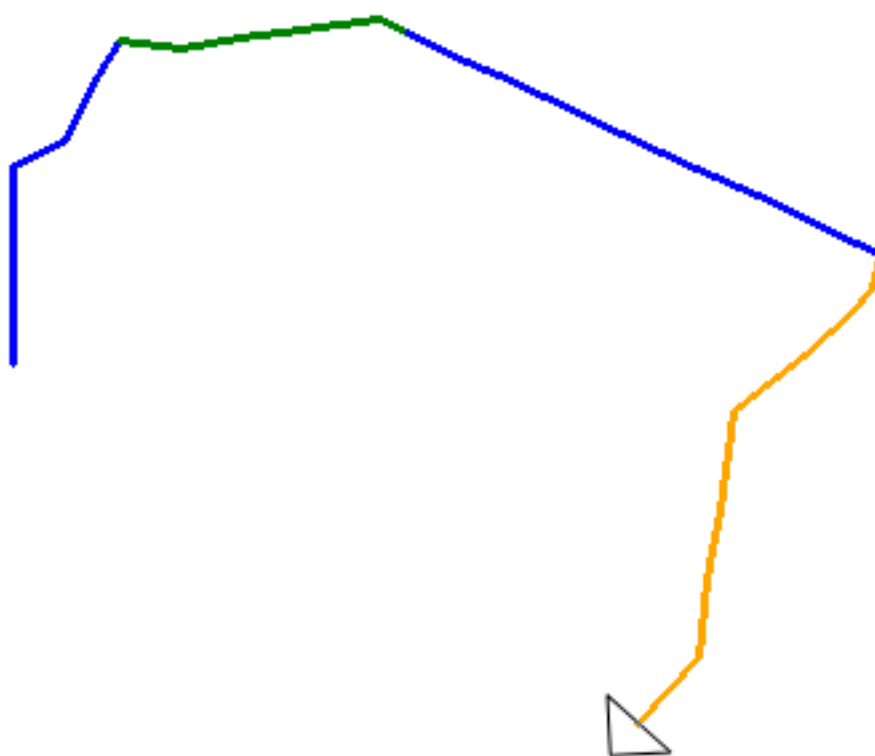
Sve tri procedure idu jedna ispod druge, a poziv igra na dno:

```
to potez
  ifelse (random 2) = 0 [
    fd 10 + random 30
  ] [
    rt (random 90) - 45
  ]
end

to slucajna_boja
  make "izbor random 5
  if :izbor = 0 [ setpencolor "red ]
  if :izbor = 1 [ setpencolor "blue ]
  if :izbor = 2 [ setpencolor "green ]
  if :izbor = 3 [ setpencolor "purple ]
  if :izbor = 4 [ setpencolor "orange ]
end

to igra
  cs
  setpensize 3
  make "i 0
  repeat 50 [
    if (remainder :i 10) = 0 [
```

```
        slucajna_boja  
    ]  
    potez  
    make "i :i + 1  
]  
end  
  
igra
```



Slika 37.1: Jedan mogući rezultat igre, svaki put je drugačiji

37.6 Pokreni više puta

Pritisni  **Pokreni** više puta. Svaki put dobiješ **drugačiji** crtež, jer random svaki put daje druge brojeve.

Možeš i ovako: nakon pokretanja u **redak za naredbe** ispod konzole upiši **igra** i pritisni Enter. Tina odmah odigra novu šetnju, a cs na početku procedure prvo obriše staru.

37.7 Eksperimentiraj

Probaj mijenjati brojeve:

- Više poteza: repeat 100 [...] umjesto 50
- Veći koraci: fd 30 + random 50 umjesto fd 10 + random 30
- Drugi kutovi: rt (random 180) - 90 (Tina se može jače okrenuti ulijevo ili udesno)
- Češća promjena boje: if (remainder :i 5) = 0 [...]

37.8 Što je sve u igri?

Pogledajmo koje smo alate iskoristili:

- ☒ **Variable** (make "i 0, make "izbor)
- ☒ **Procedure** (potez, slucajna_boja, igra)
- ☒ **Petlje** (repeat 50 [...])
- ☒ **Uvjeti** (if, ifelse)
- ☒ **Slučajni brojevi** (random)
- ☒ **Računanje** (remainder :i 10, :i + 1)
- ☒ **Boje** (setpencolor)
- ☒ **Olovka** (setpensize)
- ☒ **Brisanje platna** (cs)

To je **gotovo sve** što smo naučili! Mali program, a koristi gotovo sve programerske alate.

37.9 Vježba

Napravi vlastitu varijantu igre. Na primjer:

- "Spirala koja mijenja boju": Tina crta spiralu, a kad crta postane dulja od 100, mijenja boju
- "Kocka bira oblik": u svakom potezu Tina slučajno bira hoće li nacrtati kvadrat, trokut ili krug
- "Duga šetnja": Tina nakon svakog poteza nasumično mijenja i debljinu olovke

Sjedeće: [Projekti](#) →

IX. dio

Projekti

Poglavlje 38

Projekti

Sad kad znaš sve osnove (kretanje, oblike, ponavljanje, varijable, popise, procedure i uvjete), vrijeme je za **projekte**.

Projekt nije samo “primjer u tekstu”. Projekt je **cijela slika koju sam zamisliš** i gradiš je dio po dio, kao projekte iz prve knjige.

U ovom dijelu su tri projekta, od jednostavnijeg prema složenijem:

1. **Mnogokutni cvijet**: jedan cvijet u bojama duge
2. **Stablo**: jednostavan fraktal, crtež koji ponavlja sam sebe
3. **Spirala duga**: šarena spirala koja mijenja boju

U cvijetu i spirali boje ćeš uzimati s **popisa boja**, baš kao u poglavlju [Popis u kutijici](#).

Cilj nije samo “prepiši kod”. Cilj je **razumjeti** kako svaki dio funkcionira i što se događa kada nešto promijeniš.



