

Zadaci – 10. dio, Funkcije (drugi dio)

(Pozvana) funkcija poziva drugu funkciju

Zadatak 99:

Pogledajte ponovo zadatak broj 110. Da se podsjetimo, zadatak je glasio:

Napravite program koji će od korisnika zahtijevati unos dva broja, m za indeks početka niza i n za indeks kraja niza. Funkcija `main` treba pozvati funkciju `f1` za svaki cijeli broj iz tog raspona $[m, n]$. Funkcija `f1` treba provjeriti da li je vrijednost koju ona prima kvadrat nekog broja. Samo ako jeste, funkcija ga treba ispisati na ekran.

Implementacija definicije funkcije `f1` nam nije sada bitna. Obratite pažnju na poziv i prototip funkcije `f1`.

```
1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  void f1(int);
6:
7:  void main()
8:  {
9:      int m, n;
10:
11:      cout << "Unesi pocetak i kraj niza: \n";
12:      cin >> m >> n;
13:
14:      for (int i=m; i<=n; i++)
15:      {
16:          f1(i);    // ovu funkciju pozivamo (n-m+1) puta
17:      }
18:  }
19:
20: void f1(int u1)
21: {
22:     ...
```

Prepravite gornji program slijedeći navedene zahtjeve:

- definiciju funkcije `f1` nemojte mijenjati
- `for`-petlja sa pozivom funkcije `f1` se mora sada nalaziti u novoj funkciji `f0`
- funkciju `f0` pozovite samo jednom iz funkcije `main`
- formatirajte program tako da ispisuje sljedeće poruke:

Riješen C++ kôd se nalaze na stranici 136.

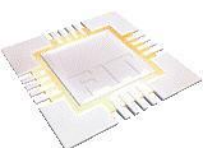
Pomoć:

- ovdje ćete morati koristiti prototipove funkcija
- `for`-petlju izvršite u funkciji `f0`
- funkciji `f0` moramo (iz `main`-a) reći koliko će puta ponoviti `for`-petlju: to možemo činiti na dva načina:

```
Unesi pocetak i kraj niza:
2
33

----Ulazak u funkciju f0----
4 = 2 * 2
9 = 3 * 3
16 = 4 * 4
25 = 5 * 5
---Izlazak iz funkcije f0---

Kraj programa
```



1. funkciji `f0` ćemo proslijediti dva parametra za početak i kraj niza, a to su:
 - aktuelni parametri u funkciji `main`: `m` i `n`
 - formalni parametri u funkciji `f0`: `up` i `uk` (*p* za početak, *k* za kraj)
2. funkciji `f0` ćemo proslijediti samo jedan parametar za ukupan broj ponavljanja *for*-petlje, a to je:
 - aktuelni parametar u funkciji `main`: vrijednost (`n-m+1`)
 - formalni parametar u funkciji `f0`: `ux`

Funkciju `f1` ćemo ostaviti neizmijenjenu!

Zadatak 100:

Slijedi zadatak sa ugniježđenom *for*-petljom u kojem nećete koristiti funkcije.

Koristeći izdvojeni dijagram toka za prost broj koji vam je dat u zadatku 100 (str. 95, workshop br. 9) napravite dijagram toka za sljedeći program:

Program treba izračunati sumu prostih brojeva od *x* do *y*. (Korisnik unosi vrijednosti *x* i *y*).

Dijagram toka riješite iz dva koraka:

- nacrtajte dio dijagrama za provjeru je li broj prost (*for*-petlja i *if-else* iskaza (dvostruki izbor)) - prepravite desni dio dijagrama toka iz zadatka br. 100
- nacrtajte (prepravite lijevi dio dijagrama toka iz zadatka br. 100) ostatak programa sa praznim prostorom u kojem bi se trebao nalaziti prethodno nacrtani dijagram toka

Riješen dijagram toka i bez C++ kôda se nalaze na stranici 137.

Pomoć:

- u lijevom dijagramu toka postavite početnu vrijednost varijable za sumu *s* na nulu
- pošto program treba samo računati sumu a ne ispisivati, morate u desnom dijagramu toka umjesto *if-else*-iskaza (dvostruki izbor) za ispis poruke '*...je prost*' ili '*...NIJE prost*' ubaciti *if*-iskaz (jednostruki izbor) koji će ako je broj *n* prost povećati sumu *s* za vrijednost *n*

Zadatak 101:

Napišite program (prepravite program iz prethodnih *workshop*-ova) koji će koristeći funkciju `ispisi_ako_je_prost` ispisati proste brojeve iz datog raspona.

Riješen C++ kôd se nalaze na stranici 137.

