Osnovi programiranja

2024/25

Struktura C programa

- Svaki program se sastoji od:
  - deklaracija labela
  - definicija konstanti
  - definicija tipova
  - deklaracija promenljivih
  - definicija funkcija
- Glavni program (glavna funkcija) počinje rezervisanom reči **main()**, a potom rezervisanim znakom **{**, a završava se rezervisanim znakom **}**.

Zdravo, svete!

- Program koji na ekranu ispisuje tekst “hello, world”

```
#include <stdio.h>

main()
{
  printf("hello, world\n");
}
```

- Program u C-u se sastoji od funkcija i promenljivih
- **main** je osnovna funkcija od koje počinje izvršavanje programa
- Svaki program mora imati **main** funkciju
- Funkcija **main** poziva druge funkcije

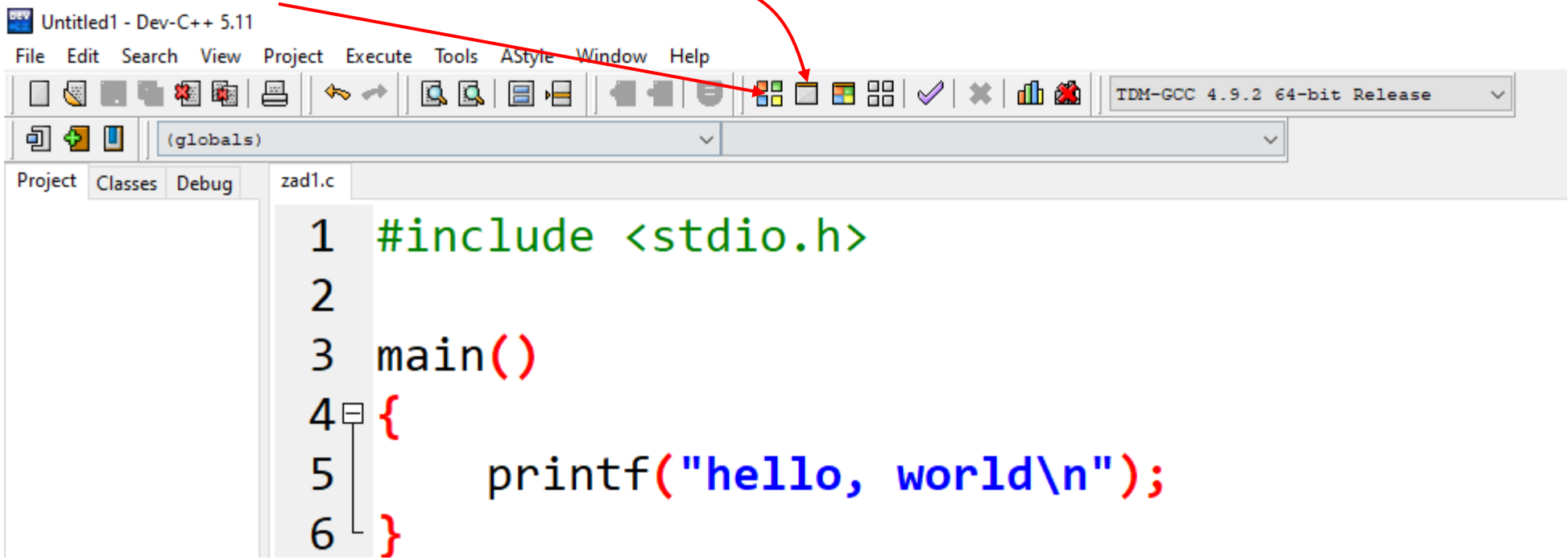
C okruženje

- Zato se naredbe grupišu u program koji je zapisan u datoteci na disku.
  - Programski kod se zapisuje u fajl – skript
  - Ekstenzija fajla je `.c`
- Program se zatim kompajlira i pokreće

Prvi program

- Editor se pokreće iz menija `File`, u okviru osnovnog prozora razvojnog okruženja, izborom opcije `New` -> `Source File`

  Po snimanju na disk, u datoteku sa ekstenzijom .c (u primeru zad1.c), program se kompajlira I pokreće.



Struktura C programa

- Program sadrži niz naredbi, koje se izvršavaju nad određenim podacima u cilju dobijanja traženog rezultata.
- Kako bi program bio što čitljiviji najbolja praksa je pisanje jedne komande u svakom redu. Komande koje pripadaju određenom bloku treba pisati uvučeno za jednu ili dve pozicije.
- Sličan način pisanja treba koristiti i za ostale podređene strukture, kako bi se jasno označila njihova pripadnost drugoj programskoj strukturi.
- Programe uvek treba razbijati na manje logičke segmente koji čine zasebnu celinu. Promenljivama, funkcijama i ostalim elementima programa treba uvek davati nazive koji označavaju njihovu namenu.
- Program u kome su stvari nazvane pravim imenima je znatno čitljiviji i često ne zahteva dodatne komentare u kodu.