Sljedeće: [Projekt: Mnogokutni cvijet](#) →

Poglavlje 39

Projekt: Mnogokutni cvijet

Naš prvi projekt: cvijet napravljen od **dvanaest mnogokuta** raspoređenih u krug, svaki u **drugoj boji** duge.

39.1 Kako će izgledati

Zamisli sat. Na svakom broju sata (12 mjesta) stoji jedan **mnogokut** (u našem cvijetu šesterokut) i svi zajedno tvore latice cvijeta. Svi mnogokuti kreću iz iste točke u sredini cvijeta.

39.2 Plan

1. Napišemo proceduru koja crta jednu **laticu** (mnogokut)
2. Napravimo **popis boja** iz kojeg svaka latica uzme svoju boju
3. Glavna petlja crta 12 latica, svaku u drugoj boji, oko cijelog kruga

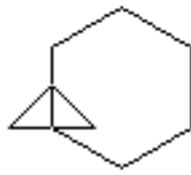
39.3 Korak 1: Latica

Latica je mnogokut sa zadanim brojem stranica i veličinom. Ispod definicije (iza retka `end`) dopiši poziv latica 6 30:

```
to latica :stranice :duljina
  repeat :stranice [
    fd :duljina
    rt 360 / :stranice
  ]
end

latica 6 30
```

Šesterokut sa stranicom 30. To je jedna latica.



Slika 39.1: Jedna latica: šesterokut

39.4 Korak 2: Popis boja

Imamo 12 latica i želimo 12 boja duge. Sjećaš se popisa iz prve knjige? U poglavlju [Popis u kutijici](#) napisao si ga i kodom. Svih 12 boja stavimo u jednu kutijicu boje:

```
make "boje [red orange yellow green cyan blue
           purple magenta pink red orange yellow]
print count :boje
print item 1 :boje
print item 12 :boje
```

U konzoli ćeš vidjeti 12, red i yellow: na popisu je 12 boja, prva je crvena, a dvanaesta žuta. Neke boje se ponavljaju, jer duga nema baš 12 različitih boja. Popis smije ići i u dva retka: Logo ga čita sve do zagrade] koja ga zatvara.

39.5 Korak 3: Cijeli cvijet

```
to cvijet
  setpensize 2
  make "boje [red orange yellow green cyan blue
              purple magenta pink red orange yellow]
  repeat 12 [
    setpencolor item repcount :boje
    latica 6 30
    rt 30
  ]
end
```

Što radi:

1. setpensize 2: malo deblja olovka
2. make "boje [...]: popis od 12 boja
3. Petlja 12 puta:
 - repcount je redni broj ponavljanja (1, 2, 3... 12), pa item repcount :boje u prvom prolazu uzme prvu boju, u drugom drugu, i tako dalje
 - Nacrtaj jednu laticu (šesterokut)
 - Okreni se za 30 ($12 \times 30 = 360$, pun krug)

39.6 Pokreni

Sve spojimo u jedan program: prvo obje definicije, a ispod njih (iza zadnjeg retka end) poziv cvijet:

```
to latica :stranice :duljina
  repeat :stranice [
    fd :duljina
    rt 360 / :stranice
  ]
end

to cvijet
  setpensize 2
  make "boje [red orange yellow green cyan blue
             purple magenta pink red orange yellow]
  repeat 12 [
    setpencolor item repcount :boje
    latica 6 30
    rt 30
  ]
end

cvijet
```

Trebao bi vidjeti **dugin cvijet** s 12 šesterokutnih latica.



Slika 39.2: Šareni cvijet od 12 šesterokuta u bojama duge

39.7 Vježbe za eksperimentiranje

1. **Drugi mnogokut:** promijeni latica 6 30 u latica 5 30 i dobit ćeš peterokutne latice. Ili latica 8 30 za osmerokutne. Što ti se najviše sviđa?
2. **Veći ili manji cvijet:** promijeni duljinu stranice latice u 50 ili 20.
3. **Više latica:** promijeni repeat 12 [u repeat 24 [i rt 30 u rt 15. Cvijet ima dvostruko više latica. Na popisu je samo 12 boja, pa neka Tina za svaku laticu boju bira nasumično: umjesto item repcount :boje napiši pick :boje.
4. **Drugačije boje:** promijeni boje na popisu tako da budu samo nijanse plave: cyan, blue, purple... Pazi da na popisu i dalje bude 12 boja (smiju se ponavljati).
5. **Cvijet sa stabljikom:** iza poziva cvijet dopiši setpencolor "green i bk 150. Tina nakon cvijeta stoji u sredini i gleda gore, pa bk 150 nacrtá stabljiku prema dolje.

Sljedeće: [Projekt: Stablo \(fraktal\)](#) →

Poglavlje 40

Projekt: Stablo (fraktal)

Fraktal je crtež koji **ponavlja sam sebe**: mali dio izgleda kao cijela slika. U prirodi su stabla pravi primjer: glavna grana se dijeli u dvije, svaka od njih se opet dijeli u dvije, pa opet...

U ovom projektu ćemo napraviti **stablo** koje se grana sve do određene dubine.

40.1 Kako stablo nastaje

Stablo crtamo ovako:

1. Nacrtaj **deblo** prema gore
2. Na vrhu, **podijeli se na dvije grane** (lijevo i desno)
3. Svaka grana je **manja verzija stabla**
4. Ponavljaj dok grane ne postanu prekratke

Tako stablo ima **mnogo grana** koje se sve više smanjuju.

40.2 Procedura: stablo

```
to stablo :duljina
  if :duljina < 5 [ stop ]

  fd :duljina

  lt 30
  stablo :duljina * 0.7
  rt 60
  stablo :duljina * 0.7
  lt 30

  bk :duljina
end
```

Ovo je **rekurzivna** procedura, jer poziva **samu sebe** (stablo unutar stablo).