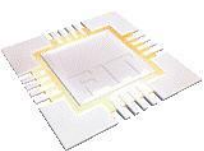
Pomoć:

- funkcija `ispisi_ako_je_prost` treba provjeriti da li je ulazni parametar prost broj
- samo ako je ulazni parametar prost, funkcija treba ispisati taj broj
- funkcija `ispisi_ako_je_prost` treba imati jedan ulazni parametar tipa *int*

Zadatak 102:

Prepravljamo prethodni program:

- ostavite neizmijenjen prototip i definiciju funkcije `ispisi_ako_je_prost`
- u funkciji `main` unesite broj *n* na zahtjev korisnika (koliko puta želi da ponovi program)

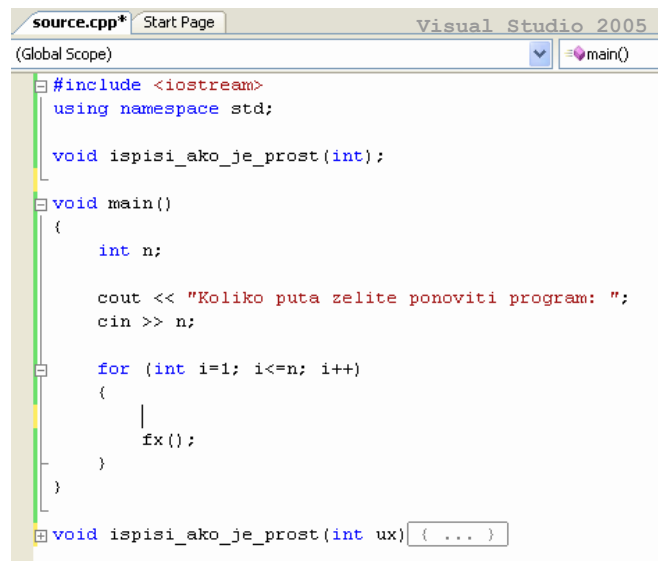


- u *for*-petlju dodajte poziv nove funkcije `fx()` koja nema parametara
- izvršite *for*-petlju `n` puta

Preporučujemo, nakon što ste implementirali definiciju funkcije `ispisi_ako_je_prost` i uspješno testirali u prethodnom zadatku, da sakrijete tijelo funkcije. Ako koristite Visual Studio 2003 ili 2005 ili 2008 to možete učiniti jednim klikom na znak '-' ispred funkcije:

Sakrivanjem kôda koji trenutno nećete mijenjati, lakše ćete moći održavati vaš program, a ujedno program postaje biti pregledniji.

Implementirajte funkciju `fx`, tako da ona poziva funkciju `ispisi_ako_je_prost` za svaki cijeli broj iz niza `[m,n]`. Korisnik unosi brojeve `m` i `n` u funkciji `fx`.



```
source.cpp* Start Page Visual Studio 2005
(Global Scope)
#include <iostream>
using namespace std;

void ispisi_ako_je_prost(int);

void main()
{
    int n;

    cout << "Koliko puta zelite ponoviti program: ";
    cin >> n;

    for (int i=1; i<=n; i++)
    {
        fx();
    }
}

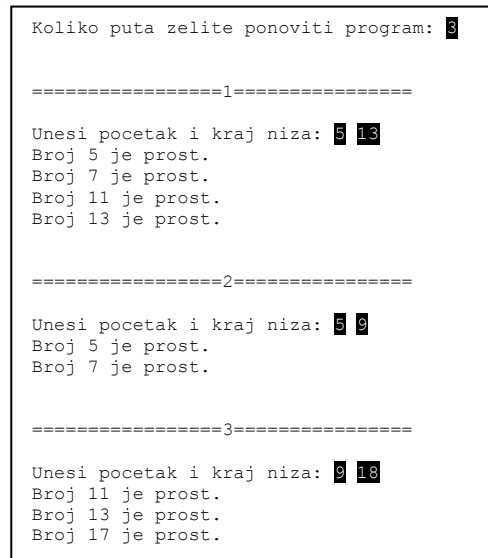
void ispisi_ako_je_prost(int ux) { ... }
```

Program formatirajte na sljedeći način.

Riješen C++ kôd se nalaze na stranici 138.

Dijagram toka funkcije

Dijagram toka za program sa funkcijom se crta kao dva odvojena programa, s tim da se sadržaj funkcije `main` nalazi između **START** i **KRAJ**, a sadržaj pozvane funkcije (potprogram) se nalazi između **'PP'** (početak programa) i **'PUP'** (povratak u program). Izraz za poziv funkcije (potprograma) se crta sa dvostrukim vertikalnim linijama.

