

# C programski jezik



Univerziteta u Novom Sadu  
Fakultet tehničkih nauka  
Departman za računarstvo i automatiku  
Odsek za primenjene računarske nauke i informatiku  
Katedra za informatiku

**Dr Željko Marčičević**

# Rad u Code::Blocks okruženju



Da bi se napisao i izvršio program napisan na programskom jeziku C, potreban je:

- ✓ Tekst editor u kojem će program biti napisan,
- ✓ Kompajler i Bilder koji će napisani kod kompajlirati i prevesti u mašinski kod, razumljiv računaru.

Postoji dosta aplikacija koji u sebi integrišu ove alate i one nose zajedničko ime IDE (integrated development environment). One u sebi sadrže pomenute alate:

- ✓ Kod editor,
- ✓ Kompajler ili Interpreter,
- ✓ Bilder i
- ✓ Debugger (alat za testiranje programa).

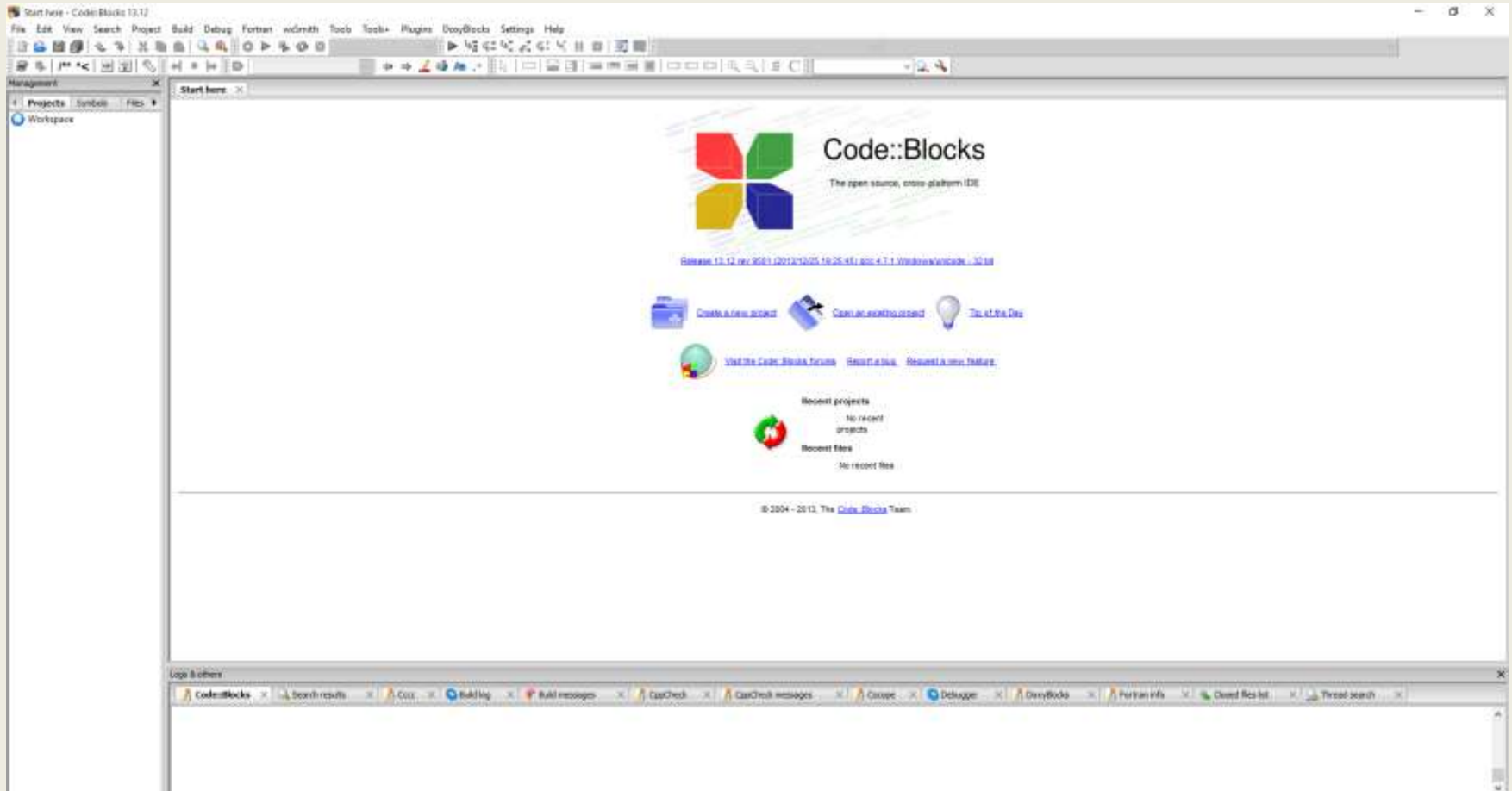
Jedan od takvih alata je Code::Blocks IDE, open source alat koji je besplatan i koji se može skinuti sa adrese <http://www.codeblocks.org>. Da bi se program mogao kompajlirati i pokrenuti, sa pomenute adrese je potrebno skinuti instalaciju codeblocks-xxxmingw-setup koja u sebi sadrži:

- ✓ GCC kompajler i
- ✓ GDB dibager.

# Rad u Code::Blocks okruženju



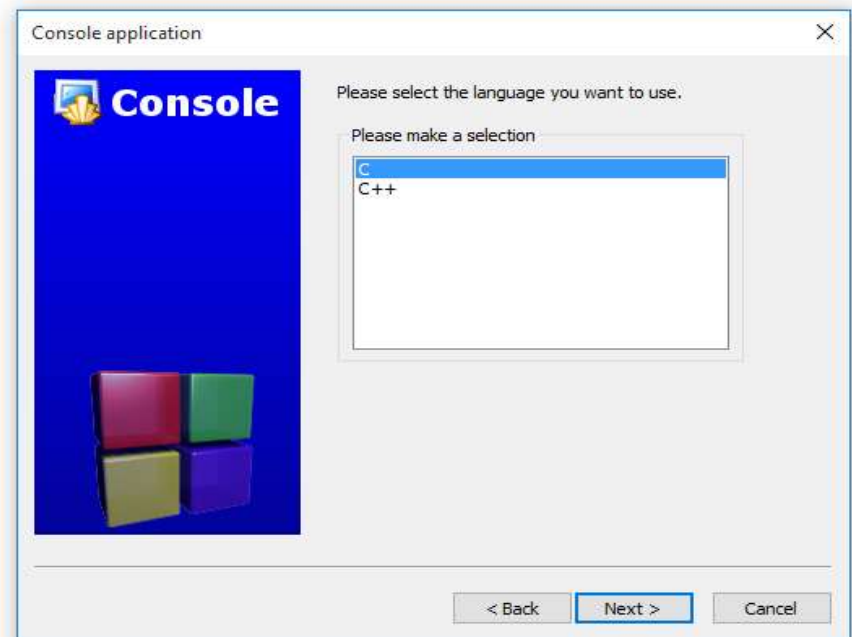
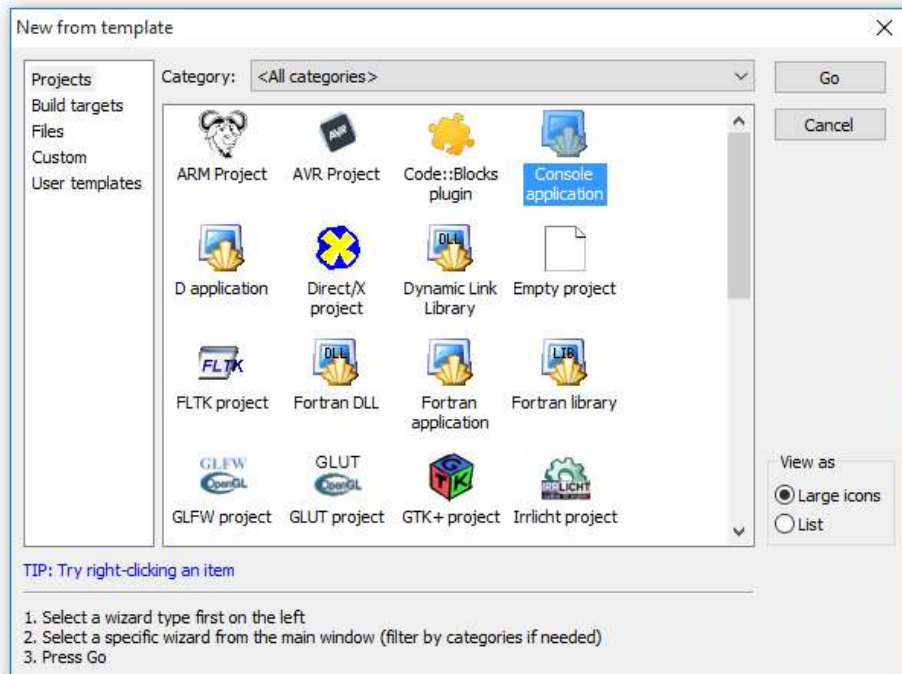
Nakon uspješne instalacije i pokretanja programa Code::Blocks dobija se sljedeći izgled ekrana:



