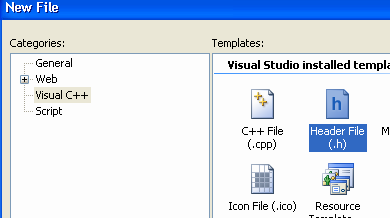
# Zadaci – 10b. dio, *Funkcije (treći dio)*

Zadatak 109:

Napravite sljedeće funkcije koje nisu void (napravite funkcije sa povratnom vrijednošću):

* **min**: čiji će izlaz biti manji broj od dva ulazna broja
* **max**: čiji će izlaz biti veći broj od dva ulazna broja
* **fakt**: čiji će izlaz biti izračunati faktorijel
* **prost**: koja će vraćati vrijednost 1 (true) ako je njezin parametar prost broj, ako nije prost onda vraća vrijednost 0 (false)
* **sumakvadrata**: koja vraća sumu kvadrata od u1 do u2

Zadatak riješite tako što ćete svaku funkciju pojedinačno implementirati, a zatim je testirati.

Testiranu funkciju snimite u jedan fajl (prototip i definiciju funkcije). Najjednostavnije vam je da u *Visual Studio*-u otvorite novi fajl (ne projekat) na sljedeći način:

*File* –> *New File* i odaberite *Header File*, i u njega prebacite (*cut - paste*) gotovu funkciju.

Premještanje implementacija funkcija možete činiti, takođe, pomoću nekog *editora*. (*Notepad* i sl.)

Svaku gotovu funkciju testirajte pojedinačno - jednim jednostavnim pozivom kao npr.:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8: | void main()  {  int a, b;  a = prost(17);  b = prost(15);  cout << a << endl;  cout << b << endl;  } |

Ako ste ispravno implementirali funkciju prost, u liniji 4 bi se trebala ispisati vrijednost 1 (jer je 17 prost broj) a u liniji 5 vrijednost 0 (jer 15 nije prost broj).

Primjer za testiranje funkcije **max** i **min**:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5: | void main()  {  float x = max(2.5, 2.6);  cout << x << endl; // ispis: 2.6  } |

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5: | void main()  {  float x = min(2.5, 2.6);  cout << x << endl; // ispis: 2.5  } |

Primjer za testiranje funkcije **fakt**:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5: | void main()  {  int x;  x = fakt(5);  cout << x << endl; |

Ovaj program bi trebao ispisati vrijednost 120 - znamo da je:



*Pomoć za funkciju fakt:*

|  |
| --- |
| *Pomoć 1:*  Ovaj program je dosta sličan programu koji računa sumu brojeva od 1 do n  *Pomoć 2:*  U *for*-petlji za računanje sume morali ste ponavljati naredbu s=s+i, dok u petlji za računanje faktorijela morate ponavljati naredbu f=f\*i.  Razmislite o slijedećem važnom pitanju za ovaj zadatka: Koja je početna vrijednost varijable f?  *Pomoć 3:*  Početna vrijednost varijable f nije nula, jer proizvod broja 0 i bilo kog drugog broja predstavlja opet nulu. |

Primjer za testiranje funkcije **sumakvadrata**:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6: | void main()  {  int x;  x = sumakvadrata(2,6);  cout << x << endl;  } |

Ovaj program bi trebao ispisati vrijednost 90 - znamo da je:

22 + 32 + 42 + 52 + 62 = 4 + 9 + 16 + 25 + 36 = 90

Nakon što ste iskopirali ispravno testirane definicije i prototipove funkcija u novi dokument, snimite fajl sa opcijom *Save As* pod nazivom moje\_funkcije u neki folder, kao npr: c:\temp

Nakon što ste snimili fajl moje\_funkcije.h zaboravite kako funkcioniraju funkcije. Nemojte se time više zamarati! Od sada vam nije više važno kako funkcije rade, nego šta funkcije rade. Morate da znate imena funkcija i koliko parametara prosljeđujete.