40.3 Što radi korak po korak

Pretpostavi stablo 100:

1. **Provjera:** je li :duljina < 5? **Ne** (100 je veće), idemo dalje.
2. **fd 100:** nacrtaj deblo.
3. **lt 30:** okreni se ulijevo za 30 stupnjeva.
4. **stablo 70:** pozovi novi stablo s manjom duljinom ($100 * 0.7 = 70$).
5. Unutar tog: nacrtaj manje deblo, pa opet dvije grane od 49 ($70 * 0.7$), i tako dalje.
6. Kad se vrati, **rt 60:** okreni se za 60 udesno (sad gleda 30 stupnjeva desno od početnog smjera).
7. **stablo 70:** druga grana.
8. **lt 30:** vrati se u početni smjer.
9. **bk 100:** vrati se na početak debla.

40.4 Naredba stop

Redak `if :duljina < 5 [ stop ]` je **uvjet za zaustavljanje**. Bez njega bi se procedura pozivala **beskonačno**, svaki put s manjom duljinom (5, 3.5, 2.45, 1.71...), sve dok Logu ne ponestane memorije. Program se tada zaustavi s greškom ili se zaglavi. Ako se dugo ništa ne događa, osvježi stranicu u pregledniku.

`stop` je naredba koja **prekida** trenutnu proceduru i vraća se u onu koja ju je pozvala.

40.5 Pokreni

```
to stablo :duljina
  if :duljina < 5 [ stop ]

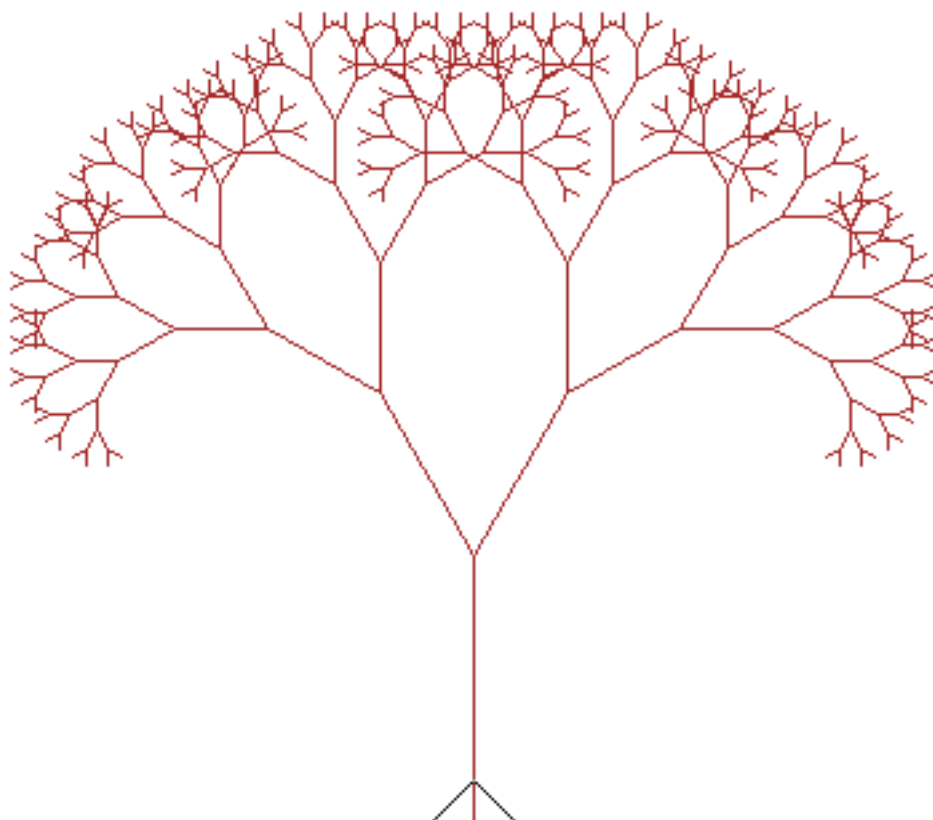
  fd :duljina

  lt 30
  stablo :duljina * 0.7
  rt 60
  stablo :duljina * 0.7
  lt 30

  bk :duljina
```

```
end

setpencolor "brown
pu bk 200 pd
stablo 100
```



Slika 40.1: Fraktalno stablo

Naredbe prije stablo 100:

- setpencolor "brown: smeđa boja (kao deblo)
- pu bk 200 pd: pomakni Tinu dolje (da bi stablo "raslo" prema gore)

40.6 Eksperimenti

40.6.1 Drugačiji kut grananja

Probaj lt 45 i rt 90 umjesto lt 30 i rt 60. Zadnji lt 30 promijeni u lt 45, da se Tina vrati u početni smjer:

