ALGEBRA

PROGRAMIRANJE

Predavanje 07 – Funkcije

Ishod učenja 3

1

Uvod

- Funkcija je blok koda koji obavlja specifičan posao
- Funkcija obično ima ime
  - Ili može biti anonimna (lambda izraz)
- Funkcija se piše jednom, a koristi puno puta
- Funkcija je dobra alternativa kopiranju dijelova koda
- Tipična upotreba: primi podatke, obradi ih, vrati rezultat
- Ponekad se naziva i rutina, potprogram, procedura, metoda, itd.

ALGEBRA

2

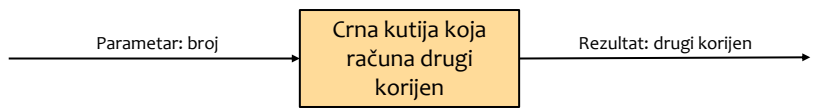
Dvije uloge kad pričamo o funkcijama

ALGEBRA

3

Princip crne kutije

- Funkcija koristi drugu funkciju kao crnu kutiju
- Korisnik crne kutije zna sljedeće:
  - Što crna kutija radi
  - Koje parametre moramo dati crnoj kutiji
  - Koji rezultat (izlaz) će proizvesti crna kutija
- Korisnik crne kutije ne zna kako crna kutija obavlja svoj posao



- Primjer crne kutije koju smo već koristili: `getline()`

ALGEBRA

4

Ugrađene funkcije

- C++ dolazi s puno funkcija spremnih za korištenje
  - Već smo koristili `main()` i `getline()`
  - `size()`, `length()`, `ignore()` su metode: funkcije koje se pozivaju na objektima
  - https://www.cplusplus.com/reference/
- Nekoliko korisnih funkcija:
  - `abs(n)` – vraća apsolutnu vrijednost of *n*, nalazi se u cstdlib
  - `sqrt(n)` – vraća drugi korijen od *n*, nalazi se u cmath
  - `pow(n, exp)` – vraća n^{exp}, nalazi se u cmath
  - `atof(string)` – vadi `double` iz C stringa, nalazi se u cstdlib
  - `atoi(string)` – vadi `int` iz C stringa, nalazi se u cstdlib

ALGEBRA

5

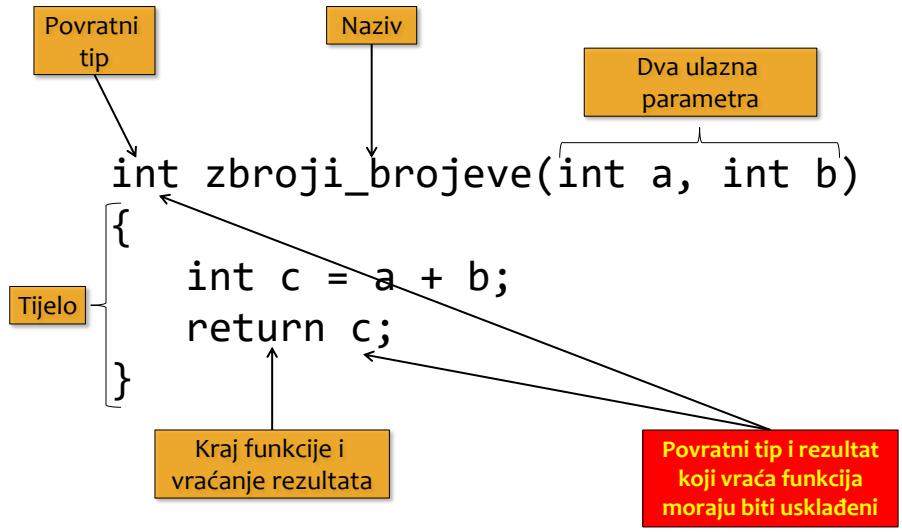
Sastavni elementi funkcija

- Funkcija se sastoji od sljedećih elemenata:
  - **Naziv** – kao kod imenovanja varijabli
  - **Ulazni parametri (argumenti)** – nula ili više parova tip/naziv odvojenih zarezima
  - **Povratni tip** – tip podataka koji vraća funkcija (ako ne vraća ništa, koristimo ključnu riječ **`void`**)
  - **Tijelo funkcije** – niz naredbi za rješavanje problema
  - **Naredba `return`** – radi dvije stvari:
    - Završava izvršavanje funkcije
    - Vraća vrijednost pozivatelju (ako funkcija nije void)
    - Može se izostaviti ako je funkcija void