Rješenje ovog zadatka, tj. sadržaj fajla moje\_funkcije.h (prototipovi i definicije funkcija) se nalazi na stranici 150.

Od sada ćemo koristiti funkcije, koje smo prethodno definisali, kao gotove funkcije.

Počnite novi program. Dodajte novu naredbu za uključivanje biblioteke moje\_funkcije.h:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11: | #include <iostream>  #include <math.h>  using namespace std;  #include <c:\temp\moje\_funkcije.h>  void main()  {  int x;  x = sumakvadrata(2,6);  cout << x << endl;  } |

Naredba za uključivanje biblioteke moje\_funkcije.h bi se trebala nalaziti ispod naredbe using namespace std, inače ne bismo mogli koristiti naredbe koje nudi biblioteka iostream (kao npr. cin, cout, system, …), jer u fajlu moje\_funkcije.h nismo uključili iostream i using namespace std. Mada je to nebitno u ovoj lekciji, a ipak je bolje da biblioteku moje\_funkcije.h uključimo kao zadnju.

U slučaju da smo u biblioteci moje\_funkcije.h koristili funkciju sqrt, naredba za uključivanje biblioteke math.h bi se, takođe, trebala nalaziti ispred naredbe za uključivanje moje\_funkcije.h. Prilikom pokretanja ovog programa linija br. 4 će se zamijeniti sa sadržajem fajla moje\_funkcije.h.

Ako ne želite svaki put unositi čitavu putanju (*path*) do fajla moje\_funkcije.h, onda snimite fajl moje\_funkcije.h u folder u kojem se nalaze biblioteke (header fajlovi) vašeg C++ kompajlera. Taj folder bi treba biti:

- za Visual Studio 6 :

'C:\Program Files\Microsoft Visual Studio\VC98\**Include'**

- za Visual Studio 2003 .NET :

'C:\Program Files\Microsoft Visual Studio .NET 2003\VC7\**include'**

- za Visual Studio 2005 :

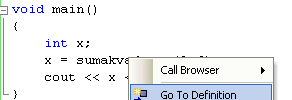
'C:\Program Files\Microsoft Visual Studio 8\VC\**include'**

- za Visual Studio 2008 :

'C:\Program Files\Microsoft Visual Studio 9\VC\**include'**

- za Borland C++ 4.5 (napomena: ime fajla može biti maksimalne dužine do 8 karaktera):

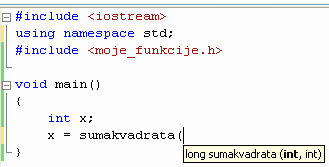
'C:\BC45\INCLUDE'

Header fajl možete otvoriti sa desnim klikom na ime fajla (u naredbi #include) i odabirom opcije *Open dokument* ili sa desnim klikom na ime neke funkcije i odabirom opcije *Go To Definition* ili *Go To Declaration*.

Ako ste snimili header fajl u folder Include onda uključivanje tog header fajla možete učiniti i bez navođenja pune putanje:

#include <moje\_funkcije.h>

Naredbu #include <math.h> možete prebaciti u fajl moje\_funkcije.h, tako da možete koristit matematičke funkcije (u fajlu moje\_funkcije.h i u glavnom programu) bez dodavanja te naredbe u glavnom programu.

Isprobajte sljedeće:

Upišite ime neke funkcije i otvorite zagradu. Prikazat će vam se prototip funkcije, tako da možete vidjeti koliko i koje vrste parametara od vas očekuje funkcija. Pogledajte sliku.

Zadatak 110:

Napravite program koji će u funkciji main zahtijevati unos dvije vrijednosti m i n od korisnika. Funkcija main treba da proslijedi parametre m i n novoj funkciji nad koja treba izračunati vrijednost (čita se: *m* nad *n*) i vratiti tu vrijednost. U funkciji main pridružite izlaznu vrijednost ispisu cout. Koristite svoju gotovu funkciju fakt.

Formula koju biste trebali znati: .