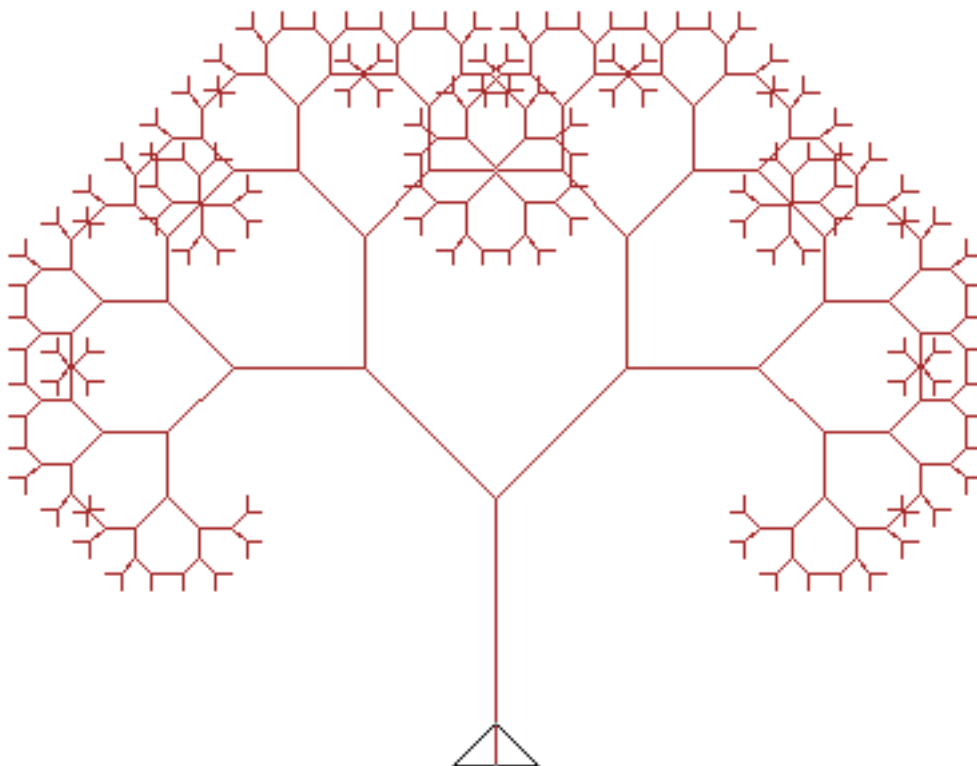
```

to stablo :duljina
  if :duljina < 5 [ stop ]
  fd :duljina
  lt 45
  stablo :duljina * 0.7
  rt 90
  stablo :duljina * 0.7
  lt 45
  bk :duljina
end

setpencolor "brown
pu bk 200 pd
stablo 100

```

Šire raširene grane.



Slika 40.2: Stablo sa šire raširenim granama

40.6.2 Drugi broj smanjivanja

Mijenjaj 0.7 (na oba mjesta):

- 0.5: puno manje grane (kratko stablo)
- 0.8: malo veće grane (gušće stablo)
- 0.9: stablo postane ogromno i Tina mora nacrtati više od pola milijarde grana, pa program praktički nikad ne završi. Zaustavi ga gumbom  pokraj gumba . Drži se brojeva od 0.5 do 0.8.

40.6.3 Boje koje se mijenjaju ovisno o dubini

```
to stablo :duljina
  if :duljina < 5 [ stop ]

  ifelse :duljina > 30 [
    setpencolor "brown
  ] [
    setpencolor "green
  ]

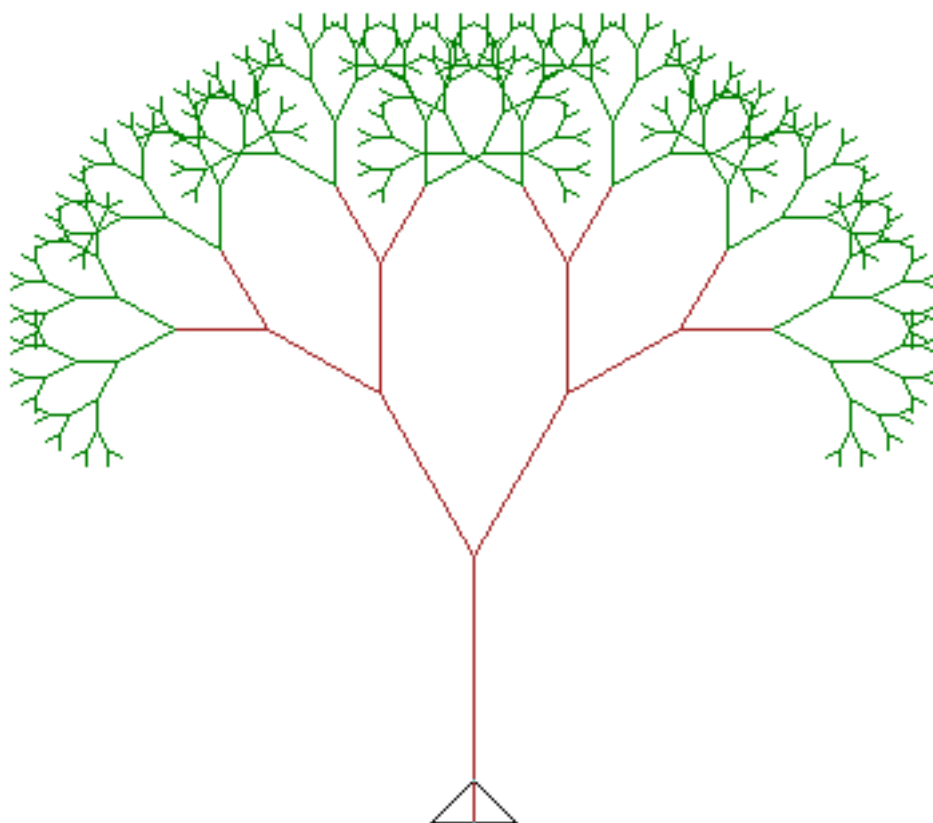
  fd :duljina

  lt 30
  stablo :duljina * 0.7
  rt 60
  stablo :duljina * 0.7
  lt 30

  pu bk :duljina pd
end

pu bk 200 pd
stablo 100
```

Dulje grane (dulje od 30) crtaju se smeđom bojom, a kratke zelenom, **kao lišće!** Pazi na zadnji redak procedure: Tina se vraća s podignutom olovkom. Da se vraća s olovkom dolje, prešla bi zelenom bojom preko smeđih grana i cijelo stablo bi bilo zeleno.



Slika 40.3: Stablo sa smeđim deblom i granama i zelenim lišćem

40.7 Što si naučio

Ovaj projekt te uveo u **rekurziju**, jedan od najmoćnijih programerskih alata. Procedura koja poziva samu sebe može napraviti vrlo složene stvari s vrlo malo koda.

Probaj sam istražiti druge fraktale:

- **Pahuljica** (Kochova krivulja)
- **Sierpińskijev trokut** (trokut iz trokuta iz trokuta...)
- **Spirala s rekurzijom** umjesto petlje

Kochovu pahuljicu i Sierpińskijev trokut crtati ćeš i u trećoj knjizi, **Logo: Naprijed**. Mnoge fraktale naći ćeš i na internetu pod imenom “Logo fraktali”, ali probaj sam izmisliti svoje!

