# Zadaci – 9. dio, *Funkcije (prvi dio, void-funkcije)*

# Šta je funkcija?

Svaki C++ program ima bar jednu funkciju main. Kada vaš program počne s izvršavanjem, automatski se poziva main, koja može pozvati druge funkcije, od kojih će neke pozvati neke druge.

Dobro dizajnirane funkcije obavljaju specifičan i lahko razumljiv zadatak. Komplikovane zadatke bi trebalo rastaviti na više funkcija. Te funkcije se, u glavnom programu, koriste tako da se navede prototip funkcije (prije funkcije main) a potom se pozivaju iz funkcije main.

Postoje dva tipa funkcija: korisnički-definisane i ugrađene. Ugrađene funkcije su dio paketa vašeg kompajlera (biblioteke funkcija) – njih obezbjeđuje proizvođač, primjer: sqrt(), pow(), sin(), cos()…

## Primjeri korisnički-definisanih funkcija

* Prilikom pokretanja nekog programa, prvo se izvršava funkcija main, bez obzira gdje se ona nalazila. U sljedećem primjeru izvršavat će se prvo linija br. 10.
* Ako funkcija main pozove neku drugu pomoćnu funkciju onda će se tek izvršiti pozvana funkcija.
* U liniji br. 14 se vrši poziv funkcije ispisi\_poruku

Analizirajte sljedeći program:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17: | #include <iostream>  using namespace std;  void **ispisi\_poruku()**  {  cout << "a: Evo nas u funkciju 'ispisi\_poruku' " << endl;  cout << "b: Kraj - izlazak iz funkcije 'ispisi\_poruku' " << endl;  }  void **main()**  {  cout << "c: Evo nas u funkciji main" << endl;    ispisi\_poruku(); //poziv funkcije  cout << "d: Evo nas OPET u funkciji main" << endl;  } |

Ovaj program se sastoji od :

* funkcije main()
* funkcije ispisi\_poruku()

Kada se izvršava ovaj program, prvo se poziva glavna funkcija main, koja će ispisati poruku **c**, a zatim će se izvršiti naredba (u liniji br. 14) za poziv funkcije ispisi\_poruku. Ova funkcija će ispisati poruku **a** i **b** (linija br. 6 i br. 7). Nakon završetka funkcije (izvršenja njene definicije), funkcija vraća vrijednost kroz svoje ime na mjesto odakle je pozvana, nastavlja se izvršavanje funkcije main od onog mjesta poziva funkcije (linija br. 15), tj. izvršit će se naredba za ispis poruke **d** (linija br. 16). Zatim slijedi kraj čitavog programa zatvaranjem vitičaste zagrade funkcije main u liniji br. 17.

Znači, ovaj program će ispisat poruke sljedećim redoslijedom: c, a, b, d.

Iz ovog primjera se vidi da je ova najjednostavnija funkcija ispisi\_poruku građena na isti način kao i funkcija main. Obje funkcije su tipa void, što znači da **nemaju** u tijelu naredbu **return**, i ne vraćaju nikakvu vrijednost onome ko ih poziva. Funkciju main poziva operativni sistem, a funkciju ispisi\_poruku poziva funkcija main. Obje funkcije nemaju ulaznih parametara (argumenata), jer između zagrada – main**()** i ispisi\_poruku**()** – nisu navedeni nikakvi parametri (varijable, vrijednosti).

Zadatak 89:

Koje će poruke ispisati sljedeći program? Zapišite redoslijed poruka i usporedite za rezultatom koji se nalazi na stranici br. 122, npr: b,d,e,c,e,d,a,d,…

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18: | #include <iostream>  using namespace std;  void ispisi\_poruku()  {  cout << "a: Evo nas u funkciju 'ispisi\_poruku' " << endl;  cout << "b: Kraj - izlazak iz funkcije 'ispisi\_poruku' " << endl;  }  void main()  {  ispisi\_poruku(); //poziv funkcije  cout << "c: Evo nas u funkciji main" << endl;  ispisi\_poruku(); //poziv funkcije  ispisi\_poruku(); //poziv funkcije  cout << "d: Evo nas OPET u funkciji main" << endl;  ispisi\_poruku(); //poziv funkcije  } |

*Rješenje se nalazi na kraju dokumenta.*

Zadatak 90:

Koje će poruke ispisati sljedeći program? Zapišite redoslijed poruka i usporedite za rezultatom koji se nalazi na stranici br. 113, npr: b,d,e,c,e,d,a,d,…

