

# Lekcija 2 – Uvod u C programiranje

## Pregled

- 2.1 Uvod
- 2.2 Prvi C program: štampanje linije teksta
- 2.3 Drugi C program: sabiranje 2 cijela broja
- 2.4 Memorijski koncepti
- 2.5 Aritmetika u C-u
- 2.6 Jednakost i relacioni operatori
- 2.7 Ključne riječi

# Ciljevi lekcije

- U ovoj lekciji:
  - Napisaćete jednostavne programe u C-u.
  - Koristićete jednostavne ulazno-izlazne naredbe.
  - Upoznaćete osnovne tipove podataka.
  - Shvatićete memorijske koncepte.
  - Koristićete aritmetičke operatore.
  - Shvatićete prioritet aritmetičkih operatora.
  - Napisaćete jednostavne naredbe odlučivanja.

## 2.1 Uvod

- Programski jezik C
  - Struktuiran i disciplinovan pristup kreiranju programa
- Strukturno programiranje
  - Uvodi se u sledećim lekcijama
  - Koristi se kroz čitavo Programiranje I i Programiranje II

## 2.2 Prvi C Program: Štampanje linije teksta

```
1  /* A first program in C */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main()
7  {
8      printf( "Welcome to C!\n" );
9
10     return 0; /* indicate that program ended successfully */
11
12 }
```

Welcome to C!

### Komentari

- Tekst između /\* i \*/ se ignorše
- Koristi se za opis programa, algoritma, itd.
- `#include <stdio.h>`
  - Pretprocesorska direktiva
    - Kompjuter treba da učitati sadržaj određene datoteke
  - `<stdio.h>` dopušta standardne input/output operacije

## 2.2 Prvi C Program: Štampanje linije teksta

- `int main()`
  - C program sadrži jednu ili više funkcija, a tačno jedna od njih mora biti `main`
  - Zagrade označavaju funkciju
  - `int` znači da `main` “vraća” cio broj
  - Vitičaste zagrade `{ i }` označavaju blok
    - Tijela svih funkcija moraju biti unutar `{ i }`.

## 2.2 Prvi C Program: Štampanje linije teksta

- `printf( "Welcome to C!\n" );`
  - Instrukcija kompjuteru da izvede akciju
    - Štampa string (niz karaktera) unutar navodnika ( " ")
  - Cijela linija je naredba (ne obavezno)
    - Sve naredbe moraju završiti sa tačka-zarezom (;)
  - Escape karakter (\)
    - Označava da printf treba da uradi “neuobičajeno”
    - `\n` je newline karakter (prelazak u novi red)

## 2.2 Prvi C Program: Štampanje linije teksta

| Escape sekvenca | Opis  |
|-----------------|---|
| \n              | Newline. Pozicionira kursor na pocetak sledece linije.          |
| \t              | Horizontalni tab. Pomjera kursor na sledeci tab stop.           |
| \a              | Alert. Sistemska zvonca.  |
| \\              | Backslash. Umece backslash karakter u string.                   |
| \"              | Double quote. Umece double quote karakter (navodnike) u string. |
|                 |   |

## 2.2 Prvi C Program: Štampanje linije teksta

- `return 0;`
  - Jedan način izlaska iz funkcije
  - `return 0`, u ovom slučaju, znači da je program upsješno završio rad
- Desna zagrada `}`
  - Označava da je `main` došao do kraja
- Linker
  - Kada se funkcija pozove, linker je locira u biblioteci
  - Umeće je u objektni kod
  - Ako ime funkcije nije ispravno, linker prijavljuje grešku, jer nije u mogućnosti da nađe funkciju u biblioteci



**fig02\_03.c**

```
1  /* Fig. 2.3: fig02_03.c
2      Printing on one line with two printf statements */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main()
7  {
8      printf( "Welcome " );
9      printf( "to C!\n" );
10
11      return 0; /* indicate that program ended successfully */
12
13 } /* end function main */
```

Welcome to C!