*Riješen C++ kôd se nalaze na stranici 151.*

Nakon testiranja funkcije nad, dodajte prototip i definiciju u *header* fajl moje\_funkcije.h.

Pomoću ove funkcije možete, takođe, računati broj kombinacija **n**-te klase od **m** elemenata (ili obrnuto).

Zadatak 111:

Program iz prethodnog zadatka br. 131 je malo nepouzdan. Da bi program ispravno radio, korisnik mora da unese drugi broj koji je veći od prvog, inače bi program neispravno radio. Ovo bi mogli poboljšati kada bi vršili provjeru prilikom poziva funkcije nad. U slučaju da je prvi broj manji od drugog pozvali bi funkciju nad sa parametrima n i m, umjesto m i n.

Umjesto *if-else*-iskaza za provjeru, možemo koristiti naše gotove funkcije min i max:

|  |  |
| --- | --- |
| 17:  18:  19:  20:  21:  22:  23: | ...  int veci, manji;  veci = max(m,n);  manji = min(m,n);  cout << nad(veci, manji) << endl;  } |

Da ne bi svaki put kad želimo koristiti funkciju nad morali dodavati funkcije max i min, bolje bi bilo da te funkcije dodamo u definiciju funkcije nad:

…Smjestite sljedeće naredbe (ako su ispravne) negdje u funkciji nad i isprobajte funkciju.

|  |
| --- |
| m = max(m, n);  n = min(m, n); |

*Nakon što se dodali naredbe, pročitajte rješenje na stranici na kraju dokumenta.*

Zadatak 112:

Napravite program u kojem ćete u funkciji main zahtijevati od korisnika unos dvije vrijednosti, m za početak niza i n za kraj niza. Funkcija main treba proslijediti m i n novoj funkciji svi\_prosti koja će ispisati sve proste brojeve koji se nalaze u tom nizu. Koristite gotovu funkciju prost.

*Rješenje se nalaze na kraju dokumenta.*

*Pomoć:*

- u funkciji svi\_prosti dodajte *for*-petlju sa varijablom i koja će ponavljati naredbe:

* vrijednost (izlaz) funkcije prost za parametar i pridružite varijabli x (koja je tipa *int* ili *bool*)
* ako je vrijednost x različita od nule (tj. ako je broj i prost) ispišite broj i na ekran

Pokušajte riješiti ovaj program i bez dodatne varijable x.

Zadatak 113:

#### Napravite funkciju brojac\_prostih sa dva ulazna parametra, čija će povratna vrijednost biti vrijednost koja predstavlja broj prostih brojeva od u1 do u2, tj. izbrojane proste brojeve u rasponu od u1 do u2.

#### Napravite test program za funkciju.

*Rješenje se nalaze na kraju dokumenta.*

#### Pomoć:

#### - prototip funkcije glasi: int brojac\_prostih(int, int);

**Zadatak 114:**

#### a) Napravite funkciju suma\_faktorijela sa dva ulazna parametra, čija će povratna vrijednost biti suma faktorijela za brojeve iz raspona od u1 do u2.

#### Napravite test program za funkciju.

*Rješenje se nalaze na kraju dokumenta.*

b) Nakon što ste to odradili, prepravite funkciju sumafakt tako da ta funkcija može računati sumu ako korisnik unese prvi broj veći od drugog, a u slučaju da korisnik unese, kao početak ili kraj niza, negativan broj funkcija sumafakt treba vratiti vrijednost **0**. Koristite vaše gotove funkcije min i max.

*Rješenje se nalaze na kraju dokumenta.*

**Zadatak 115:**

Napravite program koji će za pet unesenih brojeva ispisati najmanji, koristeći samo gotovu funkciju min bez upotrebe *if-(else)-*iskaza.

*Rješenje se nalaze na kraju dokumenta.*

Funkcije: call by reference

Naredni zadatak može poslužiti kao uvod u ovu lekciju.