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24:  25:  26:  27:  28:  29:  30: | #include <iostream>  using namespace std;  void ispisi\_poruku1()  {  cout << "b: Evo nas u funkciju 'ispisi\_poruku1' " << endl;  }  void ispisi\_poruku2()  {  cout << "d: Evo nas u funkciju 'ispisi\_poruku2' " << endl;  }  void main()  {  cout << "a: Evo nas u funkciji main" << endl;    ispisi\_poruku1();    cout << "c: Evo nas OPET u funkciji main" << endl;  ispisi\_poruku2();  ispisi\_poruku2();  ispisi\_poruku1();  cout << "e: Evo nas OPET u funkciji main" << endl;  ispisi\_poruku1();  ispisi\_poruku2();  } |

*Rješenje se nalazi na kraju dokumenta.*

## Opseg važenja varijabli

*Obavezno zapamtite:* Varijabla definirana u jednom bloku ne vrijedi (ne postoji) izvan tog bloka. (Blok čini bilo koji par vitičastih zagrada „**{ }**“).

Opseg važenja varijable je samo blok u kojem smo je deklarisali. Slijede primjeri sa greškom:

Primjer br. 2:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12: | #include <iostream>  using namespace std;  void main()  {  if (10 > 1)  {  int a = 5;  }  cout << "a = " << a << endl; // kompajler će prijaviti grešku: 'a' undeclared identifier  } |

Primjer br. 2:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15: | #include <iostream>  using namespace std;  void main()  {  if (10 > 1)  {  int b = 4;  }  else  {  int b = 5;  }  cout << "b = " << b << endl; //greska  } |

Primjer br. 3:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10: | #include <iostream>  using namespace std;  void main()  {  {  int b = 4;  }  cout << "b = " << b << endl; //greska  } |

Isto tako i tijelo neke funkcije čini jedan blok. Pa sve lokalne varijable definirane u toj funkciji (bloku) ne vrijede u drugoj funkciji (bloku), evo primjera koji će prijaviti grešku:

Primjer br. 4:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17: | #include <iostream>  using namespace std;  void neka\_funkcija()  {  cout << "a = " << a << endl; //Greška: 'a' : undeclared identifier  }  void main()  {  int a;  a = 12;  cout << "a = " << a << endl;  neka\_funkcija();  } |

## Opseg važenja globalnih varijabli

Globalne varijable vrijede u glavnoj funkciji main i u svim ostalim pomoćnim funkcijama datog programa, jer se deklaracija globalna varijable ne vrši ni u jednom bloku.

Globalne varijable se najčešće deklarišu na početku programa (npr. u liniji br. 3). Slijedi prepravljeni (prethodni) program sa globalnom varijablom a tipa *int:*

Primjer br. 4:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17: | #include <iostream>  using namespace std;  int a; // ovo je sad globalna varijabla  void neka\_funkcija()  {  cout << "a = " << a << endl; // ispisat ce se poruka: 'a = 12'  }  void main()  {  a = 12;  cout << "a = " << a << endl; // ispisat ce se poruka: 'a = 12'  neka\_funkcija();  } |

Zadatak 91:

Analizirajte sljedeći program. Koje će poruke ispisati?

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24: | #include <iostream>  using namespace std;  int a = 10; //globalna varijabla  void neka\_funkcija()  {  a++;  cout << "a = " << a << endl;  }  void main()  {  a = 12;  cout << "a = " << a << endl;    neka\_funkcija();    cout << "a = " << a << endl;    neka\_funkcija();    cout << "a = " << a << endl;  } |

*Rješenje se nalazi na kraju dokumenta.*

Zadatak 92:

Dovršite sljedeći program tako da radi na sljedeći način:

**Koliko puta zelis da ti nesto kazem? ... 3**

**1. Peace be upon you!**

**2. Peace be upon you!**

**3. Peace be upon you!**

**Kraj programa**

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  ...  22: | #include <iostream>  using namespace std;  void neka\_funkcija()  {  cout << "Peace be upon you! \n";  }  void main()  {  int n;  cout << "Koliko puta zelis da ti nesto kazem? ... ";  ...  } |

*Rješenje se nalazi na kraju dokumenta.*

Zadatak 93:

Napravite program u kome će te u funkciji f1() ispisati koliko ima brojeva (u opsegu od **1** do **10000**) koji su djeljivi sa 7 i koji će u funkciji f2() ispisati koliko ima brojeva koji su djeljivih sa 8 (u opsegu od **1** do **10000**).