**Program Output**

**fig02\_04.c**

```
1  /* Fig. 2.4: fig02_04.c
2      Printing multiple lines with a single printf */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main()
7  {
8      printf( "Welcome\nto\nC!\n" );
9
10     return 0; /* indicate that program ended successfully */
11
12 } /* end function main */
```

```
Welcome
to
C!
```

**Program Output**

## **2.2 Drugi C Program: Sabiranje 2 cijela broja**

- Napisati program koji učitava 2 cijela broja i štampa njihov zbir



## fig02\_05.c

```
1  /* Fig. 2.5: fig02_05.c
2      Addition program */
3  #include <stdio.h>
4
5  /* function main begins program execution */
6  int main()
7  {
8      int integer1; /* first number to be input by user */
9      int integer2; /* second number to be input by user */
10     int sum;      /* variable in which sum will be stored */
11
12     printf( "Enter first integer\n" ); /* prompt */
13     scanf( "%d", &integer1 );        /* read an integer */
14
15     printf( "Enter second integer\n" ); /* prompt */
16     scanf( "%d", &integer2 );        /* read an integer */
17
18     sum = integer1 + integer2;        /* assign total to sum */
19
20     printf( "Sum is %d\n", sum );    /* print sum */
21
22     return 0; /* indicate that program ended successfully */
23
24 } /* end function main */
```

[Outline](#)**Program Output**

```
Enter first integer
45
Enter second integer
72
Sum is 117
```

## 2.2 Drugi C Program: Sabiranje 2 cijela broja

- Kao i u prvom zadatku:
  - Komentari, `#include <stdio.h>` i `main`
- `int integer1, integer2, sum;`
  - Definicija promjenljivih (varijabli)
    - Promjenljiva: memorijska lokacija koja može čuvati vrijednost
  - `int` znači da promjenljiva može imati cjelobrojne vrijednosti kao što su (-1, 3, 0, 47)
  - Imena promjenljivih (identifikatori)
    - `integer1, integer2, sum`
    - Identifikatori: niz slova, cifara (ne može biti na početku) i underscores( `_` )
      - Case sensitive (važna su mala i velika slova)
  - Definicije se pojavljuju prije naredbi
    - Ako naredba referencira nedeklarisanu promjenljivu, dolazi do sintaksne greške (compiler error)

## 2.2 Drugi C Program: Sabiranje 2 cijela broja

- `scanf( "%d", &integer1 );`
  - Preuzimanje vrijednosti od korisnika
    - `scanf` koristi standardni ulaz (najčešće tastatura)
  - U našem slučaju, `scanf` ima 2 argumenta
    - `%d` – označava da bi podataka trebao biti dekadni cio broj
    - `&integer1` – memorijska lokacija za čuvanje vrijednosti
    - `&` malo zbunjuje – za sada, zapamtite da morate dopisati simbol `&` ispred imena promjenljive u `scanf`
  - Pri izvršavanju programa, korisnik reaguje na `scanf` unošenjem (tipkanjem) broja na tastaturi i pritiskom na *enter* (return) taster

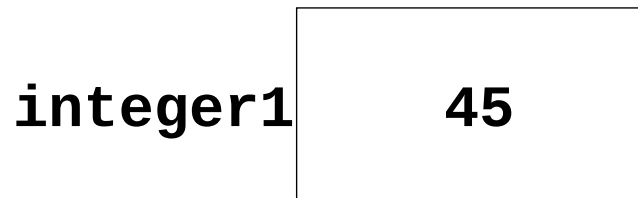
## 2.2 Drugi C Program: Sabiranje 2 cijela broja