Konstante

- Definišu se korišćenjem direktiva pretprocesora jezika C.
- Počinje diretkivom **#define** za kojom slede simboličko ime i vrednost konstante.
- Iza nje se ne nalazi “tačka – zapeta”

```
#include <stdio.h>

// Definisanje konstanti pomoću #define direktive
#define PI 3.14159
#define MAX 100

int main() {
    printf("Vrednost PI je: %f\n", PI);
    printf("Maksimalna vrednost je: %d\n", MAX);
}
```

Makroi sa parametrima

```
#include <stdio.h>

// Makroi sa parametrima
#define SQUARE(x) ((x) * (x))
#define CIRCLE_AREA(r) (PI * SQUARE(r))

int main() {
    double radius = 3.0;
    printf("Kvadrat broja %d: %d\n", a, SQUARE(a));
    printf("Površina kruga sa poluprečnikom %.2f: %.2f\n", radius,
  CIRCLE_AREA(radius));}
```

Promenljive

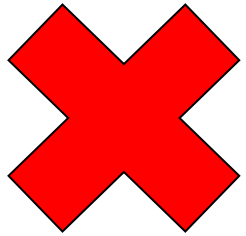
- Promenljive ili varijable su objekti čija vrednost može biti promenjena tokom izvršavanja programa. Svaka promenljiva koja se koristi u programu mora biti prethodno deklarisana.
- Deklaracijom promenljive ne mora da se deklariše i njena vrednost, već samo tip podatka koji će u njoj biti smešten. Na osnovu navedenog tipa podatka svakoj promenljivoj se dodeljuje memoriski prostor odgovarajuće veličine u kome će biti upisana vrednost promenljive.
- Dodeljivanje odgovarajućeg memorijskog prostora (alokacija) se obavlja prilikom pokretanja programa ili potprograma, a njegova pozicija i veličina se ne mogu menjati tokom izvršenja, pa se iz tog razloga ovaj način alokacije naziva **statička alokacija**. Pored ovakvog načina dodeljivanja memorije postoji i **dinamička alokacija**, koji omogućava alokaciju memorije određene veličine u bilo kom trenutku tokom izvršavanja programa.

Promenljive

- Sve konstrukcije se grade od skupa osnovnih simbola jezika koji čine slova, cifre i specijalni znaci.

```
Alfa        ana_voli_milovana
ALFA        Vrlo_dugacko_a_moze_i_duze
Alfa        _ne_preporucuje_se
X55_123     datum_rodjenja
```

```
13a55 - prvi znak ne sme da bude cifra
Alfa,beta   - znakovi interpukcije nisu dozvoljeni
x-z     - operatori nisu dozvoljeni
Int    - rezervisane reči nisu dozvoljene
```



Izrazi

- Izrazi su kombinacije simbola koje definišu poredak i način izračunavanja neke vrednosti korišćenjem:
  - operanda (konstante, promenljive, funkcije)
  - operatora
  - oblik zagrada
- Operatori definišu koje je operacije potrebno izvršiti nad operandima, a zagrade definišu redosled vršenja tih operacija.
- U slučajevima kada redosled vršenja operacija nije određen zagradama, primenjuju se pravila o prioritetu operacija

Operatori	Prioriteti	Asocijativnost
++ -- () [] . -> (type){list}	1 (najveći)	Sa leva na desno
++ -- + - * & sizeof ! ~ (type)	2	Sa leva na desno
* / %	3	Sa leva na desno
- +	4	
<< >>	5	
< <= > >=	6	
== !=	7	
&	8	
^	9	
|	10	
&&	11	
||	12	
?:	13	Sa leva na desno
= += -= *= /= %= <<= >>= &= ^= |=	14	
,	15	Sa leva na desno

Operatori

- Nad svim tipovima podataka moguće je izvršiti određene operacije. Operacija preslikava konačan skup podataka (operande) u konačan skup podataka (rezultate).
- **Operatori** definišu operacije koje je potrebno izvršiti nad operandima da bi se dobio rezultat.

Operacije nad celobrojnim (int) tipom podataka

- Nad celobrojnim operandima se mogu izvoditi sledeće operacije koje daju celobrojni rezultat

- Operacije *, / i % imaju viši prioritet od operacija + i -. Operacije istog prioriteta se izvršavaju sa leva na desno.