ALGEBRA

6

Primjer funkcije

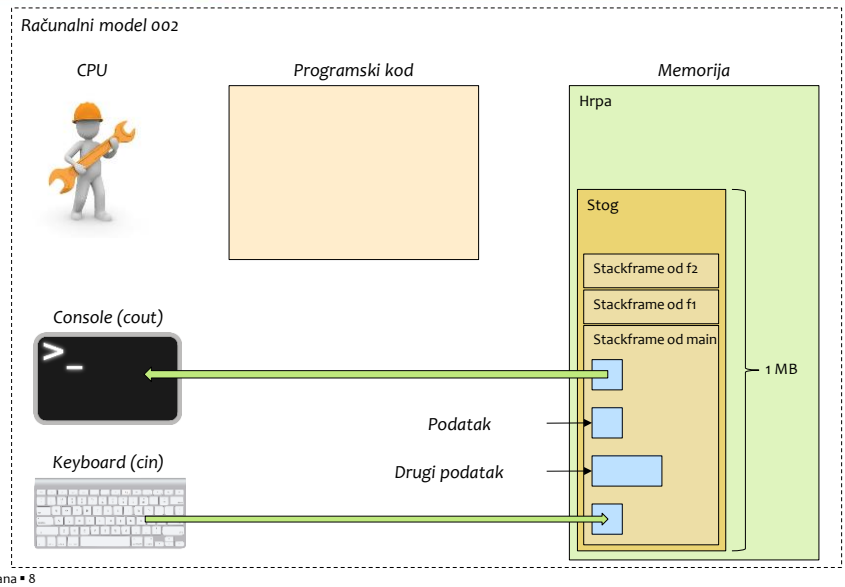


Strana ▪ 7

ALGEBRA

7

Model 002



Strana ▪ 8

ALGEBRA

8

Stog i *stackframovi*

- Kad se program počne izvršavati, 1 MB memorije se rezervira za stog
- Kad se funkcija `main()` počne izvršavati, njen *stackframe* se smješta na dno stoga
  - Kad deklariramo neki element (varijablu ili polje ili vektor) u `main()`, smještaj se u taj *stackframe*
  - *Stackframe* raste kako deklariramo više elemenata
  - Elementi jedne funkcije su lokalni za tu funkciju
    - Svi parametri funkcije se također smještaju u *stackfram*u te funkcije i smatraju se lokalnim za tu funkciju
  - Funkcija ne može pristupati elementima drugih funkcija

Stog i *stackframovi*

- Kad se pozove druga funkcija `f1()` i počne se izvršavati, novi *stackframe* se smješta iznad prethodnog *stackframa*
  - Novi *stackframe* pripada funkciji `f1()`
  - Novi stackframe se koristi za smještanje varijabli, polja i vektora iz funkcije `f1()`
  - Novi *stackframe* se sad smatra aktivnim => *stackframe* koji pripada funkciji `main()` postaje neaktivan
  - CPU vodi računa o tome koji *stackframe* je trenutno aktivan

Stog i *stackframovi*

- Kad funkcija završi, njen *stackframe* se uklanja sa stoga
  - Neće se zaista obrisati već će samo biti proglašen slobodnim prostorom
- Prvi *stackframe* ispod njega postaje aktivan
- Kad funkcija `main()` završi, završava i program
  - Svi resursi se vraćaju operacijskom sustavu

ALGEBRA

11

Crtajmo *stackframove* dok se program izvršava

```
void second(int n) {
    n++;
}
void first() {
    string hi = "Hi from first function.";
    cout << hi << endl;

    int number = 22;
    second(number);
    cout << number << endl;
}
int main() {
    int a = 44;
    first();
    second(a);
    return 0;
}
```