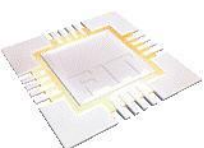


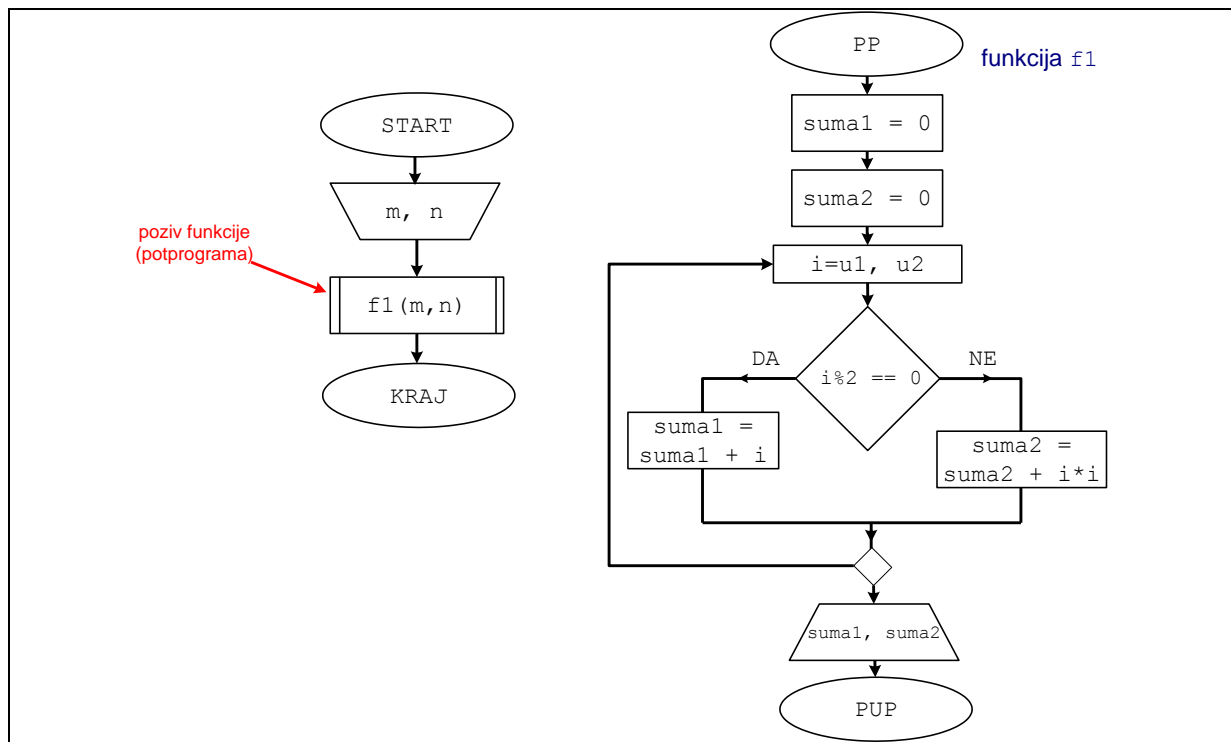
```
Koliko puta zelite ponoviti program: 3

=====1=====
Unesi pocetak i kraj niza: 5 13
Broj 5 je prost.
Broj 7 je prost.
Broj 11 je prost.
Broj 13 je prost.

=====2=====
Unesi pocetak i kraj niza: 5 9
Broj 5 je prost.
Broj 7 je prost.

=====3=====
Unesi pocetak i kraj niza: 9 18
Broj 11 je prost.
Broj 13 je prost.
Broj 17 je prost.
```





Zadatak 103:

Koristeći Visual Studio napravite neki koristan program koristeći DOS-ove naredbe u funkciji system. Evo primjera:

```

7: void formatiraj_disketu()
8: {
9:     cout << "Formatiranje usb memory stika: Pritisni 'Y' i ENTER \n";
10:    system("format f: > null");
11:    system("dir f:");
12: }
    
```

Primjer gotovih funkcija

Zadatak 104:

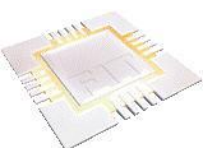
Napraviti program koji će od korisnika zahtijevati unos pet brojeva (a,b,c,d,e). Potrebno je izračunati:

- $y_1 = \sqrt{a}$ (uslov: $a \geq 0$)
- $y_2 = \log_{10} b$ (uslov: $b > 0$)
- $y_3 = \log_e b = \ln b$ (uslov: $c > 0$)
- $y_4 = \cos(c)$ (vrijednost c je u radijanima)
- $y_5 = \sin(c)$
- $y_6 = d^e$ (ako je $d=0$, onda e mora biti različito od 0, jer je 0^0 nedefinisano)

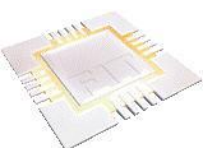
Riješen C++ kôd se nalaze na stranici 138.