U funkciji main se trebaju nalaziti samo sljedeće naredbe:

|  |  |
| --- | --- |
| ...  32:  33:  34:  35:  36: | ...  void **main**()  {  f1(); //poziv funkcije f1  f2(); //poziv funkcije f2  } |

*Rješenje se nalazi na kraju dokumenta.*

## Parametri funkcije

Do sada smo vidjeli da, kada pozivamo istu funkciji više puta, svaki put se ispisuje isti rezultat. Sada ćemo vidjeti kako za istu funkciju koju pozovemo možemo dobiti različite rezultate. To bi mogli iskoristiti u prethodnom programu (pogledajte program u rješenju zadatka br. 114).

Obje funkcije f1() i f2() su skoro iste, samo se razlikuju za broj 7, odnosno 8.

U funkciji main ćemo pozvati funkciju f1, ali sada ćemo proslijediti jedan *aktuelni parametar* toj funkciji prilikom poziva. **Aktuelni parametar** predstavlja vrijednost (broj ili varijabla) koja se nalazi u zagradama u pozivu funkcije:

|  |  |
| --- | --- |
| ...  32:  33:  34:  35:  36: | ...  void **main**()  {  f1(**8**); //poziv funkcije f1 sa aktuelnim parametrom 8  ...  } |

Ovdje smo u liniji br. 34 pozvali funkciju f1 za vrijednost **8**. Poziv iste funkcije možemo vršiti i u liniji br. 35, ali sada sa vrijednošću **7**:

|  |  |
| --- | --- |
| ...  32:  33:  34:  35:  36: | ...  void **main**()  {  f1(8); //poziv funkcije f1 za vrijednost 8  f1(7); //poziv funkcije f1 za vrijednost 7  } |

Kako možete vidjeti, funkcija f2 nam je od ovog trenutka nepotrebna i možemo je izbrisati.

Funkciju main smo prepravili. Potrebno je još prepraviti i definiciju funkcije f1:

Možete primijeti da je parametar funkcije f1 vrijednost tipa *integer* (broj 8 i broj 7).

Da bi funkcija f1 znala šta da radi sa primljenim parametrom, moramo i to 'reći', tako što ćemo u zagradama (linija br. 4) dodati deklaraciju jedne varijable. Ime te varijable može biti bilo koji valjani identifikator. U rješenjima ćemo koristiti imena u1, u2, … što asocira na skraćenicu od riječi 'ulaz'. Ovakva varijabla koja je deklarisana u zaglavlju (definicije) funkcije kao ulazni parametar se naziva **formalni parametar**.

Zadatak 94:

Prepravite funkciju f1. Dodajte jedan formalni parametar u1 tipa *int*.

*Rješenje se nalazi na kraju dokumenta.*

*Pomoć:*

Sada ćemo umjesto konstante 8, korištene u funkciji f1, koristiti varijablu u1. Varijabla u1 će, kada se funkcija prvi put pozove, imati vrijednost 8, dok će drugi put imati vrijednost 7.

## Funkcija sa više parametara

Umjesto običnih brojeva (7, 8) možete prosljeđivati i varijable. A možete prosljeđivati i više parametara u funkciju. Oni se, u pozivu funkcije, moraju navesti istim redoslijedom kao u prototipu i zaglavlju definicije funkcije. Evo primjera poziva funkcije f1 koja sada ima tri aktuelna parametra, a to su:

1. broj sa kojim se vrši provjera djeljivosti
2. početak niza
3. kraj niza

|  |  |
| --- | --- |
| ...  18:  19:  20:  21:  22: | ...  void **main**()  {  f1(8, 1, 1000); //poziv funkcije f1  f1(7, 1, 1000); //poziv funkcije f1  } |

Pošto imamo tri aktuelna parametra, moramo deklarisati tri formalna parametra:

|  |  |
| --- | --- |
| ...  4:  5:  6: | ...  void f1(int u1, int u2, int u3)  {  ... |

Sada možemo u funkciji f1 umjesto brojeva 1 i 10000 koristiti varijable u2 i u3.