Operator	Operacija
*	Množenje
/	Celobrojno deljenje
%	Ostatak pri celobrojnom deljenju
+	Sabiranje
-	Oduzimanje
++	Uvećava vrednost za jedan
--	Umanjuje vrednost za jedan
<<	Bitovsko pomeranje u desno
>>	Bitovsko pomeranje u levo
~	Bitovski komplement
&	Bitovska konjukcija
|	Bitovska disjunkcija
^	Bitovsko XOR

Operacije nad celobrojnim (int) tipom podataka

```
3 * 7 = 21
15 / 2 = 7
15 % 2 = 1
4 + 9 = 13
4 - 9 = -5
-4 + 9 = 5
3 * 7 / 5 % 3 = 1
3 + 2 * 4 - 2 * 5 = 1
5++ = 6
```

```
++5 = 6
5-- = 4
--5 = 4
16 >> 1 = 8
16 << 1 = 32
17 & 9 = 1
17 | 9 = 25
~16 = -17
16 ^ 4 = 20
```

Operacije nad realnim tipovima (float, double, long double) podataka

- Operacije * i / imaju viši prioritet od operacija + i -. Operacije istog prioriteta se izvršavaju sa leva na desno.
- Pri radu sa realnim brojevima treba voditi računa o tome da se oni u memoriji ne beleže apsolutno tačno, već sa određenom tačnošću (određenim brojem decimalnih mesta). Iz tog razloga su i rezultati operacija nad realnim brojevima približni.

Operator	Operacija
*	Množenje
/	Deljenje
+	Sabiranje
-	Oduzimanje

```
3.47 * 7.21 = 25.0187
8.0 / 2.0 = 4.0
1.0 / 3.0 = 0.3333333
1.0 / 3.0 * 3.0 = 0.9999999
2.71 + 6.525 = 9.235
-7.812 – 0.1 = -7.912
6.17 + 2.14 * 3.81 - 4.5 = 9.8234
```

Operacije nad logičkim (bool) tipom podataka

- Operacija ! ima viši prioritet od operacija && i ||. Operacije istog prioriteta se izvršavaju sa leva na desno.

Operator	Operacija
!	negacija
&&	Konjukcija
||	Disjunkcija
~	Bitovski complement
&	Bitovska konjukcija
|	Bitovska disjunkcija
^	Bitovsko XOR

p	q	!p	p&&q	p||q	p^q
netačno	netačno	tačno	netačno	netačno	netačno
tačno	netačno	netačno	netačno	tačno	tačno
netačno	tačno	tačno	netačno	tačno	tačno
tačno	tačno	netačno	tačno	tačno	netačno

Relacijski operatori

- Relacijski operatori omogućavaju poređenje dva operanda istog tipa, a dobijeni rezultat je logičkog tipa.
- Iako su celi i realni brojevi formalno različiti tipovi podataka, korišćenjem relacijskih operatora moguće ih je međusobno porediti.

Operator	Operacija
==	Jednako
!=	Različito
>	Veće
<	Manje
>=	Veće ili jednako
<=	Manje ili jednako
Operatori nad logičkim tipom	
==	Ekvivalencija
!=	Ekskluzivna disjunkcija
<=	Implikacija
Operatori nad skupovima	
==	Jednakost skupova
!=	Različitost skupova
<=	Podskup
>=	Nadskup

Relacijski operatori

- Relacijski operatori su najnižeg prioriteta i obavljaju se tek pošto se prethodno obave svi aritmetički operatori.
- U slučaju primene navedenih operatora na znakovni tip podatka, rezultat se dobije poređenjem njihovih ASCII kodova, tj. njihovih pozicija u ASCII tabeli. Tako je rezultat relacije 'A'<'C' jednak tačno, pošto je ASCII kod znaka 'A' 65, a znaka 'C' 67.

Izraz	Vrednost
5 <= 0	netačno
(2*2) != (3*3)	tačno
true < false	netačno
(5 > 0) > (5 < 0)	tačno
(5 > 0) == (5 !=0)	tačno
(5 < 7) && (7 < 9) <= (9 < 5)	netačno
'C' < 'M'	tačno

Matematičke funkcije

- Pored operacija za koje su definisani operatori, postoje i operacije koje se mogu izvršiti korišćenjem standardnih matematičkih funkcija.
- Da bi smo koristili ove matematičke funkcije potrebno je da u program uključimo biblioteku *math.h*, tako što na početku programa navedemo **#include <math.h>**.
- Argument funkcije može biti promenljiva i/ili izraz odgovarajućeg tipa.
- Funkcije se u izrazima koriste na sličan način kao i operatori, sa tom razlikom što se, umesto znaka koji predstavlja operaciju, navodi naziv funkcije, a zatim odgovarajući argument u zagradama.