ALGEBRA

12

Gdje pisati funkcije

- Opcija 1: iznad funkcije `main()`

```
int add_numbers(int a, int b) {
   return a + b;
}

int main() {
     cout << add_numbers(12, 3) << endl;
     return 0;
}
```

Definicija

ALGEBRA

13

Gdje pisati funkcije

- Opcija 2: prototip se piše iznad funkcije `main()`, definicija se piše ispod funkcije `main()` (ili bilo gdje drugdje)

```
int add_numbers(int a, int b);

int main() {
     cout << add_numbers(12, 3) << endl;
     return 0;
}

int add_numbers(int a, int b) {
   return a + b;
}
```

Prototip (najava)

Definicija

ALGEBRA

14

Pozivanje funkcija

- Funkciju pozivamo tako da:
  1. Napišemo njen naziv
  2. U okruglim zagradama navedemo vrijednosti za svaki od njenih parametara (ako funkcija ne prima parametre, zagrade ostavljamo praznima)
     - Redoslijed vrijednosti mora odgovarati redoslijedu parametara
     - Vrijednosti se kopiraju u parametre
- Funkciju možemo pozivati iz funkcije `main()` ili iz bilo koje druge funkcije
  - Ako funkciju zovemo iz te iste funkcije, onda kažemo da je funkcija rekurzivna

ALGEBRA

15

Primjeri pozivanja funkcije

- Ako je funkcija definirana kao

```
void ispisi(int a, int b) {
    for (int i = a; i <= b; i++) {
        cout << i << endl;
    }
}
```

- Koji pozivi su ispravni?

```
ispisi(3);
ispisi(1, 5);
ispisi(1, 5, 6);
int i = 1;
ispisi(i, i + 4);
ispisi(1, "10");
```


ALGEBRA

16

Vraćanje vrijednosti iz funkcije

- Funkcija može vratiti nula ili jednu vrijednost
  - Koristimo:

```
return;   // za void funkcije
return x; // za non-void funkcije
```

  - Tip podataka od x mora biti jednak povratnom tipu funkcije
- Nakon poziva, vrijednost koju vrati funkcija možemo ili izravno koristiti ili spremiti u varijablu pa onda koristiti
  - Primjer izravnog korištenja:

```
cout << add_numbers(7, 8) << endl;
```

  - Primjer spremanja u varijablu pa naknadnog korištenja:

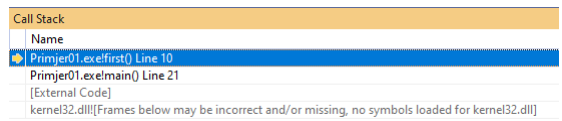
```
int sum = add_numbers(7, 8);
cout << sum << endl;
```


ALGEBRA

17

Debugging s funkcijama

- Kad je žuta strelica zaustavljena na pozivu funkcije, imamo dvije opcije:
  - Sa Step into (F11) ćemo ući u funkciju
  - Sa Step over (F10) ćemo izvršiti funkciju bez ulaska u nju
- Kad smo unutar funkcije, Step out (shift+F11) će izvršiti funkciju do kraja i izaći iz funkcije
  - Korisno ako dođete u tuđi dio koda u kojem ne želite biti
- Pogled Call Stack prikazuje *stackframove*



ALGEBRA

18

Primjer: *debugging* s funkcijama

```
void second(int n) {
    n++;
}

void first() {
    string hi = "Hi from first function.";
    cout << hi << endl;

    int number = 22;
    second(number);
    cout << number << endl;
}

int main() {
    int a = 44;
    first();
    second(a);
    return 0;
}
```


Primjeri

1. Napišite funkciju koja vraća kvadrat proslijeđenog joj cijelog broja. Pozovite funkciju i ispišite rezultat.
2. Napišite funkciju koja ispisuje sve kvadrate brojeva od 1 do zadanog broja. Pozovite funkciju.
3. Napišite funkciju koja vraća manji od dva decimalna broja učitana od korisnika. Pozovite funkciju i ispišite rezultat.
4. Napišite funkciju koja prima jedan cijeli broj i vraća je li njegov kvadrat veći od 50. Pozovite funkciju i u `main()` ispišite poruku je li veći ili nije.
5. Napišite funkciju koja vraća aritmetičku sredinu proslijeđenih 5 cijelih brojeva. Pozovite funkciju i ispišite rezultat.