Pomoć:

Funkcija u matematici	Funkcija u C++	Uslov
$y = \sqrt{x}$	<code>y = sqrt(x)</code>	$x \geq 0$



$y = \log_{10} x$	<code>y = log10(x)</code>	$x > 0$
$y = \ln(x)$	<code>y = log(x)</code>	$x > 0$
$y = \cos(x)$	<code>y = cos(x)</code>	
$y = \sin(x)$	<code>y = sin(x)</code>	
$y = (x_1)^{x_2}$	<code>y = pow(x1, x2)</code>	<code>(x1 != 0) (x2 != 0)</code> barem jedna vrijednost mora biti različita od nule



Razlika između void-funkcije i funkcije sa povratnom vrijednošću

Analizirajte pozive funkcije u sljedećem dijelu programa! Veoma je bitno da shvatite razlike između ove dvije vrste funkcija. Čitajte kôd pažljivo.

```

7:  ...
8:  void main()
9:  {
10:     float x;
11:     x = sqrt(9.0);    //poziv funkcije koja nije void
12:     cout << "x = " << x << endl;
13:
14:     ispisi_ako_je_prost(17);    //poziv void-funkcije
15: }
16: ...

```

Funkcija `sqrt` (linija br. 11):

- pozivana je funkcija koja **nije** tipa `void`, to znači da:
 - o funkcija `sqrt` dodjeljuje vrijednost (broj 3) onoj varijabli kojoj je funkcija pridružena znakom jednakosti, tj. varijabli `x`
 - o ova funkcija ne ispisuje nikakvu vrijednost na ekran

Zadatak 105:

Šta će sljedeći program ispisati:

```

5:  void main()
6:  {
7:     double x;
8:
9:     x = sqrt(9.0) + 2 * sqrt(4.0);
10:    cout << "x = " << x << endl;
11:
12:    x = pow(sqrt(9.0), 3);
13:    cout << "x = " << x << endl;
14:
15:    x = sqrt(sqrt(16.0));
16:    cout << "x = " << x << endl;
17:
18:    x = pow(sqrt(sqrt(16.0)), sqrt(9.0));
19:    cout << "x = " << x << endl;
20:
21:    if (sqrt(9.0) > sqrt(8.9))
22:        cout << sqrt(9.0) << " je vece od " << sqrt(8.9) << endl;
23: }

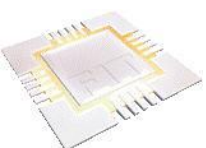
```

„sqrt(9.0)” predstavlja vrijednost 3
 „sqrt(4.0)” predstavlja vrijednost 2
 $x = 3 + 2 * 2$

Funkcija `ispisi_ako_je_prost` (linija br. 14):

- pozivana je funkcija koja **je** tipa `void`, to znači da:
 - o funkcija ne vraća nikakvu vrijednost u program, već je izlaz ove funkcije poruka ispisana na ekran pomoću naredbe `cout`
 - o ova funkcije se ne smije pridružiti (znakom jednakosti ili sl.) nekoj varijabli, niti se smije nalaziti u naredbi `cout` kao što to smije funkcija `sqrt` (linija br. 22)

Pitanje: Koja je povratna vrijednost funkcije `system`? Da li je `void` ili ne?



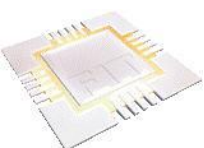
Odgovor: Funkcija `system` je istog tipa kao i funkcija `ispisi_ako_je_prost`.

Zadatak 106:

Pronađite greške u pozivima funkcija u sljedećem programu uz pretpostavku da su prototipovi funkcija ispravni.

```
1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  void f0();           // prototip funkcije
6:  void f1(int);        // prototip funkcije
7:  void f2(int, int);   // prototip funkcije
8:  void f3(float);      // prototip funkcije
9:
10: void main()
11: {
12:     double x = f1();
13:     f0();
14:     f2(3,4);
15:     f2(sqrt(9.0), sqrt(4.0));
16:     f2(pow(2.0), sqrt(4.0));
17:     f3(pow(2.0, 2.0));
18:     f1(f3());
19:     pow(3,4);
20:     x = sqrt(f1(), f3());
21:     cout << f3() << endl;
22:     cout << int (f3()) << endl;
23:     char y = system("PAUSE");
24: }
25: ...
```

Rješenje se nalaze na stranici 139.



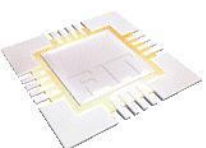
Funkcija sa povratnom vrijednošću (funkcija koja nije *void*)

Pogledajte ponovo funkciju `volumen` iz zadatka br. 120. Funkcija računa zapreminu u litrima za tri ulazna parametra koji predstavljaju dužinu, širinu i visinu bazena u metrima.