Zadatak 116:

Napravite dijagram toka, i na osnovu njega napišite C++ kôd za sljedeći zadatak:

Napravite funkciju void ispisi\_rjesenja(float, float, float) koja na osnovu tri ulazna parametra a, b, c treba ispisati na ekran rješenja (*x1* i *x2*) kvadratne funkcije (*ax2+bx+c=0*). U funkciji main zahtijevati od korisnika unos članova kvadratne jednačine (a, b, c).

U slučaju da kvadratna jednačina postane linearna (za *a=0*), nemojte ispisati rješenja.

*Riješen C++ kôd (bez dijagrama toka) se nalaze na kraju dokumenta*

*Pomoć:*

* funkciji izračunajte vrijednost diskriminante *D = b2 - 4ac.*
* na osnovu vrijednosti *D* i vrijednosti *a* možete zaključiti da li kvadratna jednačina ima rješenja:

|  |  |  |
| --- | --- | --- |
|  | |  |
|  |  | Nema realnih rješenja |
| Rješenja kvadratne jednčine: | Postaje linearna jednačina:    ali u ovom slučaju ćemo ispisati da nema rješenja |

* uslov za računanje rješenja x1 i x2 glasi: (D>=0 && a!=0)
* testirajte program, unesite barem jednu negativnu vrijednost:
* za unesene vrijednosti **1**,**3**,**2** program treba ispisati:

x1 = 2

x2 = 1

* za unesene vrijednosti 0,8,2 i 1,2,3 program treba ispisati:

Greska: Nije moguce izracuanti rjesenja

Zadatak 117:

Kad bi ste željeli napraviti funkciju sa povratnom vrijednošću za računanje rješenja kvadratne jednačine (samo računanje a ne ispisivanje) možda bi ste imali jedan problem o implementacije. Problem je u tome što kvadratna jednačina ima dva rješenja. Funkcije u C++ mogu imati samo jednu povratnu vrijednost.

U tom slučaju mogli bi ste napraviti dvije slične funkcije, jednu koja računa rješenje kvadratne jednačine ***x1*** a drugu za ***x2***.

Napravite program za računanje rješenja kvadratne jednačine, tako da koristite funkcije fx1 i fx2 (funkcija fx1 računa ***x1***, a funkcija fx2 računa ***x2***).

Funkcije pozovite na sljedeći način:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16: | void main()  {  float **a**m, **b**m, **c**m;  cout << "Unesite clanove a, b, c " << endl;  cin >> **a**m;  cin >> **b**m;  cin >> **c**m;  float **x1**m, **x2**m;  **x1**m = fx1(**a**m, **b**m, **c**m);  cout << "Rjesenje kvadratne jednacine x1: " << **x1**m << endl;    **x2**m = fx2(**a**m, **b**m, **c**m);  cout << "Rjesenje kvadratne jednacine x2: " << **x2**m << endl;  } |

Ove funkcije ne bi trebale ispisivati rješenja na ekran, nego rješenja trebaju biti izlaz iz funkcije, a funkcija main bi trebala ispisivati rješenja kvadratne jednačine ***x1*** i ***x2***, tj. varijable x1m i x2m. Ovdje nam se javlja još jedan matematički problem: Šta ako jednačina nema rješenja? (npr. za vrijednosti 0,8,2 **i** 1,2,3)

Kako će nam funkcija koja ne ispisuje ništa, koja samo vraća gotovu vrijednost (broj), javiti da ne postoji rješenje. Taj problem ne možemo lahko riješiti pa ćemo se dogovoriti, u slučaju da ne postoje rješenja, da funkcija ispiše poruku o grešci i vrati vrijednost 0.

*Rješenje se nalaze na kraju dokumenta.*

U ovom primjeru smo koristili dvije funkcije sa po jednom izlaznom (povratnom) vrijednošću. Dvije izlazne vrijednosti iz jedne funkcije bi mogli postići pomoću globalnih varijabli. U našem slučaju bi koristili dvije ***globalne*** varijable x1 i x2 i jednu funkciju tipa ***void*** koja bi pomoću primljenih parametara a, b, c odnosno **a**m, **b**m, **c**m izračunala (izmijenila) globalne varijable x1 i x2. To možete uraditi samostalno za vježbu.