# Rad u Code::Blocks okruženju

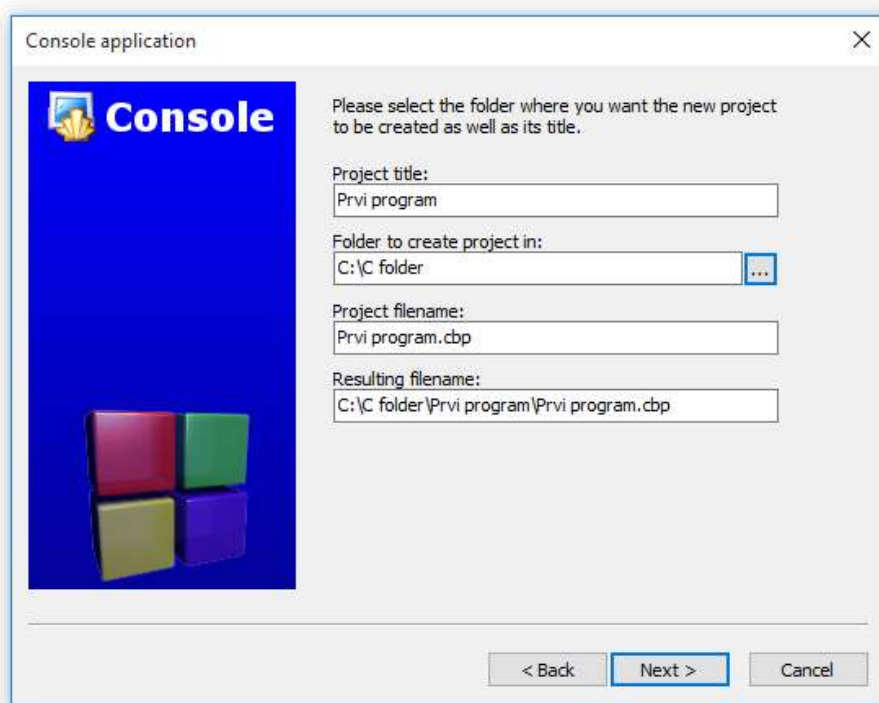
Da bi napisali i pokrenuli program u programskom jeziku C, potrebno je:

- Kreirati novi projekat (*File->New->Project*)
- Zatim izabrati *Console application*
- U polju za izbor jezika izabrati *C*



# Rad u Code::Blocks okruženju

- *Dati ime projektu kao i lokaciju na koju će se kod sačuvati,*
- *Zatim će se izvršiti kompajliranje putem GCC kompajlera.*



Console application

Please select the folder where you want the new project to be created as well as its title.

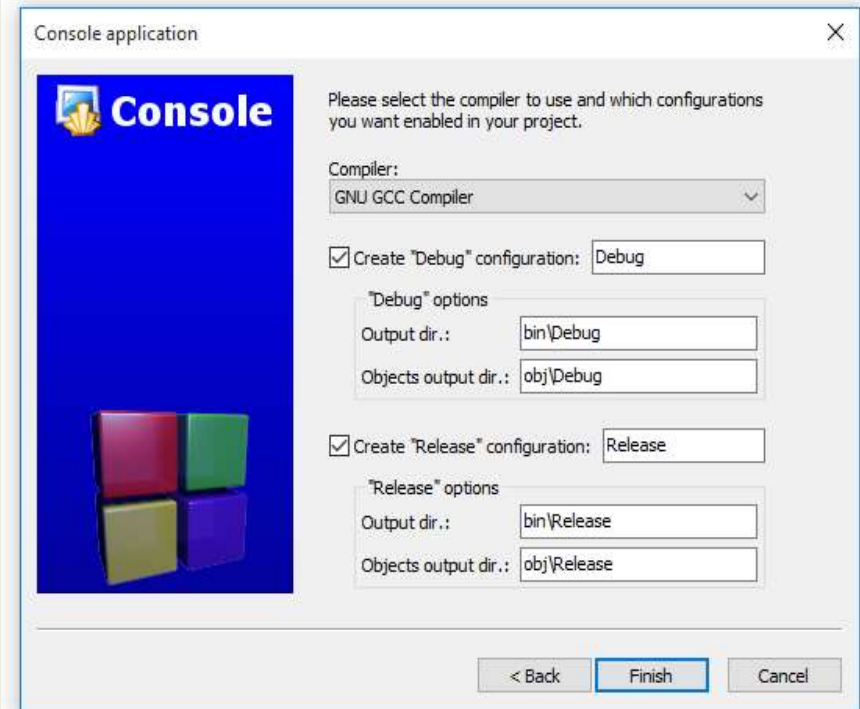
Project title:  
Prvi program

Folder to create project in:  
C:\C folder

Project filename:  
Prvi program.cbp

Resulting filename:  
C:\C folder\Prvi program\Prvi program.cbp

< Back Next > Cancel



Console application

Please select the compiler to use and which configurations you want enabled in your project.