Zadatak 95:

Prepraviti prethodni program tako da u glavnoj funkciji main zahtijevate od korisnika da unese dva cijela broja - m za početak i n za kraj niza, zatim još dva cijela broja - a i b. Program treba da pomoću funkcije f1:

* ispiše sa koliko brojeva iz niza [m, n] je djeljiv broj a
* ispiše sa koliko brojeva iz niza [m, n] je djeljiv broj b

*Rješenje se nalazi na kraju dokumenta.*

*Pomoć:*

* u funkciji main ćemo izvršiti unos za m, n, a, b
* pozvat ćemo funkciju f1 i proslijediti a (umjesto 8), kao i vrijednosti m i n
* pozvat ćemo funkciju f1 i proslijediti b (umjesto 7), kao i vrijednosti m i n
* u funkciji f1 ćemo primiti parametre kao u1, u2 i u3 koji su tipa *int*
* kao u prethodnom program, izvršiti ćemo *for*-petlju, ali sada ćemo koristiti varijable u2 i u3, umjesto 1 i 10000
* u funkciji ćemo ispisati vrijednost varijable brojac
* funkciji f1 ćemo proslijediti akuelne parametre redoslijedom: (a, m, n), odnosno (b, m, n), a primat ćemo ih redoslijedom u1, u2, u3, što znači da će u funkciji f1:
  + - u1 preuzeti vrijednost a, odnosno b
    - u2 preuzeti vrijednost m
    - u3 preuzeti vrijednost n

Zadatak 96:

Napravite program u kome će te, pomoću jedne funkcije, ispisati sumu *parnih brojeva* i sumu *kvadrata neparnih* brojeva od m do n. U funkciji main ćete od korisnika tražiti da unese cijeli broj m za početak niza i n za kraj niza.

*Rješenje se nalazi na kraju dokumenta.*

*Pomoć:*

* u funkciji main ćemo izvršiti unos za m, n
* pozvat ćemo funkciju f1 i proslijediti, kao aktuelne parametre, vrijednosti m i n
* u funkciji f1 ćemo primiti parametre kao u1, u2 koji su tipa *integer*
* postaviti ćemo varijable suma1 i suma2 na nulu koje su tipa  *int*,
* suma1 će zbrajati sve parne brojeve
* suma2 će zbrajati sve neparne brojeve
* izvršiti ćemo *for*-petlju sa brojačem i koji se mijenja od u1 do u2
  + - provjeriti ćemo da li je broj i paran, tj. da li je i djeljivo sa 2:

*ako jeste*: onda ćemo povećati suma1 za i

*ako nije*: onda ćemo povećati suma2 za i\*i

* ispisati ćemo suma1 i suma2 na ekran

Zadatak 97:

Napravite program koji će u funkciji main zahtijevati od korisnika da unese dimenzije bazena (širina, dužina, visina) u metrima i koji će pomoću funkcije izračunati i ispisati zapreminu bazenu u litrama. Deklarišite prototip funkcije.

*Rješenje se nalazi na kraju dokumenta.*

## Prototip (deklaracija) funkcije

Funkcije u programi treba koristiti na takav način da se prvo (u polju deklaracija – prije funkcije main) navede njena deklaracija (prototip) a da se definiše van funkcije main. Deklaracija „saopštava“ kompajleru povratni tip, ime i parametre funkcije i tipove parametara. Ni jedna funkcija se ne može pozvati iz neke druge funkcije ako nije prvo deklarisana. Deklaracija se naziva ***prototip***.

Postoje dva načina da se deklariše funkcija:

1. Zapisivanjem prototipa u istu datoteku u kojoj se nalazi vaša glavna funkcija main, tj. iznad funkcije main.
2. Definisanjem funkcije (prije funkcije main ili u funkciji main) prije nego što je pozove neka druga funkcija. U ovom slučaju nije potrebno navoditi deklaraciju funkcije.

Preporučuje se korištenje funkcija na način br. 1.

U svim prethodnim programima nije korišten prototip funkcije.

Prototip je sličan zaglavlju definicije funkcije, tj. sličan je definiciji funkcije bez tijela, samo što završava na tačka-zarez (**;**). Evo razlike:

Preporučuje se ovaj način !

|  |  |
| --- | --- |
| **bez prototipa – slučaj 2** | **sa prototipom - slučaj 1** |
| #include <iostream>  using namespace std;  void **volumen**(int u1, int u2, int u3)  {  long int V;  V = u1 \* u2 \* u3 \* 1000;  cout << "Zapremina: " << V << " m \n";  }  void main()  {  int a, b, c;  cout << "Unesi dimenzije u metrima:\n";  cin >> a >> b >> c;  volumen(a,b,c);  } | #include <iostream>  using namespace std;  void **volumen**(int, int, int);  void main()  {  int a, b, c;  cout << "Unesi dimenzije u metrima:\n";  cin >> a >> b >> c;  volumen(a,b,c);  }  void **volumen**(int u1, int u2, int u3)  {  long int V;  V = u1 \* u2 \* u3 \* 1000;  cout << "Zapremina: " << V << " m \n";  } |