Kako smo već dosta puta rekli upotreba globalnih varijabli nije preporučljiva, stoga ćemo ovaj problem riješiti pomoću varijabli koje će biti ***referenca***. Deklarisat ćemo dodatne dvije lokalne varijable x1m i x2m, koje možemo proslijediti kao aktuelne parametre funkciji fx zajedno sa varijablama **a**m, **b**m, **c**m.

Funkcija fx imati pet ulaznih parametara. Prva tri parametra, koja predstavljaju članove kvadratne jednačine, će biti tipa *float*, dok ćemo parametre x1m i x2m (iz funkcije main) primati u funkciji fx pod imenom x1 i x2. Ta dva parametra će biti tipa ***referenca***na *float*.

*Šta, inače, predstavlja referenca?* Do sada smo parametre koristili za prenos vrijednosti u funkcije (*pass-by-value*). To znači, da se pri pozivu neke funkcije, vrijednosti aktuelnih parametara (**a**m, **b**m, **c**m) **kopiraju** u formalne parametre (varijable a, b, c). U ovom načinu poziva funkcije (*pass-by-value*) bilo koja promjena varijabli a, b, c u pozvanoj funkciji fx neće uticati na varijable **a**m, **b**m, **c**m funkcije main.

U prenosu parametara po referenci (*pass-by-reference*), vrijednosti aktuelnih parametara (**a**m, **b**m, **c**m) se **ne** **kopiraju** u formalne parametre (a, b, c), već se formalnim parametrima (varijable a, b, c) dodjeljuje zajednički memorijski prostor u RAM-u kao aktuelnim parametrima (varijablama **a**m, **b**m, **c**m). Pošto varijable x1 i x1m, odnosno x2 i x2m dijele zajednički memorijski prostor u RAM-u za 'pamćenje' njihovih vrijednosti, onda bilo koja promjena vrijednosti varijable x1 u funkciji fx će uticati na promjenu vrijednost varijable x1m u funkciji main. Isto vrijedi i za par x2 i x2m.

Aktuelni parametri, koji se prosljeđuju kao referenca, moraju biti varijable, ne smiju konstante, niti brojevi (npr. 7...).

Kod poziva funkcije po referenci, aktuelni parametri i formalni parametri moraju biti varijable istog tipa (npr. ako je x1m tipa *float*, onda i x1 mora biti *float*, a ne smije biti *long int*, *int* niti *double*).

## Deklaracija formalnih parametara kao referenca

Formalne parametre (x1 i x2) ćemo deklarirati kao referencu dodavanjem znaka ampersand '**&**' iza tipa varijable. Slijedi primjer deklaracije reference u funkciji fx.

|  |  |
| --- | --- |
| 1:  2:  3: | // 1-ulaz 2-ulaz 3-ulaz 4-izlaz 5-izlaz  void **fx**(float a, float b, float c, float **&**x1, float **&**x2)  {  ... |

referentni parametri se nazivaju i **izlazni** parametri

**Zadatak 118:**

Prepravite prethodni program koristeći funkciju fx:

|  |
| --- |
| void fx(float, float, float, float &, float &); // 🡸 ovo je samo prototip |

*Rješenje se nalaze na kraju dokumenta.*

Zadatak 119:

Napravite prototip i definiciju funkcije uvecaj\_r i testirajte je u sljedećem programu:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10: | void main()  {  int x;  x = 5;  uvecaj\_r(x); // vrijednost varijable x će se povećati sa 5 na 6  uvecaj\_r(x); // vrijednost varijable x će se povećati sa 6 na 7  cout << x << endl; // ispisat će se vrijednost 7  } |

*Rješenje se nalaze na kraju dokumenta.*

Zadatak 120:

Napravite program koji će u funkciji main zahtijevati od korisnika unos dva broja. Napravite funkciju racunaj koja će primiti 5 parametara. Prva dva parametra će biti ulazne vrijednost (pass-by-value), a ostala 3 će biti reference tipa *float* (pass-by-reference) – izlazna vrijednost. Funkcija racunaj treba izvršiti matematičke operacije (sabiranje, oduzimanje, množenje) i rezultat tih operacija treba smjestiti u zadnja tri parametra funkcije (*pass-by-reference*).