Compiler:  
GNU GCC Compiler

☒ Create "Debug" configuration: Debug

"Debug" options  
Output dir.: bin\Debug  
Objects output dir.: obj\Debug

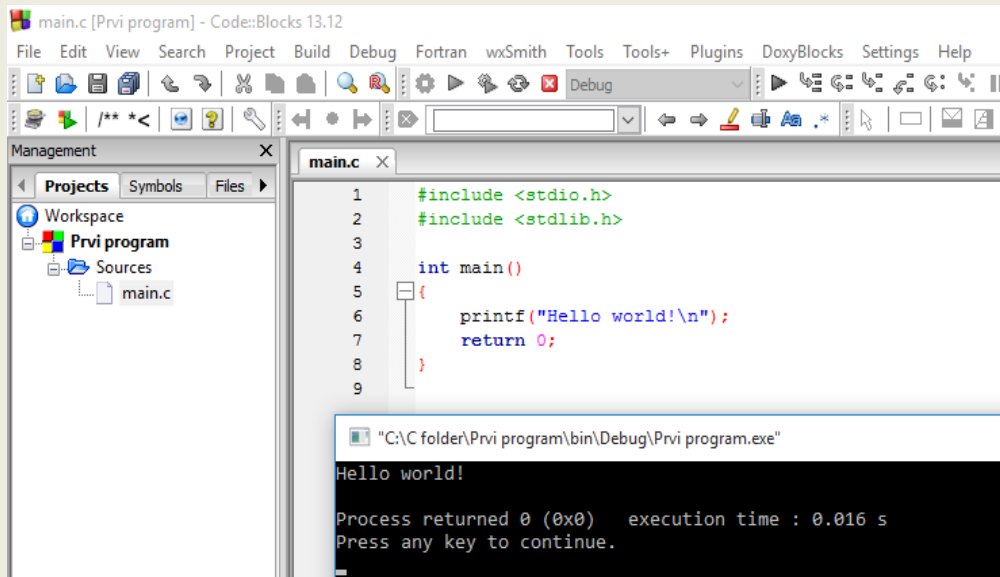
☒ Create "Release" configuration: Release

"Release" options  
Output dir.: bin\Release  
Objects output dir.: obj\Release

< Back Finish Cancel

# Rad u Code::Blocks okruženju

- Klikom na dugme *Finish*, u levom delu ekrana u delu *Projects* pojaviće se projekat koji je kreiran, kao i datoteka (*main.c*) u kojoj se nalazi glavna (*main*) funkcija kreiranog projekta.
- Otvaranjem te datoteke, dobija se kod najprostijeg C-programa, koji na ekran ispisuje tekst „Hello world“.
- Data je i struktura *Projects* na HDD u C folderu.



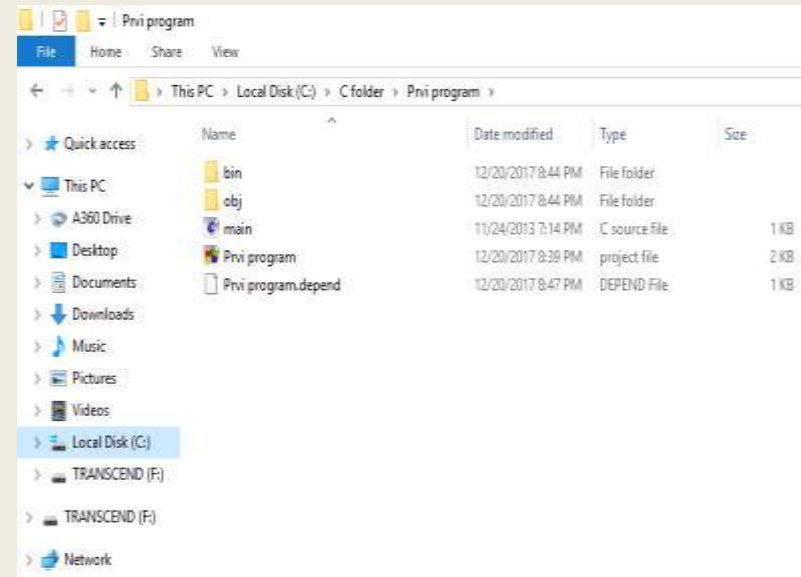
The screenshot shows the Code::Blocks IDE interface. The main editor window displays the following C code in `main.c`:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     printf("Hello world!\n");
7     return 0;
8 }
9
```

Below the editor, a terminal window shows the execution of the program:

```
"C:\C folder\Prvi program\bin\Debug\Prvi program.exe"
Hello world!
Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

The left sidebar shows the project structure under "Projects" and "Files" tabs, with "Prvi program" and "main.c" visible.



# Struktura programa



- C je tzv. **case-sensitive** programski jezik, što znači da **razlikuje mala i velika slova**.
  - ✓ Primer: Naredbe `return o` i `RETURN o` su dve različite naredbe.
- Nakon svake naredbe u C-u mora se nalaziti **karakter ;**
- C **ne poznaje razmake u kôdu**, tako da se prelazak u novi red nakon svake komande vrši samo iz estetskih razloga, radi lakšeg čitanja koda.
- **Komentari** se mogu pisati na dva načina:
  - ✓ Primer: ukoliko se komentar nalazi u jednom redu, dovoljno je na početak tog reda staviti karaktere `//`,
  - ✓ Primer: ukoliko se komentar nalazi u više redova, potrebno je komentar ograničiti parovima karaktera `/* i */`.

```
// Komentar u jednom redu
/* Komentar
   u više redova */
```

# Struktura programa



Prva dva reda programa u postojeći kôd uključuju datoteke za prevođenje i izvršavanje programa:

```
#include <stdio.h>
#include <stdlib.h>
```

- U datoteci **stdio.h** se nalaze funkcije za ulaz/izlaz, a
- U datoteci **stdlib.h** se nalazi većina funkcija koje su u najčešćoj upotrebi.

Funkcija:

```
int main()
```

- Je **glavna funkcija** programa i izvršavanje programa počinje od prve linije kôda ove funkcije.
- Svaki projekat **mora** u nekom od fajlova **imati ovu funkciju**.
- Može biti nekog drugog tipa, npr. **void** ili može **imati listu argumenata**.



# Struktura programa

Par vitičastih zagrada

```
{ }
```

- Predstavlja granice bloka naredbi, kao begin i end u programskom jeziku PASCAL

Poslednji red programa:

```
return 0;
```

- predstavlja izlaz iz glavnog programa, odnosno kraj glavnog programa. U ovom slučaju, glavni program vraća vrednost 0 (nula).

Red:

```
printf("Hello world!\n");
```

- Štampa tekst „Hello world“ na ekran. Funkcija **printf** je definisana u datoteci **stdio.h** i ona na standardni izlaz (**ekran**) ispisuje niz karaktera koji je njen argument. Nizovi karaktera su ograničeni dvostrukim navodnim znacima ("" ). Karakter **\n** koji se nalazi na kraju ovog niza je specijalni karakter koji **označava novi red**.

# Struktura programa



Osim karaktera za novi red, postoje sledeći specijalni karakteri:

- `\t` tabulator,
- `\b` backspace,
- `\“` dvostruki navodnici,
- `\\` obrnuta kosa crta (backslash).

# Uvod u korišćenje *Linux* operativnog sistema



**Linux** je jezgro operativnog sistema čiji je razvoj započeo Linus Torvalds 1991. godine kao student Univerziteta u Helsinkiju.

- Pored jezgra, za funkionisanje jednog operativnog sistema neophodni su **sistemske alati i bibliotetke iz GNU projekta**, pa se stoga ova kombinacija jednim imenom naziva GNU/Linux.
- Za razliku od komercijalnih operativnih sistema koje kontroliše određena matična kompanija, Linux je **slobodan za distribuiranje i korišćenje**.
- Postoje **verzije Linuxa** za različite hardverske platforme, kao što su npr. ARM, PowerPC ili Sun UltraSPARC.
- Treba obratiti pažnju da Linux pravi razliku između malih i velikih slova (on je **case sensitive**).
  - *Primer: imena “pera” i “Pera” označavaju dva različita korisnika*
- Komunikaciju sa korisnikom putem komandne linije omogućuje program koji se zove **shell**, i koji ima ulogu interpretera komandi. U *Linux distribucijama najčešće korišćeni shell je bash* (Bourne Again).