Funkcija	Operacija	Tip argumenta	Tip rezultata
acos(x)	arkus kosinus za vrednost x	double	double
asin(x)	arkus sinus za vrednost x	double	double
atan(x)	arkus tangens za vrednost x	double	double
atan2(y, x)	arkus tangens u radijusu y/x jer prepoznaje kvadrant	double	double
cos(x)	kosunus od x	double	double
cosh(x)	hiperbolički kosunus od x	double	double
sin(x)	sinus od x	double	double
sinh(x)	hiperbolički sinus od x	double	double
tanh(x)	hiperbolički tangens od x	double	double
exp(x)	vraća vrednost e podignutu na stepen x	double	double
log(x)	prirodni logaritam broja x	double	double
log10(x)	logaritam sa osnovom 10	double	double
pow(x, y)	vraća vrednost x podignutu na stepen y	double, double	double
sqrt(x)	koren od x	double	double
ceil(x)	zaokružuje broj x na veću ili jednaku vrednost	double	double
fabs(x)	apsolutna vrednost od x	double	double
floor(x)	zaokružuje broj x na manju ili jednaku vrednost	double	double
fmod(x, y)	ostatak pri deljenju broja x brojem y	double, double	double

Operator dodele

- Operator dodele je operator koji određenoj promenljivoj (levi operand) dodeljuje vrednost izraza sa desne strane (desni operand).
- Dodeljivanje se vrši tako što se prvo izračuna vrednost izraza sa desne strane operatora, a zatim se ta vrednost upisuje u memorijsku lokaciju promenljive sa leve strane operatora.
- Tip vrednosti izraza na desnoj strani mora odgovarati tipu promenljive kojoj se vrednost dodeljuje. Jedini izuzetak je dodeljivanje celobrojnog izraza realnoj promenljivoj.

```
a = 7;
i = i+1;
suma = suma+broj;
radijani = stepeni*PI/180.0;
dobar = (ocena>=2.5) && (ocena<3.5);
ime = “Pera”;
```

Naredbe

- Naredbe ili komande su elementi programa koji nalažu računaru da izvrši određenu akciju.
- Naredbe su međusobno razdvojene znakom ;. Naredbe se u C-u mogu podeliti na neizvršne i izvršne.
  - U neizvršne naredbe spadaju naredbe za definisanje i deklarisanje svega što će biti korišćeno u programu.
  - Izvršne naredbe čine "radni" deo programa i u njih spadaju aritmetičko-logičke, upravljačke i ulazno izlazne naredbe.

Blokovi naredbi

- Blokovi omogućavaju organizovanje naredbi u funkcionalne celine. Svaki blok naredbi počinje rezervisanim znakom **{**, a završava se rezervisanim znakom **}.**
- Sve naredbe u okviru jednog bloka čine celinu i ne mogu se izvršavati odvojeno. Iz tog razloga se svaki blok može posmatrati kao jedna složena komanda.

```
{
a = 7;
b = a+1;
};
```

Komentari

- Koriste se za opis
- Ako je komentar u C-u u jednoj liniji mora da počinje sa //.
- Ako komentar sadrži više linija mora da počunje sa /* i završava sa */.

```
//primer komentara
{
a = 7;
b = a+1;
};
/* komentar
U vise linije */
```

Izlazne naredbe

- Nakon obrade podataka u programu, potrebno je rezultate prikazati korisniku ili ih zabeležiti na nekom medijumu.
- Da bi se podaci ispisali na ekranu koristi se naredba programskog jezika C printf, koja vrši ispisivanje navedenih podataka na standardnom izlazu.
- Pored ispisa navedenih podataka, korišćenjem ove naredbe moguće je definisati i format u kome se ti podaci ispisuju, kako bi prikaz bio čitljiviji za korisnika.

```
printf(“%d %d“, 1, 2);
printf(“%d“, 1);
printf(“\n“);
printf(“+ %d\n = 3“, 2);
```

Izlazne naredbe

- Nakon obrade podataka u programu, potrebno je rezultate prikazati korisniku ili ih zabeležiti na nekom medijumu.
- Da bi se podaci ispisali na ekranu koristi se naredba programskog jezika C printf, koja vrši ispisivanje navedenih podataka na standardnom izlazu.
- Pored ispisa navedenih podataka, korišćenjem ove naredbe moguće je definisati i format u kome se ti podaci ispisuju, kako bi prikaz bio čitljiviji za korisnika.

```
printf(“%d %d“, 1, 2);
printf(“%d“, 1);
printf(“\n“);
printf(“+ %d\n = 3“, 2);
```

Specijalne sekvence

- **\n** novi red
- **\t** tabulator
- **\b** povratnik (backspace)
- **\"** navodnik
- **** za obrnutu crtu (backslash)