Sljedeće: [Projekt: Spirala duga](#) →

Poglavlje 41

Projekt: Spirala duga

Naš zadnji projekt je **šarena spirala**: Tina crta krug koji se postupno **širi**, a boja se mijenja kako spirala raste.

41.1 Kako spirala nastaje

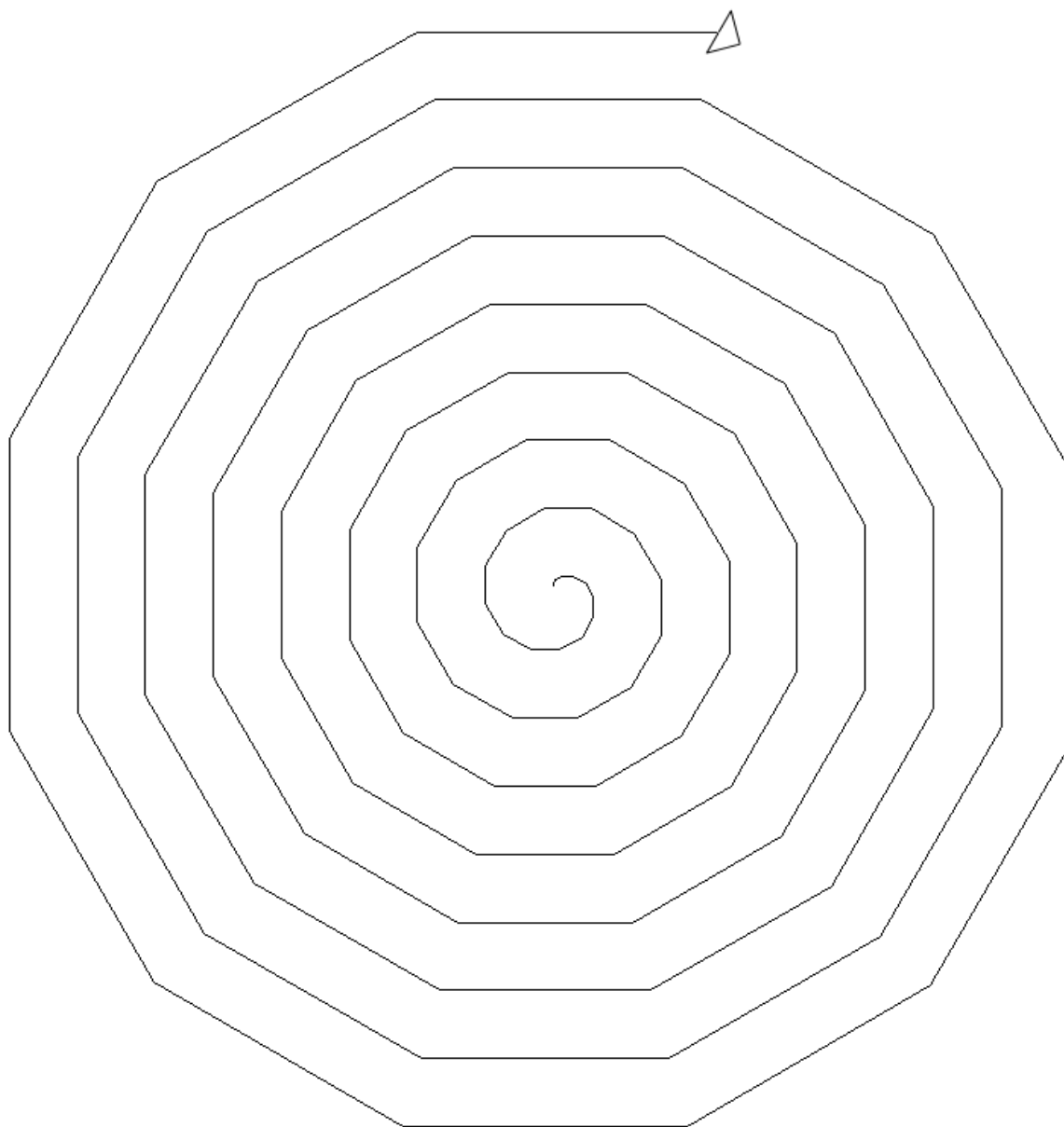
Razlika između kruga i spirale:

- **Krug**: Tina svaki put ide istu udaljenost
- **Spirala**: Tina svaki put ide **malo dalje** od prošlog puta

Ako je svaka crta malo dulja od prošle (u našem primjeru za 2), dobijemo **spiralu koja se širi**.

41.2 Najjednostavnija spirala

```
make "duljina 1
repeat 100 [
  fd :duljina
  rt 30
  make "duljina :duljina + 2
]
```

Slika 41.1: Spirala koja se širi

Što se događa:

1. Postavimo :duljina na 1
2. 100 puta:
 - Idi naprijed za :duljina (u prvom prolazu 1, pa 3, pa 5...)
 - Okreni se desno za 30
 - Povećaj :duljina za 2