*Rješenje se nalaze na kraju dokumenta.*

Zadatak 121:

Napravite funkcije **ff** (za ***f****loat*) i **fi** (za ***i****nteger*), koje će ulaznom parametru ukloniti predznak.

Napravite test program za funkcije.

*Rješenje se nalaze na kraju dokumenta.*

*Pomoć:*

Cilj ovih funkcija je da aktualni parametar nakon poziva uvijek bude pozitivan broj.

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10: | void main()  {  int x;  cout << "Unesi broj: ";  cin >> x;  fi(x); // uklanja predznak varijable x  cout << x << endl; // vrijednost x ce uvijek biti pozitivna  } |

Rješenja

|  |  |
| --- | --- |
| Rješenje zadatka br. **109**: | |
| 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:  37:  38:  39:  40:  41:  42:  43:  44:  45:  46:  47:  48:  49:  50:  51:  52:  53:  54:  55:  56:  57:  58:  59:  60 | //moje\_funkcije.h  double min(double, double);  double max(double, double);  long int fakt(int);  int prost(int);  long int sumakvadrata(int, int);  double min(double a, double b)  {  if (a<b)  return a;  else  return b;  }  double max(double a, double b)  {  if (a>b)  return a;  else  return b;  }  long int fakt(int n) // u Visual Studio-u je nebitno da li koristite *long int*  { // ili *int*, jer je njihova velicina u oba slucaja po 4 bajta  long int f;  f=1; // pocetna vrijednost je 1 (zbog mnozenja)  for (int i=1; i<=n; i++)  f=f\*i;  return f;  }  bool prost(int n) // U Borland C++u nepostoji tip podatka *bool*, pa cete morati  { // koristiti povratni tip *int*  int brojac;  brojac=0;    for (int i=2; i<=n/2; i++)  {  if (n % i == 0)  brojac++;  }  if (brojac==0)  return 1; // ili return true;  else  return 0; // ili return false;  }  long int sumakvadrata(int m, int n)  {  long int suma;  suma=0;  for (int i=m; i<=n; i++)  suma = suma + i\*i;  return suma;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **110**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19: | #include <iostream>  using namespace std;  #include <moje\_funkcije.h>  int nad(int m, int n)  {  int izlaz;  izlaz = fakt(m) / **(** fakt(n)\*fakt(m-n) **)**;  return izlaz;  }  void main()  {  int m, n;  cout << "Unesi dva broja (drugi ne smije biti veci od prvog) \n";  cin >> m >> n;  cout << nad(m,n) << endl;  } |

Iz ovog se vidi, da funkcija **nad** poziva funkciju **fakt** tri puta za tri različita parametra. Ovdje smo, takođe, koristili iste nazive varijabli (**m**, **n**) u funkciji **main** i u funkciju **nad** (možda i u funkciji **fakt**). Nazivi varijabli se smiju ponavljati u različitim funkcijama.

Testirajte program sa vrijednostima *6* i *2*: 

|  |
| --- |
| Rješenje zadatka br. **111**: |

Ako ste dodati naredbe …

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5: | int nad(int m, int n)  {  m = max(m, n);  n = min(m, n);  ... |

...onda vam funkcija nad neće raditi ispravno. Kad bi vrijedilo da je *n>m*, u liniji br. 3 bi se varijabli m dodijelila vrijednost n, jer je *n>m*. Obje varijable bi imale istu vrijednost (*m=n*). Tako bi i nakon linije br. 4 varijabla n imala istu vrijednost kao m.