# Uvod u korišćenje *Linux* operativnog sistema



- Nakon pokretanja *basha*, ispisuje se njegov odziv, odnosno **prompt**. Podrazumevani izgled prompta je:

```
user@computer:~/download$
```

- Sve ranije zapamćene komande se mogu pregledati pritiskom na kursorske tastere ↑ i ↓.
- Ukoliko je potrebno videti prethodne sadržaje ekrana, koristimo **Shift+PgUp**, odnosno **Shift+PgDn**.
- Postoji više načina da se završi rad ***basha*** zadavanjem komande **exit**.
- **Osnovni direktorijum** u *Linux* fajl sistemu je **root**, ili korenski direktorijum i označava se sa “/” (*slash*). *Svi drugi direktorijumi se nalaze ispod njega*:
  - ✓ Direktorijum **etc** sadrži razne konfiguracione fajlove, većinom u tekstualnom obliku.
  - ✓ Direktorijum **bin** sadrži osnovne sistemske programe.
  - ✓ Direktorijum **usr** sadrži korisničke programe i sve što je potrebno za njihovo korišćenje (slike, dokumentacija, razni pomoćni i konfiguracioni fajlovi).

# Uvod u korišćenje *Linux* operativnog sistema



- U svakom direktorijumu se nalaze i dva specijalna poddirektorijuma:
  - “.” označava tekući direktorijum, dok
  - “..” označava nadređeni direktorijum (**parent**), koji se nalazi iznad tekućeg.
- Korisnikov home direktorijum ima posebnu oznaku “~” (tilda)
- Spisak direktorijuma se nalazi u promenljivom okruženju (environment variable) **PATH**.
- Ako treba pokrenuti program koji se nalazi u tekućem direktorijumu (a koji se ne nalazi u PATH-u), potrebno je kucati:

```
./naziv_programa
```

# Uvod u korišćenje *Linux* operativnog sistema



- Komanda koja služi za kretanje po stablu direktorijuma je **cd** (change directory).
- Može se kucati apsolutna putanja:

```
cd /home/pera/drugi/
```

- Ukoliko je potrebno saznati punu putanju do tekućeg direktorijuma, to se može postići kucanjem komande **pwd** (print working directory).
- Prikaz sadržaja tekućeg direktorijuma, omogućuje komanda **ls** (list).
- Jedna od osnovnih operacija pri radu sa fajlovima je njihovo kopiranje. To se postiže komandom **cp** (*copy*). Osnovni oblik ove komande je:

```
cp šta_se_kopira gde_se_kopira
```

- Ukoliko treba obrisati jedan ili više fajlova, koristi se komanda **rm** (remove). *Na primer, ako treba* obrisati sve fajlove koji počinju sa aaa, komanda bi bila:

```
rm aaa*
```

# Uvod u korišćenje *Linux* operativnog sistema



- Novi direktorijum se može kreirati komandom **mkdir** (*make directory*),
- dok se postojeći direktorijum može obrisati komandom **rmdir** (*remove directory*). Treba obratiti pažnju na to da rmdir može obrisati samo prazan direktorijum.
- Najčešće korišćeni tekst editori u *Linux* okruženju su:
  - *vim/gvim* i
  - *emacs*.
- *Pored njih postoji i mnoštvo drugih tekst editora, kao što su:*
  - *nano*,
  - *kwrite*,
  - *nedit*,
  - ***gedit***,
  - *xedit*...

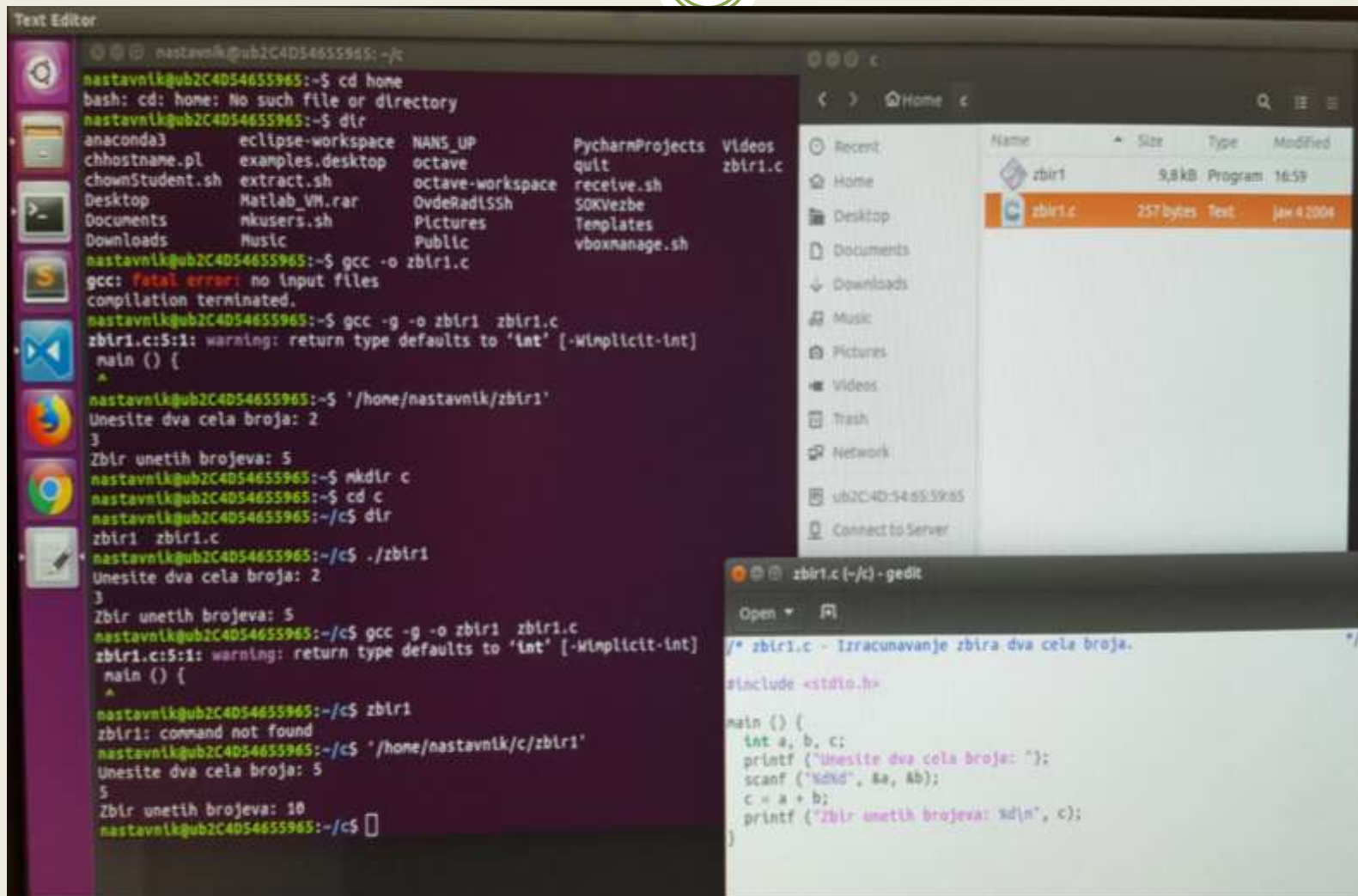
# Kompajler



- C naredbe u sastavu C programa se prvo pišu u **gedit** tekst editoru i fajl se snima sa ekstenzijom “c” .
- Za prevođenje C programa u izvršni program se koristi **gcc** kompajler (*GNU Compiler Collection*), koji se pokreće u Linux-ovom **Terminalu**:  
Primer: **gcc -m32 -g -o primer primer.c**
  - navedena opcija, **-m32**, se koristi u slučajevima kada je neophodno prevođenje 32 bitnih programa na 64 bitnu platformu
  - opcija **-g** omogućava upisivanje informacija neophodnih za **dibagiranje programa**,
  - dok se opcijom **-o** zadaje ime izlaznog fajla tj. naziv **izvršnog programa** (ukoliko se opcija -o izostavi, izvršni program će imati naziv **a.out**)
- Fajlovi se mogu analizirati u Linux-ovom **Exploreru**.