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

U funkciji `main` unosimo vrijednosti za tri varijable (`a`, `b`, `c`) i prosljeđujemo ih funkciji `volumen` pri njenom pozivu. U funkciji `volumen` te tri varijable primamo kao `u1`, `u2`, `u3`. Funkcija `volumen` izračunava zapreminu `V` i ispisuje je na ekran. Varijabla `V` koja pamti zapreminu postoji samo u funkciji `volumen` a ne u funkciji `main`. Stoga funkcija `main` 'ne zna' vrijednost zapremine. Kad bi iz funkcije `main` htjeli npr. ispisati vrijednost zapremine na ekran ili u fajl, ne bi smo to mogli učiniti. Vrijednosti izračunate u ovakvim funkcijama, koje su tipa `void` (kao što je to funkcija `volumen`), mogu se ispisati samo na ekran a ne mogu se 'prenijeti' u funkciju `main`.

Prenos vrijednosti iz `void`-funkcije u funkciju `main` možemo postići kad bi koristili globalne varijable. U tom slučaju bi istu varijablu mogli koristiti u funkciji `volumen` za računanje zapremine i u funkciji `main` za neke druge operacije (kao što je ispis u fajl i ponovni ispis na ekran). Slijedi primjer sa globalnom varijablom.




```

1:  #include <iostream>
2:  using namespace std;
3:
4:  int v;           //globalna varijabla
5:
6:  void volumen(int u1, int u2, int u3)
7:  {
8:      v = u1 * u2 * u3 * 1000;
9:      cout << "Zapremina: " << v << " litara \n";
10: }
11:
12: void main()
13: {
14:     int a, b, c;
15:
16:     cout << "Unesi dimenzije u metrima:\n";
17:     cin >> a >> b >> c;
18:
19:     volumen(a,b,c);
20:
21:     cout << "Zapremina u litrima iznosi: " << v << endl;
22: }

```

Rekli smo već da nije preporučljivo koristiti globalne varijable, a u narednim zadacima slijedi objašnjenje. Najbolji način je koristiti funkcije sa povratnom vrijednošću (funkcije koje nisu void).

Za vježbu riješite sljedeći zadatak.

Zadatak 107:

Prepravite prethodni primjer tako da koristite funkciju sa povratnom vrijednošću (funkcija koja nije void) pod imenom `vol` sa tri ulazna parametra. Prepravite samo funkciju `main`. Kasnije ćete dodati prototip i definiciju funkcije `vol`.

Dodajte u funkciju `main` još jedan *if*-iskaz koji će, ako je zapremina veća od 1000, ispisati poruku „Bazen je velik!“.

Napravite dvije verzije:

- jednu sa korištenjem varijable `v`,
- drugu bez korištenja varijable `v`.

Primjer prepravljene funkcije `main` se nalazi na stranici 139.

Prototip i definicija funkcije sa povratnom vrijednošću

Možemo zaključiti, da ime funkcije `vol` predstavlja vrijednost zapremine koja je tipa *integer*. Znači da ova funkcija `vol` (koja je nije void) vraća vrijednost tipa *int*. Stoga ćemo u prototipu funkcije umjesto void koristiti *int* kao povratnu vrijednost.

*** Pod povratnom vrijednošću se ne podrazumijeva vrijednost (zapremina) koja se ispisuje pomoću naredbe `cout` na ekran, već vrijednost (zapremina) koja se vraća iz neke funkcije u funkciju `main`. (Funkcija vraća vrijednost kroz svoje ime). ***

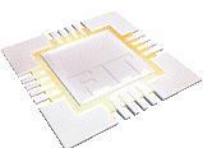
U prototipu funkcije `vol` govorimo da će funkcija vraćati vrijednost tipa *int*:

```

5:  int vol(int, int, int); // prototip

```

Definicija funkcije koja nije void mora sadržavati naredbu koja će vraćati izračunatu vrijednost u funkciju koja je poziva, ovdje `main`. To je naredba `return x;` koja će vraćati vrijednost varijable `x`. Pogledajte liniju br. 9.



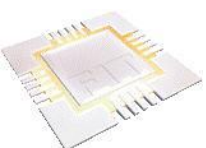
```
5:  int vol(int u1, int u2, int u3)
6:  {
7:      int x;          // varijabla x bi trebala da bude tipa int, jer funkcija vraća int
8:      x = u1 * u2 * u3 * 1000;
9:      return x;       // vrijednost x se vraća u funkciju main
10: }
```

Dodajte funkciju `vol` u prethodni program (zadatak br. 128).