U prototipu se ne moraju navoditi imena varijabli (formalnih parametara). Ako ih koristimo, možemo koristiti imena parametara koja se razlikuju od imena parametara u definiciji funkcije. U prethodnom primjeru smo mogli i ovako navesti prototip:

void volumen(**int** BilaSta, **int** OvoNijePotrebno, **int** OviNaziviSeIgnorisu);

Preporuka je ne navoditi nikakva imena varijabli u prototipu, jer prototip služi samo da „kaže“ kompajleru tip funkcije (u našem slučaju je to void), ime funkcije i tip ulaznih argumenata. U definiciji funkcije moramo tačno navoditi ona imena varijabli koja ćemo koristiti u tijelu definicije funkcije (u našem slučaju: **u1**, **u2**, **u3**). U daljnjim zadacima ćemo koristiti i prototip, ali bez imena varijabli:

void **volumen**(int, int, int);

Zadatak 98:

Napravite program koji će od korisnika zahtijevati unos dva broja, m za početak niza i n za kraj niza. Funkcija main treba pozvati funkciju f1 za svaki cijeli broj iz tog niza [m, n]. Funkcija f1 treba provjeriti da li je broj koji ona prima kvadrat nekog broja, tj. da li korijen tog broja cijeli broj. Samo ako jeste, funkcija treba broj ispisati na ekran.

*Rješenje se nalazi na kraju dokumenta.*

*Pomoć:*

* u funkciji f1 treba provjeriti da li je u1 pozitivan broj
* da bi provjerili da li je korijen broja u1 cijeli broj moramo učiniti sljedeće:
* vrijednost korijena od u1 ćemo smjestiti u varijablu korijen\_**f** koja je tipa ***f****loat*
* vrijednost korijena od u1 ćemo smjestiti u varijablu korijen\_**i** koja je tipa ***i****nt*
* pošto je varijabla korijen\_i tipa *int*, decimalni dio (ako postoji) će se zanemariti
* ako je korijen\_f jednako korijen\_i, što znači je vrijednost korijen\_f cijeli broj, ispisat ćemo na ekran broj u1

Rješenja

|  |
| --- |
| Rješenje zadatka br. **89**: |
| a, b, c, a, b, a, b, d, a, b |

|  |
| --- |
| Rješenje zadatka br. **90**: |
| a, b, c, d, d, b, e, b, d |

|  |  |
| --- | --- |
| Rješenje zadatka br. **91**: | |
| *Opis programa:*   * kada se pokrene program varijabla **a** će imati vrijednost 10, ali se neće nigdje ispisati * poziva se funkcija main(): | |
| * + - varijabla **a** dobiva vrijednost 12     - "**a=12**"se ispisuje na ekran     - iz main poziva se neka\_funkcija()       * **a** se inkrementira sa 12 na 13       * "**a=13**" se ispisuje na ekran     - "**a=13**"se ispisuje na ekran     - iz main poziva se neka\_funkcija()       * **a** se inkrementra sa 13 na 14       * "**a=14**" se ispisuje na ekran     - "**a=14**"se ispisuje na ekran | // linija broj 14  // linija broj 15  // linija broj 17  // linija broj 8  // linija broj 9  // linija broj 19  // linija broj 21  // linija broj 8  // linija broj 9  // linija broj 23 |

Korištenje globalnih varijabli nije preporučeno, ali je moguće. Za korištenje vrijednosti varijabli u različitim funkcijama koristit ćemo prosljeđivanje više varijabli kao parametre (argumente) funkcijama pri njihovom pozivu, kao i vraćanje jedne izlazne vrijednosti kroz return za funkcije koje nisu tipa void, a o tome malo kasnije.