# Kompajler



The screenshot displays a Linux environment with a terminal window on the left and a file manager on the right. The terminal shows the user navigating to the home directory, listing files, and attempting to compile a C program named `zbir1.c`. The compilation process involves using `gcc` to create an executable `zbir1`. The user then runs the program, which prompts for two integers and outputs their sum. The file manager on the right shows the contents of the `~/c` directory, including the `zbir1` program and the `zbir1.c` source file.

```
nastavnik@ub2C4D54655965: ~/c
$ cd ..
$ ls
anaconda3  eclipse-workspace  NANS_UP  PycharmProjects  Videos
chhostname.pl  examples.desktop  octave  quit  zbir1.c
chownStudent.sh  extract.sh  octave-workspace  receive.sh
Desktop  Matlab_VH.rar  OvdeRadSSH  SOKVezbe
Documents  nkusers.sh  Pictures  Templates
Downloads  Music  Public  vboxmanage.sh

nastavnik@ub2C4D54655965: ~/c
$ gcc -o zbir1.c
gcc: fatal error: no input files
compilation terminated.

nastavnik@ub2C4D54655965: ~/c
$ gcc -g -o zbir1 zbir1.c
zbir1.c:5:1: warning: return type defaults to 'int' [-Wimplicit-int]
main () {
^
nastavnik@ub2C4D54655965: ~/c
$ ./zbir1
Unesite dva cela broja: 2
3
Zbir unetih brojeva: 5

nastavnik@ub2C4D54655965: ~/c
$ mkdir c
nastavnik@ub2C4D54655965: ~/c
$ cd c
nastavnik@ub2C4D54655965: ~/c
$ ls
zbir1  zbir1.c

nastavnik@ub2C4D54655965: ~/c
$ ./zbir1
Unesite dva cela broja: 2
3
Zbir unetih brojeva: 5

nastavnik@ub2C4D54655965: ~/c
$ gcc -g -o zbir1 zbir1.c
zbir1.c:5:1: warning: return type defaults to 'int' [-Wimplicit-int]
main () {
^
nastavnik@ub2C4D54655965: ~/c
$ zbir1
zbir1: command not found
nastavnik@ub2C4D54655965: ~/c
$ ./zbir1
Unesite dva cela broja: 5
10
Zbir unetih brojeva: 10
nastavnik@ub2C4D54655965: ~/c
$
```

The file manager shows the following files in the `~/c` directory:

Name	Size	Type	Modified
zbir1	9.8 kB	Program	16:59
zbir1.c	257 bytes	Text	Jan 4 2004

```
/* zbir1.c - Izracunavanje zbira dva cela broja. */
#include <stdio.h>

main () {
    int a, b, c;
    printf ("Unesite dva cela broja: ");
    scanf ("%d%d", &a, &b);
    c = a + b;
    printf ("Zbir unetih brojeva: %d\n", c);
}
```

# Kompajler

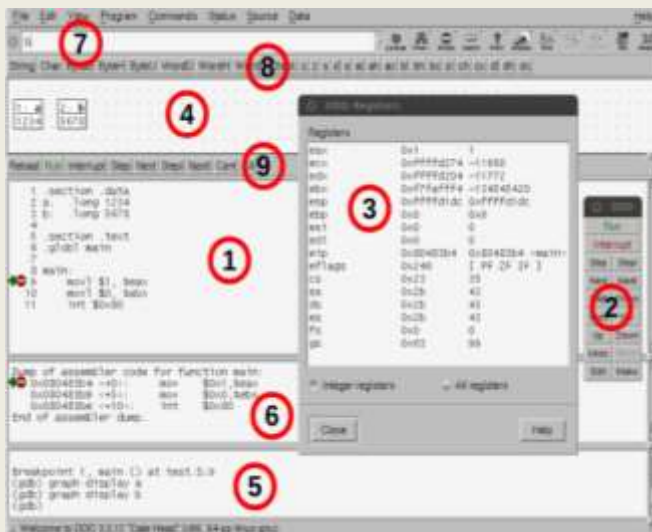
```
nastavnik@ub2C4D54655965:~$ gcc -g -o zbir1 zbir1.c
zbir1.c:5:1: warning: return type defaults to 'int'
main () {
^
nastavnik@ub2C4D54655965:~$ '/home/nastavnik/zbir1'
Unesite dva cela broja: 2
3
Zbir unetih brojeva: 5
nastavnik@ub2C4D54655965:~$ mkdir c
nastavnik@ub2C4D54655965:~$ cd c
nastavnik@ub2C4D54655965:~/c$ dir
zbir1  zbir1.c
nastavnik@ub2C4D54655965:~/c$ ./zbir1
Unesite dva cela broja: 2
3
Zbir unetih brojeva: 5
```

# Dibager

- **Dibager** služi za kontrolisano **izvršavanje programa**, i unutar njega se mogu stalno pratiti i menjati sadržaji. Dibager koji će se koristiti se zove **DDD** (Data Display Debugger) i predstavlja grafičko okruženje za **gdb** (GNU Debugger, osnovni dibager koji radi iz komandne linije Terminala u Linux-u).
- *DDD se poziva komandom:*

```
ddd primer
```

- Glavni prozor je podeljen na nekoliko panela i prozora (koji se mogu prikazati, odnosno sakriti iz **View** i **Status menija**).



- Prozor (1) sadrži izvorni kôd programa.
- Panel (2) sadrži komande vezane za izvršavanje programa.
- Prozor (3) sadrži trenutne vrednosti registara.
- Prozor (4) sadrži vrednosti koje je korisnik odabrao da prikazuje (kada takvih vrednosti nema, panel se ne prikazuje).
- Prozor (5) sadrži komandnu liniju **gdb** (tu se može videti tekstualni oblik svih komandi zadatih u **DDDu**),
- Prozor (6) sadrži izgled programskog kôda u memoriji.
- Prozor (7) Linija za “Display” taster
- Panel (8) Tasteri za prikaz podataka i registara
- Panel (9) Osnovne komande dibagera

# Promenljive



- Promenljive su objekti koji imaju svoje ime i čija se vrednost može menjati tokom izvršenja programa. Definisanje promenljivih u C-u vrši se na sledeći način:

`tip_promenljive ime_promenljive;`

- Četiri osnovna tipa podataka su:
  - `int` - celobrojni tip,
  - `float` - realni tip jednostruke tačnosti,
  - `double` - realni tip dvostruke tačnosti,
  - `char` - znakovni tip, karakter.
- Iz osnovnog celobrojnog tipa se mogu dobiti izvedeni tipovi:
  - `short int` (short),
  - `long int` (long),
  - `unsigned int` (unsigned) – pozitivan celi broj.