Zadatak 108:

Napravite definicije za sljedeće prototipove funkcija:

```
5:  float sr_vrij(float, float, float); // izlaz je aritmetička srednina
6:  int suma_parnih_brojeva(int, int); // izlaz je suma parnih brojeva od u1 do u2
7:  float max(float, float);           // izlaz je veći broj od u1 i u2
8:  float hipotenuza(float, float);    // izlaz je hipotenuza, a ulaz su dvije katete
9:  float kateta(float, float);        // izlaz je kateta, a ulaz je hipotenuza i kateta
```



Dodajte u sljedeću nepotpunu funkciju main pozive gornjih funkcija.

```
...  
11: void main()  
12: {  
13:     float a, b, c, y1;  
14:     cout << "Unesite tri broja \n";  
15:     cin >> a >> b >> c;  
16:     ...  
17:     cout << "Aritmeticka sredina od tri unesena broja je " << y1 << endl;  
18:  
19:     int m, n, y2;  
20:     cout << "\nUnesite pocetak i kraj niza \n";  
21:     cin >> m >> n;  
22:     ...  
23:     cout << "Suma parnih brojeva od " << m << " do " << n << " je " << y2 << endl;  
24:  
25:     float q, w, y3;  
26:     cout << "\nUnesite dva broja \n";  
27:     cin >> q >> w;  
28:     ...  
29:     cout << "Veci broj od dva unesena: " << y3 << endl;  
30:  
31:     float h, k1, k2;  
32:     cout << "\nUnesite dvije katete jednakokraticnog trougla \n";  
33:     cin >> k1 >> k2;  
34:     ...  
35:     cout << "Hipotenuza: " << h << endl;  
36:  
37:     float hi, ka1, ka2;  
38:     cout << "\nUnesite hipetnuzu i katetu \n";  
39:     cin >> hi >> ka1;  
40:     ...  
41:     cout << "Kateta: " << k2 << endl;  
42: }  
... ..
```

Rješenje se nalazi na stranici 140.



Dodatak: Slijede primjeri za pozive funkcija sa povratnom vrijednošću.

```

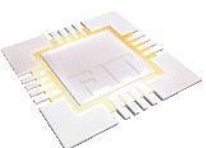
1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  float kub(float, float, float); //prototip
6:
7:  void main()
8:  {
9:      float a1, a2, a3, a4, a5, a6;
10:     a1 = kub(2,3,4);
11:
12:     a2 = kub(2,3,4) + 2;
13:
14:     a3 = kub(4/2, 3, 2*2) + kub(a1/2, a1/2+1, a1/2+2);
15:
16:     //kub(2,3,4) = 3;
17:
18:     //kub(2,3,4)++;
19:
20:     a4 = kub(2,3,kub(1,1,1)*4);
21:
22:     if (kub(a1,a1+1,a1+2) > kub(a2,a2+1, a2+2))
23:         cout << kub(a1,a1+1,a1+2) << " je vece od " << kub(a2, a2+1, a2+2);
24:     else
25:         cout << kub(a1,a1+1,a1+2) << " je manje od " << kub(a2, a2+1, a2+2);
26:
27:     a5 = sqrt(kub(a1, a1/2, a1/3));
28:
29:     a6 = sqrt(sqrt(sqrt(256.0))); // osmi korijen od 256.0
30:
31: }
32:
33:
34: float kub(float u1, float u2, float u3)
35: {
36:     return V = u1 * u2 * u3;
37: }

```

Diagram illustrating function calls and return values:

- kub(2, 3, 4)** returns 24.
- kub(12, 13, 14)** returns 2184.
- kub(12, 12+1, 12+2)** returns 2184.
- kub(2, 3, 4)** returns 24.
- kub(2, 3, 4)** returns 24.
- kub(2, 3, 4)** returns 24.
- kub(1, 1, 1)** returns 1.
- kub(2, 3, 4)** returns 24.

Greška: vrijednost funkcije je konstantna



Rješenja

Rješenje zadatka br. 99: (prvi način)

```

1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  void f1(int);      // prototip (deklaracija) funkcije
6:  void f0(int, int); // prototip (deklaracija) funkcije
7:
8:  void main()
9:  {
10:     int m, n;
11:
12:     cout << "Unesi pocetak i kraj niza: \n";
13:     cin >> m >> n;
14:
15:     f0(m, n);      //funkcija se poziva samo jednom
16:
17:     cout << "Kraj programa \n";
18: }
19:
20: void f0(int up, int uk)
21: {
22:     cout << "\n----Ulazak u funkciju f0---- \n";
23:
24:     for (int i=up; i<=uk; i++)
25:     {
26:         f1(i);      // funkcije se poziva (n-m+1) odnosno (uk-up+1) puta
27:     }
28:     cout << "----Izlazak iz funkcije f0--- \n\n";
29: }
30:
31: void f1(int u1)
32: {
33:     ... // funkcije f1 je nepromijenjena

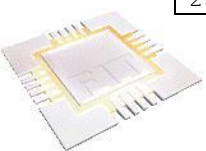
```