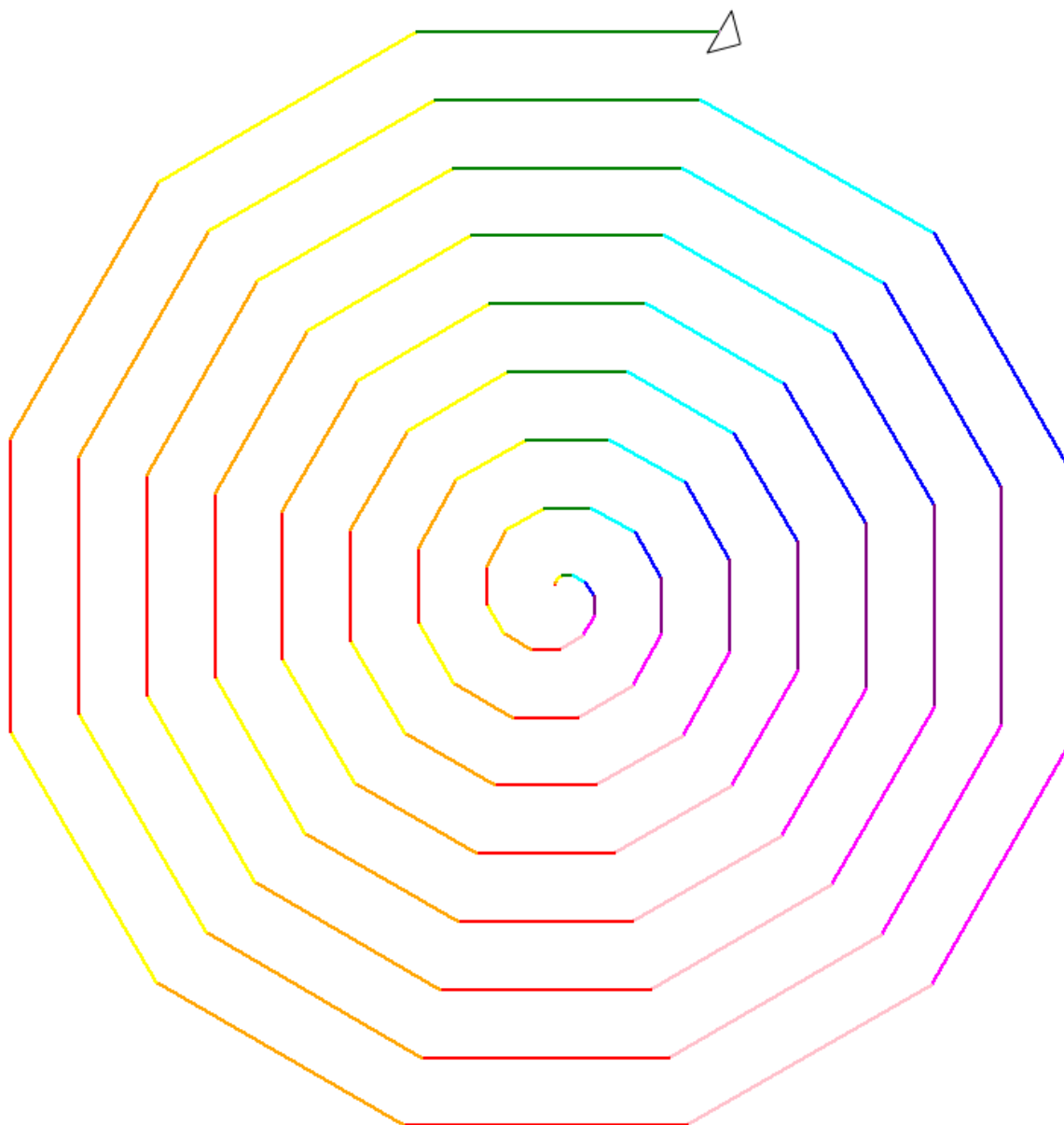
Tina se okreće za 30, pa jedan krug napravi od 12 crta. Ali svaki sljedeći krug je veći jer su crte sve dulje. To čini **spiralu**.

41.3 Spirala u boji

Sada dodajmo boje. Uzet ćemo isti popis od 12 boja kao u projektu [Mnogokutni cvijet](#). Spirala ima 100 koraka, a na popisu je samo 12 boja, pa boje moraju ići u krug: nakon dvanaeste opet dolazi prva.

```
to spirala
  setpensize 2
  make "boje [red orange yellow green cyan blue
              purple magenta pink red orange yellow]
  make "duljina 1
  make "i 0
  repeat 100 [
    setpencolor item (1 + remainder :i 12) :boje
    fd :duljina
    rt 30
    make "duljina :duljina + 2
    make "i :i + 1
  ]
end

spirala
```



Slika 41.2: Šarena spirala duga

Bitan dio je **1 + remainder :i 12**. Brojač :i raste 0, 1, 2, 3..., a remainder :i 12 uvijek daje broj od 0 do 11. Mjesta na popisu idu od 1 do 12, zato dodajemo 1. Tako boje kruže kroz popis, i to točno jednom u svakom krugu spirale. Zato svaki smjer na slici ima svoju boju.

41.4 Eksperimenti

41.4.1 Drugi kut

Promijeni rt 30 u nešto drugo. Magični kutovi:

- rt 30: obična spirala (12 stranica po krugu)
- rt 89: **kvadratna** spirala koja se zakreće (jer 89 nije baš 90)
- rt 91: drugačija kvadratna spirala
- rt 144: **zvjezdana** spirala
- rt 137: “zlatna” spirala (blizu kuta po kojem su raspoređene sjemenke u sunco-kretu)

Probaj svaki!

41.4.2 Brže ili sporije širenje

Promijeni make "duljina :duljina + 2:

- + 1: crte rastu sporo, pa su zavoji gusto jedan uz drugi
- + 5: crte rastu brzo, pa se spirala jako raširi, a zavoji su daleko jedan od drugoga
- + 0.5: crte rastu vrlo sporo, pa nastaje mala i vrlo gusta spirala

Broj zavoja ostaje isti, jer Tina i dalje napravi 100 okreta od 30 stupnjeva. Mijenja se samo koliko je spirala velika i koliko su zavoji razmaknuti.

41.4.3 Promjena debljine olovke

Ovaj put pišemo sve bez procedure, s jednim novim retkom setpenseize:

```
make "boje [red orange yellow green cyan blue
            purple magenta pink red orange yellow]
make "duljina 1
make "i 0
repeat 100 [
    setpencolor item (1 + remainder :i 12) :boje
    setpenseize round 1 + :i / 20
    fd :duljina
    rt 30
    make "duljina :duljina + 2
    make "i :i + 1
]
```

Olovka **postaje sve deblja** kako spirala raste. (Naredba round zaokružuje broj, jer setpenseize želi cijele brojeve.)