Primjeri

6. Napišite funkciju koja učitava decimalni broj od korisnika i vraća ga glavnom programu. Pomoću te funkcije učitajte polje od 5 brojeva i nakon toga ih ispišite.
7. Napišite funkciju **prost** koja vraća istinu ukoliko je proslijeđeni broj prost. Učitajte od korisnika broj i ispišite sve proste brojeve od 1 do tog broja.
8. Napišite funkciju koja kao parametar uzima parni cijeli broj ***n***, te ispisuje kvadrat koji ima ***n*** redaka i stupaca od kojih je prvih pola stupaca sastavljeno od znaka '*', a drugih pola od znaka '#'. Ako je proslijeđen neparni broj, ne ispisati ništa.

Primjeri

9. Napišite funkciju koji računa *i*-tu potenciju broja (bez korištenja funkcije pow()). U glavnom programu učitajte brojeve *n* i *x* i ispišite sve potencije broja *n*, od 1 do *x*.
10. Napišite funkciju **index_of()** koja prima string *s*, char *c* i cijeli broj *i* te vraća indeks prvog mjesta u *s* na kojem se nalazi *c*, ali na mjestu *i* ili iza. Ako nema takvog mjesta, funkcija vraća -1. Primjerice, ako bi bilo *s* = "ponedjeljak", *c* = 'e' i *i* = 0, funkcija bi vratila 3. Ako bi bilo *i* = 4, funkcija bi vratila 6, a ako bi bilo *i* = 7, funkcija bi vratila -1. U glavnom programu učitajte string i znak od korisnika i ispišite sva mjesta u stringu na kojima se pojavljuje taj znak.

Primjeri

11. Programirajte igru pogađanja riječi. U nekoj funkciji pripremite polje od 10 riječi i neka funkcija slučajnim odabirom vrati jednu riječ koju korisnik mora pogađati. U svakom potezu korisnik unosi jedno slovo, a program mu iscrtava zamišljenu riječ s otkrivenim svim slovima koje je korisnik do tada odabrao (ostala slova su skrivena znakom '.'). Korisnik ima pravo promašiti 5 slova, nakon čega igra završava neuspješno. Ako korisnik otkrije sva slova, igra završava uspješno. (nastavlja se)

Primjeri

11. (nastavak) Primjer ispisa programa:

```
Unesite slovo: a
...a.
Unesite slovo: k
...ak
Unesite slovo: p
p..ak
Unesite slovo: w
Nema slova: w, ostalo vam je jos 4 pokusaja
Unesite slovo: q
Nema slova: q, ostalo vam je jos 3 pokusaja
Unesite slovo: e
pe.ak
Unesite slovo: t
Cestitamo, pogodili ste rijec: petak
```


Primjeri

12. Napišite program koji od korisnika učitava broj n i iscrtava strelicu okrenutu prema gore visine n. Koristite funkcije gdje je zgodno. Primjerice, za n = 5:

```
    **
   *  *
  *    *
 *      *
*        *
```


Primjeri

13. Napišite program koji simulira sustav za vraćanje ostatka u aparatu za kavu, koristeći funkcije gdje je zgodno. Neka program od korisnika učita ubačenu količinu novca x (x je prirodni broj) i neka mu vrati ostatak od 4 kune (cijena kave) do ubačene količine kuna. Kod vraćanja ostatka na raspolaganju su kovanice od 5 kuna, 2 kune i 1 kune, pri čemu program treba uvijek vratiti što manju količinu kovanica. Svaku vraćenu kovanicu ispišite. Primjerice, ako korisnik upiše x = 17, trebate ispisati:

```
Vracam kovanicu od 5 kuna
Vracam kovanicu od 5 kuna
Vracam kovanicu od 2 kuna
Vracam kovanicu od 1 kuna
```


Zadaci za sljedećih 7 dana

1. Pogledati sljedeće:
   - W08-1 Recursive functions
     - https://youtu.be/WfROBxpdcgU
   - W08-2 Important exam notes
     - https://youtu.be/dmtelGq9fZc

ALGEBRA