Rješenje zadatka br. 100: (drugi način)

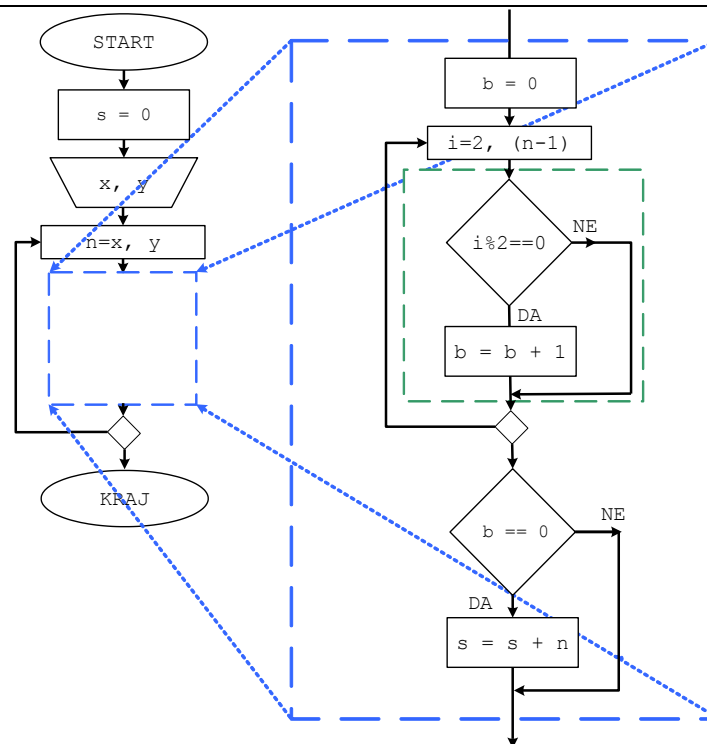
```

1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  void f1(int); // prototip (deklaracija) funkcije
6:  void f0(int); // prototip (deklaracija) funkcije
7:
8:  void main()
9:  {
10:     int m, n;
11:
12:     cout << "Unesi pocetak i kraj niza: \n";
13:     cin >> m >> n;
14:
15:     f0(n-m+1); //funkciji se prosljeđuje se samo jedan parametar
16:
17:     cout << "Kraj programa \n";
18: }
19:
20: void f0(int ux)
21: {
22:     cout << "\n----Ulazak u funkciju f0---- \n";
23:
24:     for (int i=1; i<=ux; i++)
25:     { // ostatak funkcije je isti kao u prvom načinu

```



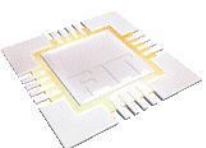
Rješenje zadatka br. 101:



Rješenje zadatka br. 102:

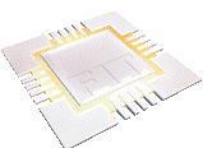
```

1:  #include <iostream>
2:  using namespace std;
3:
4:  void ispisi_ako_je_prost(int); // prototip
5:
6:  void main()
7:  {
8:      int x, y;
9:
10:     cout << "Unesite brojeve x i y \n";
11:     cin >> x >> y;
12:     cout << "=====\n";
13:
14:     for (int n=x; n<=y; n++)
15:     {
16:         ispisi_ako_je_prost(n);
17:     }
18: }
19:
20: void ispisi_ako_je_prost(int ux)
21: {
22:     int b = 0;
23:
24:     for (int i=2; i<=ux-1; i++)
25:     {
26:         if (ux % i == 0)
27:         {
28:             b = b + 1;
29:         }
30:     }
31:
32:     if (b == 0)
33:         cout << "Broj " << ux << " je prost. \n";
34: }
  
```



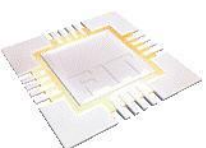
Rješenje zadatka br. **103**:

```
1:  #include <iostream>
2:  using namespace std;
3:
4:  void ispisi_ako_je_prost(int);
5:  void fx();
6:
7:  void main()
8:  {
9:      int n;
10:
11:      cout << "Koliko puta zelite ponoviti program: ";
12:      cin >> n;
13:
14:      for (int i=1; i<=n; i++)
15:      {
16:          cout << "\n\n===== "<< i << "=====\n\n";
17:          fx();
18:      }
19:  }
20:
21:  void fx()
22:  {
23:      int m, n;
24:      cout << "Unesi pocetak i kraj niza: ";
25:      cin >> m >> n;
26:
27:      for (int i=m; i<=n; i++)
28:      {
29:          ispisi_ako_je_prost(i);
30:      }
31:  }
32:
33:  void ispisi_ako_je_prost(int ux)
34:  {
35:      ...
```