*Nastavljamo ali obavezno bez globalnih varijabli.*

|  |  |
| --- | --- |
| Rješenje zadatka br. **92**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22: | #include <iostream>  using namespace std;  void neka\_funkcija()  {  cout << "Peace be upon you! \n";  }  void main()  {  int n;  cout << "Koliko puta zelis da ti nesto kazem? ... ";  cin >> n;  for (int i=1; i<=n; i++)  {  cout << " " << i << ". ";  neka\_funkcija();  }  cout << "Kraj programa \n";  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **93**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24:  25:  26:  27:  28:  29:  30:  31:  32:  33:  34:  35:  36: | #include <iostream>  using namespace std;  void f1()  {  int brojac; //varijabla koja vrijedi samo za f1()  brojac = 0;  for (int i=1; i<=10000; i++)  {  if (i%8 == 0)  brojac++;  }  cout << "Ukupno djeljivih brojeva sa 8 ima " << brojac << endl;  }  void f2()  {  int brojac; //varijabla koja vrijedi samo za f2()  brojac = 0;  for (int i=1; i<=10000; i++)  {  if (i%7 == 0)  brojac++;  }  cout << "Ukupno djeljivih brojeva sa 7 ima " << brojac << endl;  }  void main()  {  f1(); //poziv funkcije f1  f2(); //poziv funkcije f2  } |

U obje funkcije f1() i f2() smo mogli koristili isti naziv za varijablu brojac. To su različite varijable koje imaju isto ime a kao da se nalaze u različitom prostoru, jer je njihov opseg važenja samo u onoj funkciji (bloku) u kojoj su i deklarisane.

|  |  |
| --- | --- |
| Rješenje zadatka br. **94**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22: | #include <iostream>  using namespace std;  void f1(int u1) // Svi parametri proslijeđeni funkciji f1 primaju se kao integer u varijablu u1.  { // Varijabla u1 vrijedi samo za funkciju f1.  int brojac;  brojac = 0;  for (int i=1; i<=10000; i++)  {  if (i%u1 == 0)  brojac++;  }  cout << "Ukupno djeljivih brojeva sa " << u1 << " ima " << brojac << endl;  }  void main()  {  f1(8); //poziv funkcije f1  f1(7); //poziv funkcije f1  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **95**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24:  25:  26:  27:  28:  29: | #include <iostream>  using namespace std;  void f1(int u1, int u2, int u3)  {  int brojac;  brojac = 0;  for (int i=u2; i<=u3; i++)  {  if (i%u1 == 0)  brojac++;  }  cout << "Ukupno djeljivih brojeva sa " << u1 << " ima " << brojac << endl;  }  void main()  {  int m, n, a, b;    cout << "Unesi prvi i zadnji broj niza: \n";  cin >> m >> n;  cout << "Unesi jos dva broja: \n";  cin >> a >> b;    f1(a, m, n); // poziv funkcije f1 za a, m, n  f1(b, m, n); // poziv funkcije f1 za b, m, n  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **96**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24:  25:  26:  27:  28:  29:  30:  31: | #include <iostream>  using namespace std;  void f1(int u1, int u2)  {  long int suma1, suma2;    suma1 = 0;  suma2 = 0;  for (int i=u1; i<=u2; i++)  {  if (i%2 == 0)  suma1 = suma1 + i;  else  suma2 = suma2 + i\*i;  }  cout << "za niz[" << u1 << ", " << u2 << "]" << endl;  cout << "suma parnih brojeva: " << suma1 << endl;  cout << "suma kvadrata neparnih brojeva: " << suma2 << endl;  }  void main()  {  int m, n;  cout << "Unesi pocetak i kraj niza \n";  cin >> m >> n;    f1(m, n); //poziv funkcije f1 za m, n  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **97**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21: | #include <iostream>  using namespace std;  void volumen(int u1, int u2, int u3)  {  long int V;  V = u1 \* u2 \* u3 \* 1000; // kubni metar pomnozen sa 1000 daje litar  cout << "Zapremina: " << V << " m \n";  }  void main()  {  int a, b, c;  cout << "Unesi dimenzije u metrima:\n";  cin >> a >> b >> c;  volumen(a,b,c);  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **98**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21:  22:  23:  24:  25:  26:  27:  28:  29:  30:  31:  32:  33: | #include <iostream>  #include <math.h>  using namespace std;  void f1(int); // prototip (deklaracija) funkcije  void main()  {  int m, n;  cout << "Unesi pocetak i kraj niza: \n";  cin >> m >> n;  for (int i=m; i<=n; i++)  {  f1(i);  }  }  void f1( int u1)  {  if (u1>=0)  {  float korijen\_f;  korijen\_f = sqrt(float(u1));    int korijen\_i;  korijen\_i = korijen\_f; // korijen\_i će zanemariti decimalni dio od korijen\_f  if (korijen\_i == korijen\_f)  cout << u1 << " = " << korijen\_i << " \* " << korijen\_i << endl;  }  } |