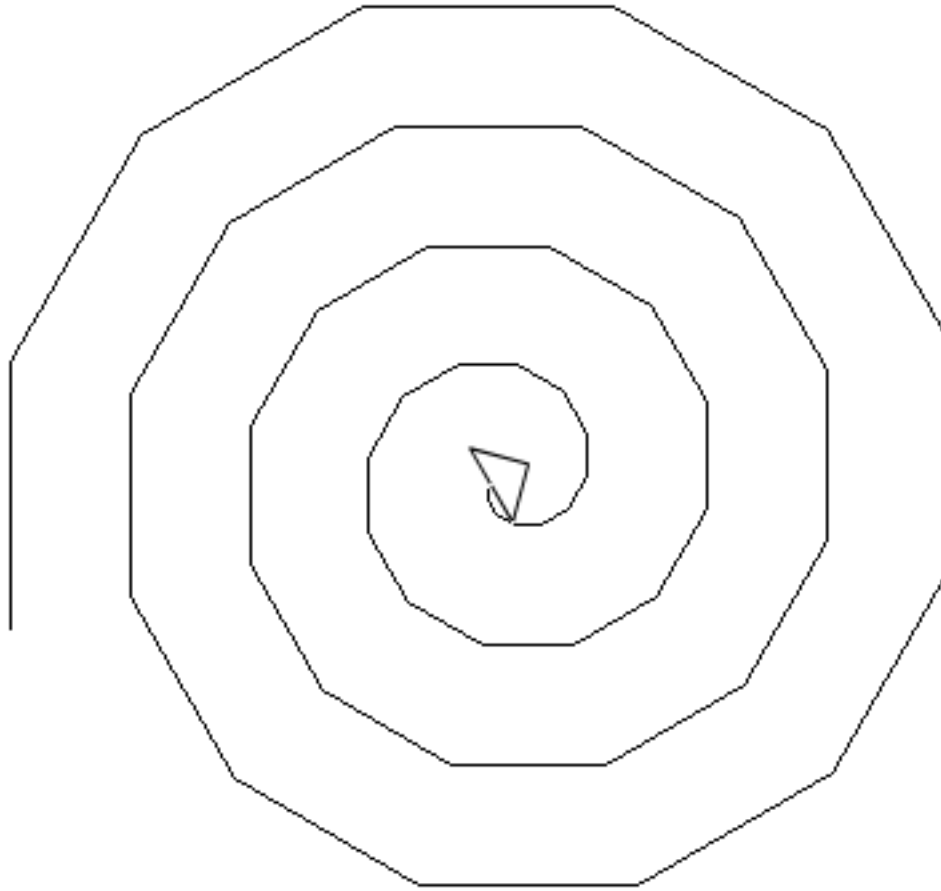
Slika 41.3: Spirala sa sve debljom olovkom

41.4.4 Spirala izvana prema unutra

Što ako Tina **skraćuje** crtu umjesto da je produljuje?

```
make "duljina 100
repeat 50 [
  fd :duljina
  rt 30
  make "duljina :duljina - 2
]
```

Izvana velika, prema unutra sve manja: **spiralala koja se uvija prema sredini**.



Slika 41.4: Spirala koja se uvija prema sredini

41.5 Završna riječ

Tri projekta (cvijet, stablo i spirala) pokazala su tri različite tehnike:

- **Cvijet:** ponavljanje s okretanjem
- **Stablo:** rekurzija (procedura poziva samu sebe)
- **Spirala:** postupno mijenjanje varijable

A u cvijetu i spirali pomogao ti je i **popis boja**. Sve te tehnike koristit ćeš u **bezbroj** drugih programa, i to ne samo u Logu, nego u svim programskim jezicima.

Sljedeće: [Što dalje? →](#)

Poglavlje 42

Što dalje?

Čestitamo! Stigao si do kraja **druge knjige** o programiranju u Logu.

Sjeti se gdje si počeo: slagao si blokove, punio kutijice i učio Tinu nove riječi. Sad sve to pišeš kodom, i još više:

- Vlastite **procedure s parametrima**
- **Petlje koje se ponavljaju** stotinama puta
- **Popise** iz kojih Tina uzima boje
- **Uvjete koji odlučuju** što Tina radi
- **Rekurzivne fraktale** poput stabla
- **Šarene spirale i cvjetove**



To je **pravo programiranje**.

42.1 Što sad?

42.1.1 Treća Logo knjiga

Treća knjiga, **Logo: Naprijed**, već te čeka. U njoj ideš dublje:

- **Uvjeti dublje**: složeniji uvjeti s and, or i not, uvjet unutar uvjeta i petlje koje traju dok je nešto istina
- **Liste**: popise iz prve knjige i iz poglavlja [Popis u kutijici](#) mijenjat ćeš, spajati i slagati u liste lista
- **Funkcije**: procedure koje **vraćaju vrijednost** (naredba output) i predikati, funkcije koje odgovaraju s true ili false
- **Rekurzija**: rekurzija s brojevima i listama, a **L-sustavi** crtaju biljke i Sierpinski trokut
- **Kochova pahuljica**: fraktal koji izgleda kao snježna pahulja
- **Animacija**: crteži koji se mijenjaju kroz vrijeme, kao mali film

42.1.2 Python

Logo je super za **učenje** programiranja, ali profesionalci koriste druge jezike. **Python** je jedan od najpopularnijih.

Studio podržava i **Python**! Kad stvaraš novi kod, u polju **Vrsta koda** odaberi **Python Turtle Code**. I tamo postoji kornjača (na engleskom *turtle*), vrlo slična našoj Tini!