<code>auto</code>	<code>break</code>	<code>case</code>	<code>char</code>	<code>const</code>	<code>continue</code>	<code>default</code>	<code>do</code>
<code>double</code>	<code>else</code>	<code>enum</code>	<code>extern</code>	<code>float</code>	<code>for</code>	<code>goto</code>	<code>if</code>
<code>int</code>	<code>long</code>	<code>register</code>	<code>return</code>	<code>short</code>	<code>signed</code>	<code>sizeof</code>	<code>static</code>
<code>struct</code>	<code>switch</code>	<code>typedef</code>	<code>union</code>	<code>unsigned</code>	<code>void</code>	<code>volatile</code>	<code>while</code>

# Štampanje vrednosti



- Funkciji **printf** štampa proizvoljan niz karaktera na standardni izlaz.
- Funkcija ima sledeći oblik:

```
printf("niz_znakova_za_ispis", ime_promjenljive1, ...);
```

- Niz znakova za ispis predstavlja niz karaktera, pri čemu se znakom % pokazuje mesto gde treba da se upišu vrednosti promenljivih, nakon znaka % ide **odgovarajuće slovo**, koje zavisi od tipa promenljive:
  - %d – celobrojni tip, zapisan u dekadnom sistemu,
  - %o – celobrojni tip, zapisan u oktalnom sistemu,
  - %x – celobrojni tip, zapisan u heksadekadnom sistemu,
  - %f – realni tip (**float i double**)
  - %lf – konverzija realne vrednosti dvostruke preciznosti (**double**)
  - %g – drugi specifikator formata za **double** (u printf)
  - %% - karakter '%' (u printf)
  - %c – karakter.
  - printf ("%d", x); // štampa promenljive x u dekadnom sistemu tj. (% definiše memorijski prostor)
  - printf ("%f %f", a, b); // štampa realnih promenljivih a i b
  - printf ("%d %o %x", x, x, x); // štampa promenljive x u dekadnom, oktalnom i heksadekad. sist.
  - printf ("%c", c); // štampa vrednosti promenljive c tipa char
  - printf ("Vrednost promenljive x je %d", x);



# Unos vrednosti sa tastature i aritmetički operatori

- Za tu svrhu se koristi funkcija **scanf** oblika:

```
scanf("format", lista_promjenljivih,...);
```

- **Aritmetički operatori**, kao što im samo ime govori, služe za izvršavanje aritmetičkih operacija. Postoji ukupno pet aritmetičkih operatora:
  - + sabiranje,
  - - oduzimanje,
  - \* množenje,
  - / deljenje,
  - ^stepenovanje (↑ - shift 6 engl. tastatura)
  - % ostatak pri deljenju (samo za cele brojeve).

Svi ovi operatori su binarni, što znači da imaju dva argumenta i njihova svrha je jasna, osim možda poslednjeg.

Npr. izraz  $8\%3$  daje ostatak pri dijeljenju broja 8 sa brojem 3, tj. daje vrednost 2.

Aritmetički operatori	Primer
+	$c = a + b;$
-	$c = a - b;$
*	$c = a * b;$
/	$c = a / b;$
%	$c = a \% b;$
++	$c++;$
--	$c--$

$a = a + 3 \Leftrightarrow a += 3$

$a = a - b \Leftrightarrow a -= b$

$a = a * c \Leftrightarrow a *= c$

$/=, \% = \dots$

$a = a + 1 \Leftrightarrow a += 1 \Leftrightarrow a++ \Leftrightarrow ++a$

$b = b - 1 \Leftrightarrow b -= 1 \Leftrightarrow b-- \Leftrightarrow --b$

$i = j++; \Leftrightarrow i = j; j = j + 1;$

$i = ++j; \Leftrightarrow j = j + 1; i = j;$

# Relacijski i logički operatori

Ovi operatori imaju isti smisao kao i u matematici. U ovu grupu operatora spadaju

- > veće,
- < manje,
- <= manje ili jednako,
- >= veće ili jednako,
- == jednako (dvostruko jednako),
- != različito (! je znak negacije).

Operatori relacija	Primer
>	<code>if (a &gt; b) ...</code>
>=	<code>if (a &gt;= b) ...</code>
==	<code>if (a == b) ...</code>
<	<code>if (a &lt; b) ...</code>
<=	<code>if (a &lt;= b) ...</code>
!=	<code>if (a != b) ...</code>

Osim ovih, u ovu grupu spadaju i logički vezinici.

- && logičko I,
- || logičko ILI. (↑ - shift Ž)

Logički operatori	Primer
&&	<code>if (number &gt; 0 &amp;&amp; number &lt; 10) ...</code>
	<code>if (number &lt; 0    number &gt; 10) ...</code>
!	<code>if ( ! result ) ...</code>

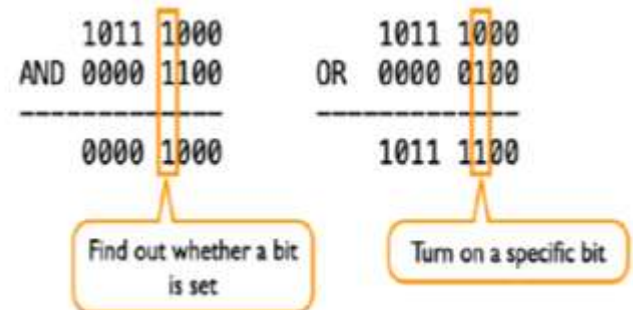
# Bitovni operatori

- Bitovni operatori su operatori koji manipulišu sa cjelobronim tipovima podataka, na nivou bitova. Za potpuno razumevanje rada ovih operatora potrebno je poznavati način zapisivanja celih brojeva u potpunom komplementu.

Postoji 6 operatora za rad s bitovima:

- & binarno I, (oduzimanje -)
- | binarno ILI, (sabiranje +)
- ^ binarno ekskluzivno ILI, ( $8 \wedge 11=3$ )
- << levi pomak (levi shift),
- >> desni pomak (desni shift),
- ~ unarna negacija.

Operatori sa bitima	Primer
&	$c = a \& b;$
	$c = a   b;$
^	$c = a \wedge b;$





# Operatori uvećavanja i umanjivanja



- U C-u postoje dva uobičajena operatora **uvećavanja i umanjivanja** za jedan. To su
  - operatori ++ i --.
  - Operator inkrementiranja ++ uvećava svoj operand za jedan,
  - dekrementiranja -- umanjuje svoj operand za jedan.
- To znači da izrazi:
  - $a=a+1$  i  $a++$ , imaju isto značenje
  - $a=a-1$  i  $a--$  imaju isto značenje.
- Oba operatora se mogu koristiti:
  - kao prefiksni ( $++a$ ) i kao
  - sufiksni ( $a++$ ).
- Razlika je što se u prefiksnom korišćenju vrednost promenljive a uveća pre njenog korišćenja, a u sufiksnom obrnuto.
- Slično kao kod operatora inkrementiranja i dekrementiranja, postoje **skraćeni** zapisi i za operatore u kojima je promenljiva na levoj strani a neki od operatora na desnoj. Za zapis  $i=i+2$  skraćeni zapis za ovaj izraz je  $i+=2$ .