- `=` (operator dodjele, assignment operator)
  - Dodjeljuje vrijednost promjenljivoj
  - Binarni operator (ima 2 operanda)
    - `sum = variable1 + variable2;`  
sum dobija vrijednost `variable1 + variable2`;
  - Promjenljiva koja dobija vrijednost je sa lijeve strane
- `printf( "Sum is %d\n", sum );`
  - Slično kao kod `scanf`
    - `%d` znači da će biti štampan dekadni cio broj
    - `sum` specificira koj će se vrijednost štampati
  - Unutar `printf` mogu se vršiti izračunavanja
    - `printf( "Sum is %d\n", integer1 + integer2 );`



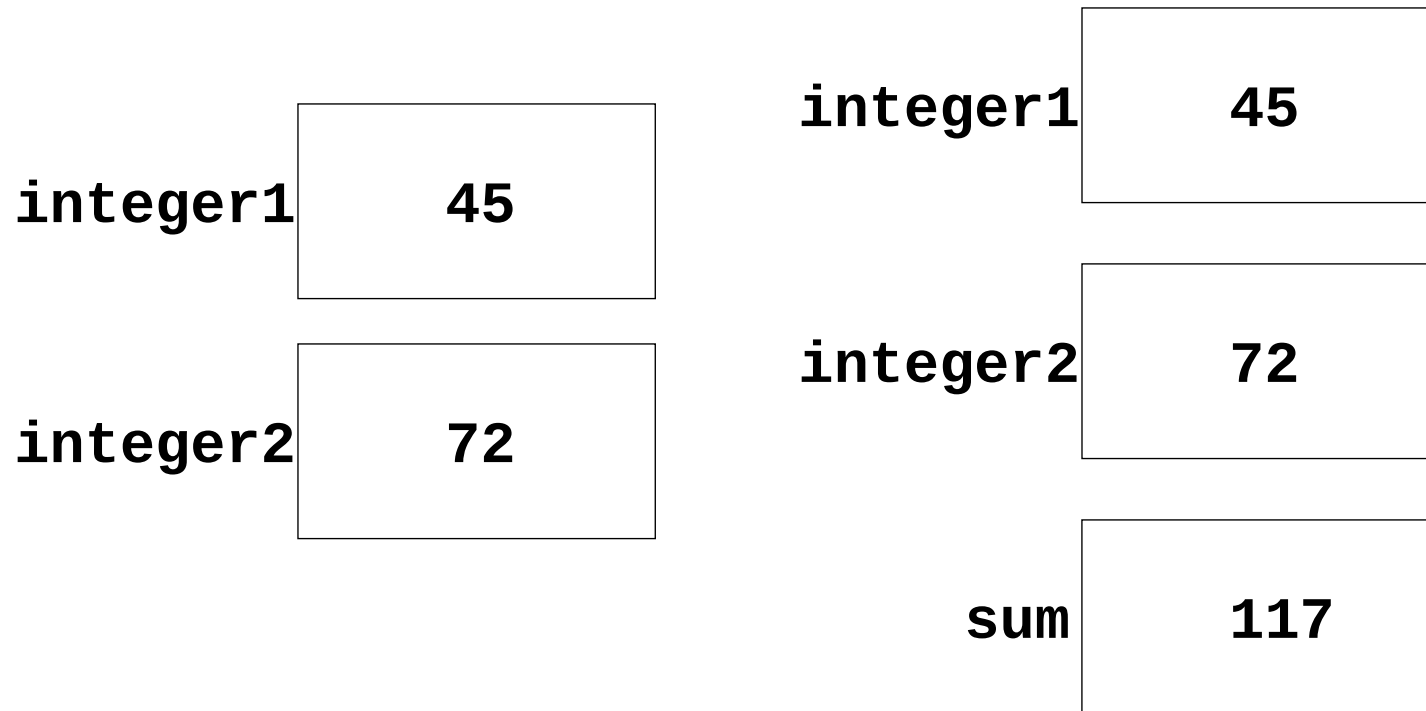
## 2.4 Memorijski koncepti

- Promjenljive
  - Imena promjenljivih odgovaraju lokacijama u operativnoj memoriji
  - Svaka promjenljiva ima ime, tip, veličinu i vrijednost
  - Kada se dodijeli vrijednost promjenljivoj (pomoću `scanf`, na primjer), nova vrijednost zamjenjuje (i uništava) prethodnu vrijednost
  - Čitanjem promjenljive njena vrijednost se ne mijenja
- Vizuelna reprezentacija