Slijedi ispravan primjer:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6: | int nad(int u1, int u2)  {  int m,n;  m = max(u1, u2);  n = min(u1, u2);  ... |

|  |  |  |
| --- | --- | --- |
| Rješenje zadatka br. **112** *(bez dodatne varijable x)*: | | |
| 1:  2:  3:  4:  5:  6:  7:  8: | void svi\_prosti(int u1, int u2)  {  for (int i=u1; i<=u2; i++)  {  if (prost(i))  cout << i << endl;  }  } | void svi\_prosti(int u1, int u2)  {  for (int i=u1; i<=u2; i++)  {  if (prost(i) != 0)  cout << i << endl;  }  } |
| 9:  10:  11:  12:  13:  14:  15:  16:  17:  18: | void main()  {  int m, n;  cout << "Unesi pocetak i kraj niza: \n";  cin >> m >> n;  svi\_prosti(m,n);  } | |

|  |  |
| --- | --- |
| Rješenje zadatka br. **113**: | |
| 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: | #include <iostream>  using namespace std;  #include <moje\_funkcije.h>  int brojac\_prostih(int u1, int u2)  {  int brojac;  brojac=0;  for (int i=u1; i<=u2; i++)  {  if (prost(i))  { // ispis je moguc i u funkcijama koje  //cout << i << endl; // imaju povratnu vrijednost    brojac++;  }  }  return brojac;  }  void main()  {  int m, n;  cout << "Unesi pocetak i kraj niza: \n";  cin >> m >> n;    int ukupno;  ukupno = brojac\_prostih(m, n); // funkcija ima povratnu vrijednost // povratna vrijednost se pridruzuje varijabli ukupno  cout << "Ukupno: "<< ukupno << endl;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **114** *(a)*: | |
| 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: | #include <iostream>  using namespace std;  #include <moje\_funkcije.h>  int suma\_faktorijela (int u1, int u2) // u Borlandu obavezno koristite *long int*  { // kao povratnu vrijednost jer se ovdje  int s = 0; // radi o velikim vrijednostima    for (int i=u1; i<=u2; i++)  {  s = s + fakt(i);  }  return s;  }  void main()  {  int m, n;  cout << "Unesi pocetak i kraj niza: \n";  cin >> m >> n;    int suma;  suma = suma\_faktorijela(m, n);  cout << "Suma: "<< suma << endl;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **114** *(b)*: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15: | int suma\_faktorijela (int u1, int u2)  {  if (u1 == 0 || u2 == 0)  return 0; // izlaz iz funkcije, ostatak naredbi se nece izvrsiti  int s = 0, m, n;  m = min(u1, u2);  n = max(u1, u2);    for (int i=m; i<=n; i++)  {  s = s + fakt(i);  }  return s;  } |

Samo da znate, i ovo je moguće:

|  |  |
| --- | --- |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13: | int suma\_faktorijela (int u1, int u2)  {  if (u1 == 0 || u2 == 0)  return 0;  int s = 0;    for (int i=min(u1, u2); i<=max(u1, u2); i++)  {  s = s + fakt(i);  }  return s;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **115**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18: | #include <iostream>  using namespace std;  #include <moje\_funkcije.h>  void main()  {  int a1, a2, a3, a4, a5;  cout << "Unesi 5 brojeva \n";  cin >> a1 >> a2 >> a3 >> a4 >> a5;  cout << "====\n";  cout << min( min**(** min(a1,a2), min(a3,a4)**)**, a5) << endl; // ili  cout << min(min(min(min(a4, a5), a3), a2), a1) << endl; // ili  cout << min(a1, min(a2, min(a3, min(a4, a5)))) << endl;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **116**: | |
| 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: | #include <iostream>  #include <math.h>  using namespace std;  void ispisi\_rjesenja(float, float, float); // prototip  void ispisi\_rjesenja(float a, float b, float c)  {  float D = b\*b - 4\*a\*c;  if (D>=0 && a!=0)  {  float x1, x2;  x1 = (b+sqrt(D)) / (2\*a); // nemojte zaboraviti dodatne zagrade zbog nazivnika x2 = (b-sqrt(D)) / (2\*a);    cout << " x1 = " << x1 << endl;  cout << " x2 = " << x2 << endl;  }  else  cout << "Greska: Nije moguce izracuanti rjesenja \n";  }  void main()  {  float **a**m, **b**m, **c**m; // ovdje su koristeni drugaciji nazivi varijabli nego u fx  // zbog lakseg objasnjenja koji slijedi poslije  cout << "Unesite clanove a, b, c " << endl;  cin >> **a**m; // sufix **m** može asocirati na funkciju **m**ain  cin >> **b**m;  cin >> **c**m;  ispisi\_rjesenja(**a**m, **b**m, **c**m);  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **117**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18:  19:  20:  21: | float fx1(float a, float b, float c)  {  float D = b\*b - 4\*a\*c;  float izlaz;  if (D>=0 && a!=0)  {  izlaz = (b+sqrt(D)) / (2\*a);  }  else  {  cout << "Greska: Nije moguce izracuanti rjesenja \n";  izlaz = 0;  }  return izlaz;  }  float fx2(float a, float b, float c)  {  ... |