# Prioritet i smer grupisanja



Prioritet	Broj operanada	Operatori	Smer grupisanja
15	2	[ ] ( ) . ->	→
14	1	! ~ ++ -- + - * & (tip) sizeof	←
13	2	* / %	→
12	2	+ -	→
11	2	<< >>	→
10	2	< <= > >=	→
9	2	== !=	→
8	2	&	→
7	2	^	→
6	2		→
5	2	&&	→
4	2		→
3	3	?:	→
2	2	= += -= *= /= %= &= ^=  = <<= >>=	←
1	2	,	→

# Primeri



- Primer 1a. Napisati program koji definiše dve celobrojne promenljive, **dodjeljuje im vrednost** i štampa njihov zbir.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a, b;
    a = 5;
    b = 7;
    int c = a+b;
    printf("Zbir je %d\n", c);
    return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a = 5, b = 7;
    printf("Zbir je %d\n", a+b);
    return 0;
}
```

# Primeri



- Primer 1b. Napisati program koji **sa tastature učitava dva cela broja** i štampa njihov zbir.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a, b;
    scanf("%d", &a);
    scanf("%d", &b);
    printf("Zbir je %d\n", a+b);
    return 0;
}
```

```
1  /* Program procita dva cela broja i štampa njihov zbir.
2
3  #include <stdio.h>
4
5  main ()
6  {
7      int a, b, c;
8      printf ("a,b? ");
9      scanf ("%d%d", &a, &b);
10     c = a + b;
11     printf ("a+b= %d\n", c);
12 }
```

"C:\P (

a,b? 2 3  
a+b= 5

Process r  
Press any

# Primeri



- Primer 2. Napisati program koji štampa veličine memorijskog prostora za osnovne tipove podataka.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("int: %d\n", sizeof(int));
    printf("short: %d\n", sizeof(short));
    printf("long: %d\n", sizeof(long));
    printf("unsigned: %d\n", sizeof(unsigned));
    printf("float: %d\n", sizeof(float));
    printf("double: %d\n", sizeof(double));
    printf("char: %d\n", sizeof(char));
    return 0;
}
```

```
int: 4
short: 2
long: 4
unsigned: 4
float: 4
double: 8
char: 1
```

# Primeri



- Primjer 3. Napisati program koji sa tastature čita dva cela broja i ispisuje njihov zbir, razliku, proizvod i količnik.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int a, b;
    scanf("%d", &a);
    scanf("%d", &b);
    printf("a + b = %d\n", a+b);
    printf("a - b = %d\n", a-b);
    printf("a * b = %d\n", a*b);
    printf("a / b = %f\n", (float)a/b);
    printf("a % b = %d\n", a%b);
    return 0;
}
```

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int a, b;
7      scanf("%d", &a);
8      scanf("%d", &b);
9      printf("a+b=%d\n", a+b);
10     printf("a-b=%d\n", a-b);
11     printf("a*b=%d\n", a*b);
12     printf("a/b=%f\n", (float)a/b);
13     printf("a%b=%d\n", a%b);
14     return 0;
15 }
16
```

"C:\P O D A C

```
3
6
a+b=9
a-b=-3
a*b=18
a/b=0.500000
ab=3
```

Process returned  
Press any key to

# Primeri



- Primer 4. Napisati program za izračunavanje obima kruga i površine kruga.

```
1 //krug.c - Izracunavanje obima kruga i površine kruga
2 #include <stdio.h>
3 #define PI 3.14
4
5 int main()
6 {
7     double r;
8     printf("Poluprecnik?");
9     scanf("%lf", &r);
10    printf("Obim %lf\n", 2*r*PI);
11    printf("Povrsina %lf\n", r*r*PI);
12 }
```

 "C:\P O D A C \US

Poluprecnik? 5

Obim 31.400000

Povrsina 78.500000

Process returned 19

Press any key to co

# Primeri



- Primer 5. Napisati program koji izračunava vrednosti izraza.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("24&12 = %d\n", 24&12);
    printf("17|12 = %d\n", 17|12);
    printf("8^11 = %d\n", 8^11);
    printf("17>>2 = %d\n", 17>>2);
    printf("21<<3 = %d\n", 21<<3);
    printf("~11 = %d\n", ~11);
    return 0;
}
```

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      printf("24&12 = %d\n", 24&12);
7      printf("17|12 = %d\n", 17|12);
8      printf("8^11 = %d\n", 8^11);
9      printf("17>>2 = %d\n", 17>>2);
10     printf("21<<3 = %d\n", 21<<3);
11     printf("~11 = %d\n", ~11);
12     return 0;
13 }
```

"C:\P O D

24&12 = 8  
17|12 = 29  
8^11 = 3  
17>>2 = 4  
21<<3 = 168  
~11 = -12

Process retu  
Press any ke

128	64	32	16	8	4	2	1	→	BCD kód
			1	0	1	0	1	→	(21)
1	0	1	0	1				→	(168)



# Primeri



- Primer 6. Površine trougla u ravni, korišćenjem Heronovog obrasca.

```
3  #include <stdio.h>
4  #include <math.h>
5
6  main () {
7
8      double xA, yA, xB, yB, xC, yC, a, b, c, s, P;
9
10     /* Temena trougla: */
11     printf ("Koordinate temena trougla\n");
12     printf ("- prvo teme? ");
13     scanf ("%lf%lf", &xA, &yA);
14     printf ("- drugo teme? ");
15     scanf ("%lf%lf", &xB, &yB);
16     printf ("- trece teme? ");
17     scanf ("%lf%lf", &xC, &yC);
18
19     /* Stranice trougla: */
20     a = sqrt (pow (xB-xC, 2) + pow (yB-yC, 2));
21     b = sqrt (pow (xC-xA, 2) + pow (yC-yA, 2));
22     c = sqrt (pow (xA-xB, 2) + pow (yA-yB, 2));
23
24     /* Povrsina trougla: */
25     s = (a + b + c) / 2;
26     P = sqrt (s * (s-a) * (s-b) * (s-c));
27     printf ("Povrsina trougla: %f\n", P);
28 }
```

```
"D:\Rešeni zadaci iz C\C1\trougao.exe"
Koordinate temena trougla
- prvo teme? 1
2
- drugo teme? 2
3
- trece teme? 4
9
Povrsina trougla: 2.000000
Process returned 27 (0x1B)
Press any key to continue.
```

# Primeri



- Primer 7. U programu omogućite unos dva cela broja i ispišite njihov zbir, aritmetičku sredinu i zbir kvadrata brojeva.

```
#include <stdio.h>

void main ()
{
    int broj1, broj2;

    printf("\nUpiši dva cela broja: ");
    scanf("%d %d", &broj1, &broj2);

    printf("\nZbir je %d", broj1+broj2);
    printf("\nAritmetička sredina je %.2f", (broj1+broj2)/2.0);
    printf("\nZbir kvadrata brojeva je %d", broj1*broj1+broj2*broj2);

    return;
}
```

# Primeri



- Primer 8. U programu omogućite unos dva broja, broj sati i minuta. Ispišite koliko taj broj sati i minuta iznosi u sekundama.
- Mogući izlaz je: 2 sata i 20 minuta iznosi 8400 sekundi

```
#include <stdio.h>

void main ()
{
    int h, min;
    long int sek;

    printf("\nUpiši broj sati i minuta: ");
    scanf("%d %d", &h, &min);

    sek = (h*60+min)*60;

    printf("\n%d sati i %d minuta iznosi %ld sekundi", h, min, sek);

    return;
}
```

# Primeri



- Primer 9. U programu omogućite unos dve stranice pravougaonika i izračunajte njegovu površinu i obim.