## 2.4 Memorijski koncepti

- Vizuelna reprezentacija



## 2.5 Aritmetika

- Aritmetička izračunavanja
  - sabiranje (+), oduzimanje (-), množenje (\*), dijeljenje (/)
  - Cjelobrojno dijeljenje otkida razlomljeni dio
  - $7 / 5$  daje rezultat 1
  - Operator ostatka (modula) (%) vraća ostatak
    - $7 \% 5$  daje rezultat 2
- Prioritet operatora
  - Za aritmetičke operatore – uobičajeni prioriteti
    - Koristite zagrade kada je potrebno
  - Primjer: Prosjek vrijednosti a, b i c
    - Netačno:  $a + b + c / 3$
    - Tačno:  $(a + b + c) / 3$

## 2.5 Aritmetika

- Aritmetički operatori:


| C operacije | Aritmetički operator | Algebarski izraz | C izraz  |
|-------------|----------------------|------------------|----------|
| Sabiranje   | +                    | $f + 7$          | $f + 7$  |
| Oduzimanje  | -                    | $p - c$          | $p - c$  |
| Množenje    | *                    | $bm$             | $b * m$  |
| Dijeljenje  | /                    | $x / y$          | $x / y$  |
| Moduo       | %                    | $r \bmod s$      | $r \% s$ |

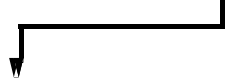
- Pravila prioriteta:

| Operatori(s) | Operacije                    | Prioritet  |
|--------------|------------------------------|--|
| ( )          | Zagrade                      | Prvo se izračunava. Ako su ugnježene, prvo se izračunava unutrašnji par. Ako su na istom nivou, izračunavaju se slijeva udesno |
| *, /, ili %  | Množenje, Dijeljenje, Moduos | Izračunavaju se drugi. Ako ih je više, slijeva udesno  |
| + ili -      | Sabiranje, Oduzimanje        | Poslednji se izračunavaju. Ako ih je više, slijeva udesnot.  |


## 2.6 Jednakost i relacioni operatori

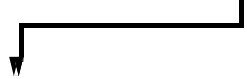
Step 1.  $y = 2 * 5 * 5 + 3 * 5 + 7;$  (Leftmost multiplication)

$2 * 5$  is 




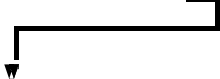
Step 2.  $y = 10 * 5 + 3 * 5 + 7;$  (Leftmost multiplication)

$10 * 5$  is 




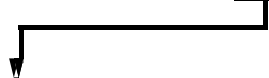
Step 3.  $y = 50 + 3 * 5 + 7;$  (Multiplication before addition)

$3 * 5$  is 



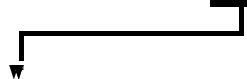
Step 4.  $y = 50 + 15 + 7;$  (Leftmost addition)

$50 + 15$  is 



Step 5.  $y = 65 + 7;$  (Last addition)

$65 + 7$  is 



Step 6.  $y = 72;$  (Last operation—place 72 in y)

## 2.6 Jednakost i relacioni operatori

- Izvršne naredbe (executable statements)
  - Izvode akcije (izračunavanja, ulaz ili izlaz)
  - Izvode odluke (decisions)
    - Na primjer, štampati "polozio" ili "pao" u zavisnosti od ocjene na testu.
- `if` kontrolna naredba (control statement)
  - U ovoj lekciji, samo prosti `if`
  - Ako je uslov tačan (`true`), izvršava se tijelo `if` naredbe
    - `0` je `false`, nenulta vrijednost je `true`
  - Kontrola uvijek nastavlja poslije `if` naredbe
- Ključne riječi
  - Specijalne riječi rezervisane za C
  - Ne mogu biti imena promjenljivih ili funkcija