|  |  |
| --- | --- |
| Rješenje zadatka br. **118**: | |
| 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: | #include <iostream>  #include <math.h>  using namespace std;  void fx(float a, float b, float c, float &x1, float &x2)  {  float D = b\*b - 4\*a\*c;  if (D>=0 && a!=0)  {  x1 = (b+sqrt(D)) / (2\*a);  x2 = (b-sqrt(D)) / (2\*a);  }  else  {  cout << "Greska: Nije moguce izracuanti rjesenja \n";  x1 = 0;  x2 = 0;  }  }  void main()  {  float am, bm, cm;  cout << "Unesite clanove a, b, c " << endl;  cin >> am;  cin >> bm;  cin >> cm;  float x1m, x2m;  fx(am, bm, cm, **x1m**, **x2m**); //funkcija je tipa void    cout << "Rjesenje kvadratne jednacine x1: " << **x1m** << endl;  cout << "Rjesenje kvadratne jednacine x2: " << **x2m** << endl;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **119**: | |
| 1:  2:  3:  4:  5:  6:  7:  8:  9:  10:  11:  12:  13:  14:  15:  16:  17:  18: | #include <iostream>  using namespace std;  **void** uvecaj\_r(int &a)  {  a = a + 1;  }  void main()  {  int x;  x = 5;  uvecaj\_r(x);  uvecaj\_r(x);  cout << x << endl;  } |

Pazite funkcija uvecaj\_r je tipa *void* pa se ne smije pridružiti ispisu naredbi cout:

|  |
| --- |
| cout << uvecaj\_**r**(x); **//GREŠKA** |

...

|  |  |
| --- | --- |
| Rješenje zadatka br. **120**: | |
| 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;  void racunaj(float a, float b, float &r1, float &r2, float &r3)  {  r1 = a+b;  r2 = a-b;  r3 = a\*b;  }  void main()  {  float x, y, mm1, mm2, mm3;    cout << "Unesi dva broja \n";  cin >> x >> y;  racunaj(x,y, mm1, mm2, mm3);  cout << "=============================== \n";  cout << "Rezultat sabiranja: " << mm1 << endl;  cout << "Rezultat oduzimanja: " << mm2 << endl;  cout << "Rezultat mnozenja: " << mm3 << endl;  } |

|  |  |
| --- | --- |
| Rješenje zadatka br. **121**: | |
| 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 fi(int &r)  {  if (r<0) // ako je r negativno  r=-r; // neka promjeni predznak  }  void fd(float &r)  {  if (r<0)  r=-r;  }  void main()  {  float d;  int i;  cout << "Unesite cijeli broj: ";  cin >> i;  fi(i);  cout << i << endl << endl;  cout << "Unesite decimalni broj: ";  cin >> d;  fd(d);  cout << d << endl;  } |