Python program s kornjačom počinje retkom `import turtle`, a svaka naredba kornjači počinje s `turtle..` Evo kvadrata:

```
import turtle

for i in range(4):
    turtle.forward(100)
    turtle.right(90)

turtle.done()
```

U ovom Python programu nema uglatih zagrada oko naredbi koje se ponavljaju. Ono što je uvučeno ispod retka `for` ponavlja se, baš kao ono između `[ i ]` u Logu. Zadnji redak `turtle.done()` kaže da je crtež gotov.

Logo i Python turtle su vrlo slični:

Logo	Python
<code>fd 100</code>	<code>turtle.forward(100)</code>
<code>rt 90</code>	<code>turtle.right(90)</code>
<code>repeat 4 [...]</code>	<code>for i in range(4):</code>
<code>make "x 5</code>	<code>x = 5</code>
<code>to kvadrat ... end</code>	<code>def kvadrat():</code>

Ako znaš Logo, **već znaš pola Pythona**.

42.1.3 Vlastiti projekti

Najbolji način za napredak je **napraviti vlastite stvari**:

- Nacrtaj **svoju kuću** (s prozorima, vratima, krovom)
- Napravi **šahovsku ploču** (8x8 crnih i bijelih kvadrata)
- Crtaj **mandale** (cvjetne uzorke s mnogo simetrije)
- Nacrtaj **zimsku noć** (bijeke pahulje od malih zvjezdica na tamnom platnu)
- Stvori **vlastito pismo** (procedure koje crtaju slova)

42.1.4 Dijeljenje s drugima

Pokaži svoje crteže prijateljima i obitelji. Snimi sliku zaslona ili kratak video. Bit će impresionirani što znaš programirati!

42.2 Najvažnija lekcija

Programiranje **nije** pamćenje naredbi, nego **rješavanje problema**:

1. **Rastavi** veliki problem na **manje dijelove**
2. **Riješi** svaki manji dio (procedurom)
3. **Sastavi** male dijelove u veliku sliku

To je **vještina za cijeli život**. Bilo da postaneš programer, inženjer, znanstvenik, dizajner ili nešto sasvim drugo, **logičko razmišljanje** uvijek će ti koristiti.

42.3 Hvala!

Hvala što si bio sa mnom kroz ovu knjigu. Nadam se da ti je bila zabavna i da si naučio puno.

Sad idi i **napravi nešto čarobno** s Tinom!

“Računalo ne radi ono što želiš. Ono radi ono što napišeš.”

- stari programerski zakon

Uživaj u programiranju!

Sljedeće: [Podsjetnik →](#)

Podsjetnik

Sve naredbe iz ove knjige na jednom mjestu. Ako nešto zaboraviš, otvori stranicu na kojoj piše više!

Naredba	Kratko	Što radi	Str.
KRETANJE			
Pokreni 	F5	pokreće tvoj program	10
forward 100	fd	Tina hoda naprijed	21
back 100	bk	Tina hoda natrag , a gleda i dalje naprijed	22
right 90	rt	Tina se okreće desno	26
left 90	lt	Tina se okreće lijevo	27
home		Tina se vraća u sredinu i gleda gore	24
; komentar		tekst koji Logo preskače, a tebi pomaže	163
Skraćenice		tablica svih kratkih imena naredbi	30
OBLICI			
Kvadrat		4 puta fd 100 i rt 90	45
Trokut		3 puta fd 100 i rt 120 (vanjski kut)	52
Čarobni broj 360		okreti pravilnog mnogokuta zajedno daju 360	47
PONAVLJANJE			
repeat 4 []		ponavlja naredbe u zagradama	65
repeatcount		redni broj prolaza kroz repeat (1, 2, 3...)	147
Mnogokut		repeat N [fd 50 rt 360 / N]	69
Krug od koraka		repeat 36 [fd 10 rt 10], ili circle 50 i circle2 50 iz prve knjige	70
Pravilo zvijezde		repeat N [fd 100 rt 360 * 2 / N]	82
repeat u repeatu		cijeli oblik se ponavlja, pa nastaje uzorak	90
BOJE I OLOVKA			
penup / pendown	pu / pd	olovka gore (ne crta) / dolje (opet crta)	100
hideturtle / showturtle	ht / st	sakrij / pokaži Tinu	104
clearscreen	cs	obriši platno usred programa	105
setpencolor "red	setpc	boja olovke riječju	106
setpencolor [255 0 0]		boja kao tri broja (RGB)	109
setscreencolor "black	setsc	boja platna , briše crtež, zato ide prva	110
setpensize 5		debljina crte	115
VARIABLE			

Naredba	Kratko	Što radi	Str.
make "strana 75		stavi broj u kutijicu (blok set strana to)	124
:strana		uzmi broj iz kutijice (okrugli blok strana)	125
make "strana :strana + 10		kutijica naraste (blok change strana by)	129
RAČUNANJE			
+ - * /		zbroji, oduzmi, pomnoži, podijeli	134
()		zgrade kažu što se računa prvo	137
random 100		slučajan broj od 0 do 99	137
POPISI			
make "boje [red blue green]		popis u kutijici	142
count :boje		koliko je stvari na popisu	143
first / last / item 2 :boje		prva / zadnja / druga stvar s popisa	143
pick :boje		nasumična stvar s popisa	144
foreach :boje [[boja] ...]		napravi nešto za svaku stvar s popisa	145
PROCEDURE			
to kvadrat ... end		nauči Tinu novu riječ	153
kvadrat		pozovi proceduru, uvijek ispod njezine definicije	154
to kvadrat :strana		procedura s parametrom	165
to mnogokut :stranice		procedura s dva parametra	168
:duljina			
stop		prekida proceduru	205
UVJETI			
if :x > 30 []		naredbe samo ako je istina	179
ifelse ... [] []		ako ovo, inače ono	181
= <> < > <= >=		usporedi dva broja	186
print remainder 10 2		ostatak dijeljenja (paran ili neparan broj)	187
and / or		oba uvjeta / barem jedan uvjet	189
print [poruka]		ispiši poruku u konzolu	191