Rješenje zadatka br. **104**:

```
1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  void main()
6:  {
7:      float a, b, c, d, e, y1, y2, y3, y4, y5, y6;
8:      cout << "Unesi broj a koji nije negativan: ";
9:      cin >> a;
10:
11:      cout << "Unesi broj b koji je pozitivan: ";
12:      cin >> b;
13:
14:      cout << "Unesi ugao c u radijanima (1 rad = 90 stepeni): ";
15:      cin >> c;
16:
17:      cout << "Unesi bazu d: ";
18:      cin >> d;
19:
20:      cout << "Unesi eksponent e: ";
21:      cin >> e;
22:
23:      if (a>=0)
24:      {
25:          y1 = sqrt(a);
26:          cout << "y1 = " << y1 << endl;
27:      }
28:
29:      if (b > 0)
30:      {
31:          y2 = log10(b);
32:          y3 = log(b);
33:          cout << "y2 = " << y2 << endl;
34:          cout << "y3 = " << y3 << endl;
35:      }
36:
37:
38:      y4 = cos(c);
39:      y5 = sin(c);
40:      cout << "y4 = " << y4 << endl;
41:      cout << "y5 = " << y5 << endl;
42:
43:      if (d!=0 || e!=0)
44:      {
45:          y6 = pow(d,e);
46:          cout << "y6 = " << y6 << endl;
47:      }
48:  }
```

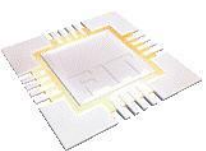
Rješenje zadatka br. **105**:




```
1:  #include <iostream>
2:  #include <math.h>
3:  using namespace std;
4:
5:  void f0();
6:  void f1(int);
7:  void f2(int, int);
8:  void f3(float);
9:
10: void main()
11: {
12:     double x = f1();           // f1 je tipa void
13:     f0();                     // OK
14:     f2(3,4);                  // OK
15:     f2(sqrt(9.0), sqrt(4.0)); // OK
16:     f2(pow(2.0), sqrt(4.0));  // u funkciji pow se nalazi samo jedan parametar
17:     f3(pow(2.0, 2.0));        // OK
18:     f1(f3());                 // f3 je tipa void
19:     pow(3,4);                 // pow nije tipa void
20:     x = sqrt(f1(), f3());      // f1 i f3 su tipa void
21:     cout << f3() << endl;     // f3 je tipa void
22:     cout << int (f3()) << endl; // f3 je tipa void
23:     char y = system("PAUSE"); // system je tipa void
24: }
25: ...
```

Rješenje zadatka br. 106 - prvi način – sa varijablom v:

```
1:  void main()
2:  {
3:      int a, b, c, V;  // varijabla V je sada lokalna
4:
5:      cout << "Unesi dimenzije u metrima:\n";
6:      cin >> a >> b >> c;
7:
8:      V = vol(a,b,c);
9:
10:     cout << "Zapremina u litrima iznosi: " << V << endl;
11:
12:     if (V > 1000)
13:         cout << "Bazen je velik! \n";
14: }
```



Rješenje zadatka br. **107** - drugi način – bez varijable v:

```
1: void main()
2: {
3:     int a, b, c;
4:
5:     cout << "Unesi dimenzije u metrima:\n";
6:     cin >> a >> b >> c;
7:
8:     cout << "Zapremina u litrima iznosi: " << vol(a,b,c) << endl;
9:
10:    if ( vol(a,b,c) > 1000 )
11:        cout << "Bazen je velik! \n";
12: }
```

Rješenje zadatka br. **108**:

```
... ...
44: float sr_vrij(float u1, float u2, float u3)
45: {
46:     float x;
47:     x = (u1 + u2 + u3)/3;
48:     return x;
49: }
50:
51: int suma_parnih_brojeva(int u1, int u2)
52: {
53:     int s = 0;
54:     for (int i=u1; i<=u2; i++)
55:     {
56:         if (i%2 == 0)
57:             s = s + i;
58:     }
59:     return s;
60: }
61: float max(float u1, float u2)
62: {
63:     if (u1 > u2)
64:         return u1; // mozemo koristiti i vise return naredbi
65:     else
66:         return u2;
67: }
68:
69: float hipotenuza(float uk1, float uk2)
70: {
71:     return sqrt(uk1*uk1 + uk2*uk2);
72: } // u naredbi return mogu se nalaziti matematicki izrazi
73:
74: float kateta(float u1, float u2)
75: {
76:     if (u1 > u2) // hipotenuza je veca od katete
77:         return sqrt(u1*u1 - u2*u2);
78:     else
79:         return sqrt(u2*u2 - u1*u1);
80: }
```