## 2.6 Jednakost i relacioni operatori

| Algebarski operatori        | C operatori | Primjer C uslova | Znacenje                       |
|-----------------------------|-------------|------------------|--------------------------------|
| <i>Equality Operators</i>   |             |                  |                                |
| =                           | ==          | $x == y$         | $x$ jednak $y$                 |
| $\neq$                      | !=          | $x != y$         | $x$ nije jednak $y$            |
| <i>Relational Operators</i> |             |                  |                                |
| >                           | >           | $x > y$          | $x$ je veci od $y$             |
| <                           | <           | $x < y$          | $x$ je manji od $y$            |
| >=                          | >=          | $x >= y$         | $x$ je veci ili jednak od $y$  |
| <=                          | <=          | $x <= y$         | $x$ je manji ili jednak od $y$ |

**fig02\_13.c (Part 1  
of 2)**

```
1  /* Fig. 2.13: fig02_13.c
2     Using if statements, relational
3     operators, and equality operators */
4  #include <stdio.h>
5
6  /* function main begins program execution */
7  int main()
8  {
9     int num1, /* first number to be read from user */
10    int num2; /* second number to be read from user */
11
12    printf( "Enter two integers, and I will tell you\n" );
13    printf( "the relationships they satisfy: " );
14
15    scanf( "%d%d", &num1, &num2 ); /* read two integers */
16
17    if ( num1 == num2 ) {
18        printf( "%d is equal to %d\n", num1, num2 );
19    } /* end if */
20
21    if ( num1 != num2 ) {
22        printf( "%d is not equal to %d\n", num1, num2 );
23    } /* end if */
24
```



**fig02\_13.c (Part 2 of 2)**

```
25  if ( num1 < num2 ) {
26      printf( "%d is less than %d\n", num1, num2 );
27  } /* end if */
28
29  if ( num1 > num2 ) {
30      printf( "%d is greater than %d\n", num1, num2 );
31  } /* end if */
32
33  if ( num1 <= num2 ) {
34      printf( "%d is less than or equal to %d\n", num1, num2 );
35  } /* end if */
36
37  if ( num1 >= num2 ) {
38      printf( "%d is greater than or equal to %d\n", num1, num2 );
39  } /* end if */
40
41  return 0;    /* indicate that program ended successfully */
42
43 } /* end function main */
```

```
Enter two integers, and I will tell you
the relationships they satisfy: 3 7
3 is not equal to 7
3 is less than 7
3 is less than or equal to 7
```

**Program Output**

**Program Output  
(continued)**

```
Enter two integers, and I will tell you
the relationships they satisfy: 22 12
22 is not equal to 12
22 is greater than 12
22 is greater than or equal to 12
```

```
Enter two integers, and I will tell you
the relationships they satisfy: 7 7
7 is equal to 7
7 is less than or equal to 7
7 is greater than or equal to 7
```

## 2.6 Jednakost i relacioni operatori

| Operators                                       |    |   |    | Associativity |
|---|----|---|----|---------------|
| *   | /  | % |    | left to right |
| +   | -  |   |    | left to right |
| <   | <= | > | >= | left to right |
| ==  | != |   |    | left to right |
| =   |    |   |    | right to left |
| Fig. 2.14 Prioritet i asocijativnost operatora. |    |   |    |               |

## 2.7 Ključne riječi

| Keywords                         |        |          |          |
|----------------------------------|--------|----------|----------|
| auto                             | double | int      | struct   |
| break                            | else   | long     | switch   |
| case                             | enum   | register | typedef  |
| char                             | extern | return   | union    |
| const                            | float  | short    | unsigned |
| continue                         | for    | signed   | void     |
| default                          | goto   | sizeof   | volatile |
| do                               | if     | static   | while    |
| Fig. 2.15 C's reserved keywords. |        |          |          |