```
#include <stdio.h>

void main()
{
    float a, b, p, o;

    printf("\nUčitaj dužine stranica a i b: ");
    scanf("%f %f", &a, &b);

    p = a*b;
    o = 2*(a+b);

    printf ("\nPovršina je %.2f\nObim je %.2f", p, o);

    return;
}
```

# Primeri



- Primer 10. U programu omogućite unos broja sekundi i ispišite odgovarajuće vreme u satima, minutama i sekundama.
- Predloženi izlaz je: 3722 sekunde iznosi 1 sat, 2 minute i 2 sekunde

```
#include <stdio.h>

void main ()
{
    int ukupno, h, min, sec;

    printf("\nUčitaj ukupan broj sekundi: ");
    scanf("%d", &ukupno);

    h = ukupno / 3600;
    min = (ukupno % 3600) / 60;
    sec = (ukupno % 3600) % 60;

    printf ("\n%d sekundi iznosi: ", ukupno);
    printf ("\n%d sat, %d minute i %d sekunde", h, min, sec);

    return;
}
```

# Primeri



- Primer 11. U programu omogućite unos stranicu **a** istostraničnog trougla. Izračunati obim i površinu trougla.
- Rezultate ispisati kao: Učitaj stranicu a trougla: 4, Obim je : 12, Površina trougla je : 6.93

```
#include <stdio.h>
#include <math.h>

void main()
{
    float a;

    printf("\nUčitaj stranicu a trougla: ");
    scanf("%f", &a);
    printf("\nObim trougla je : %.2f", 3*a);
    printf("\nPovršina trougla je : %.2f", a*a*sqrt(3)/4);

    return;
}
```

# Primeri



- Primer 12. U programu omogućite unos tri broja. Izračunati i ispisati aritmetičku sredinu sa 3 decimalna mesta.

```
#include <stdio.h>

void main()
{
    int br1, br2, br3;
    float ars;

    printf("\nUpiši tri cela broja: ");
    scanf("%d %d %d", &br1, &br2, &br3);

    ars = (br1 + br2 + br3) / 3.0;
    printf("\nAritmetička sredina je %.3f", ars);

    return;
}
```

# Primeri



- Primer 13. U programu omogućite unos koordinata točaka A(x1,y1) i B(x2,y2). Izračunati i ispisati njihovu udaljenost u koordinatnom sistemu.

```
#include <stdio.h>
#include <math.h>

void main()
{
    int x1, y1, x2, y2, dx, dy;
    float c;

    printf("\nUpišite koordinate tačke A(x1 i y1): ");
    scanf("%d %d", &x1, &y1);
    printf("\nUpišite koordinate tačke B(x2 i y2): ");
    scanf("%d %d", &x2, &y2);

    dx = x2 - x1;
    dy = y2 - y1;
    c = sqrt((dx*dx)+(dy*dy));

    printf("\nDve tačke su udaljene %.2f", c);

    return;
}
```



# Primeri



- Primer 14. U programu omogućite unos pozitivnog realnog broja. Izračunati kvadrat, kub i drugi koren tog broja. Rezultat ispisati u redu (sa dva 2 decimalna mesta):
- Unesi pozitivan realan broj: 5, Kvadrat broja 5 je 25.00, kub je 125.00, a koren 2.24

```
#include <stdio.h>
#include <math.h>

void main()
{
    float broj, kv, kub, koren;

    printf("\nUnesi pozitivan realan broj: ");
    scanf("%f", &broj);

    kv = broj*broj;
    kub = broj*broj*broj;
    korijen = sqrt(broj);

    printf("\nKvadrat broja %2f je %.2f", broj, kv);
    printf("\nKub broja %2f je %.2f", broj, kub);
    printf("\nKoren broja %2f je %.2f", broj, koren);

    return;
}
```

# Primeri



- Primer 15. Učitati 4 broja x, y, a, b. Izračunati vrednost sedećeg izraza:

$$\frac{|x - y|}{ab} + \frac{xy}{|a - b|}$$

```
#include <stdio.h>
#include <math.h>

void main()
{
    float x, y, a, b, broj1, broj2;

    printf("\nUčitaj brojeve x i y: ");
    scanf("%f %f", &x, &y);

    printf("\nUčitaj brojeve a i b: ");
    scanf("%f %f", &a, &b);

    broj1 = abs(x-y) / (a*b);
    broj2 = (x*y) / abs(a-b);

    printf("\nZbir brojeva je %.2f", broj1+broj2);

    return;
}
```

# Primeri



- Primer 16. U programu omogućite unos stranice kvadrata. Izračunati površinu, obim i dijagonalu kvadrata (na 2 decimale) npr.
- Unesi stranicu kvadrata: 5,
- Površina je: 25,
- Obim je: 20,
- Dijagonala je: 7.07

```
#include <stdio.h>
#include <math.h>

void main()
{
    int stranica, obim, površina;
    float d;

    printf("\n Unesi stranicu kvadrata: ");
    scanf("%d", &stranica);

    površina = stranica*stranica;
    opseg = 4*stranica;
    d = stranica*sqrt(2);

    printf("\nPovršina je: %d", površina);
    printf("\nObim je: %d", obim);
    printf("\nDijagonala je: %.2f", d);

    return;
}
```

# Primeri



- Primer 17. U programu omogućite unos broja dana. Izračunati koliko to iznosi godina, meseci i dana:
- Unesi broj dana: 2255, 2255 dana = 6 god, 2 mes i 5 dana

```
#include <stdio.h>

void main()
{
    int uk, god, mes, dana;

    printf("\nUnesi broj dana: ");
    scanf("%d", &uk);

    god = uk / 365;
    mes = (uk % 365) / 30;
    dana = (uk % 365) % 30;

    printf("\n%d dana = %d god, %d mes i %d dana", uk, god, mes, dana);

    return;
}
```

# Primeri



- Primer 18. U programu omogućite unos temperature u °C i izračunati koliko je to °F (formula:  $^{\circ}\text{F} = ^{\circ}\text{C} * 9/5 + 32$ ):

```
#include <stdio.h>

void main()
{
    float c, f;

    printf("\nUčitaj temperaturu u °C:");
    scanf("%f", &c);

    f = (c * 9 / 5) + 32;

    printf("\nTemperatura u °F je: %.2f", f);

    return;
}
```

# Primeri



- Primer 19. Ako se vozač vozi automobilom 10 minuta. Izračunaj za zadani broj kilometara kojom je brzinom vozio?

```
#include <stdio.h>

void main()
{
    int t, s, m, km;
    float v;

    printf("\n Upiši broj kilometara:");
    scanf("%d", &km);

    t = 10 * 60;
    s = km * 1000;
    v = (float)s / t * 3.6;

    printf("\n Vozač je vozio brzinom od %.2f km/h", v);

    return;
}
```

# Primeri



- Primer 20. Program treba od korisnika tražiti unos realnih koeficijenata linearne jednačine ( $y=ax+b$ ), izračunati i ispisati njeno rešenje.

```
#include <stdio.h>

void main()
{
    float a, b, x, f;

    printf("\nUnesi koeficijente jednačine a i b: ");
    scanf("%f %f", &a, &b);

    printf("\nUnesi vrednost x: ");
    scanf("%f", &x);

    f=a*x+b;

    printf("\nFunkcija f(%.2f) = %.2f", x, f);

    return;
}
```

Hvala na pažnji!

