Природно-математички факултет
Универзитет у Бањој Луци

Функције и рекурзија – Група 2 средње школе

Аутори: Драган Матић, Милана Грбић,
Милан Предојевић, Ненад Вилендечић

2025

Садржај

Увод

На другом предавању Пољетне школе програмирања за ученике средњих школа обрађена је тема функција и рекурзије.

У наставку се налазе задаци који су рађени током предавања, са циљем да ученици примијене и утврде стечено знање кроз практичне примјере. На крају документа налазе се задаци за домаћи рад, који служе за додатно вјежбање и самостално истраживање обрађених концепата.

Задаци

1. Задатак: Факторијел броја

Написати рекурзивну функцију и итеративну функцију, које рачунају факторијел ненегативног броја. Функције тестирати у главном дијелу програма на броју који се уноси са тастатуре.

Примјер извршавања

```
Unesite nenegativni broj: 12
Rezultat faktorijela (rekurzivno): 479001600
Rezultat faktorijela (iterativno): 479001600
```

Рјешење

```cpp
#include <iostream>

using namespace std;

// Rekurzivna funkcija za racunanje faktorijela
long long faktorijel_rekurzivno(int n) {
    if (n == 0 || n == 1) {
        return 1;
    }
    return n * faktorijel_rekurzivno(n - 1);
}

// Iterativna funkcija za racunanje faktorijela
long long faktorijel_iterativno(int n) {
    long long rezultat = 1;
    for (int i = 1; i <= n; i++) {
        rezultat *= i;
    }
    return rezultat;
}

int main() {
    int broj;
    cout << "Unesite nenegativni broj: ";
    cin >> broj;

    long long rezultat_rekurzivno = faktorijel_rekurzivno(broj);

    long long rezultat_iterativno = faktorijel_iterativno(broj);

    cout << "Rezultat faktorijela (rekurzivno): " <<
   rezultat_rekurzivno << endl;
    cout << "Rezultat faktorijela (iterativno): " <<
   rezultat_iterativno << endl;

    return 0;
}
```

2. Задатак: n-ти Фибоначијев број

Написати рекурзивну функцију и итеративну функцију које рачунају n-ти Фибоначијев број. Функције тестирати у главном дијелу програма на броју n који се уноси са тастатуре.

Примјер извршавања

```
Unesite broj n: 5
Fibonacci broj (rekurzivno) za n = 5 je: 8
Fibonacci broj (iterativno) za n = 5 je: 8
```

Рјешење

```cpp
#include <iostream>
using namespace std;

// Rekurzivna funkcija za racunanje n-tog Fibonacci broja
long long fib_rekurzivno(int n) {
    if (n == 0 || n == 1)
    return 1;
    else
    return fib_rekurzivno(n - 1) + fib_rekurzivno(n - 2);

}

// Iterativna funkcija za racunanje n-tog Fibonacci broja
long long fib_iterativno(int n) {
    if (n == 0 || n == 1)
    return 1;

    long long prvi = 1, drugi = 1, novi;
    for (int i = 2; i <= n; i++) {
        novi = prvi + drugi;
        prvi = drugi;
        drugi = novi;
    }
    return drugi;
}

int main() {
    int n;
    cout << "Unesite broj n: ";
    cin >> n;

    // Testiranje rekurzivne funkcije
    long long rezultat_rekurzivno = fib_rekurzivno(n);
    cout << "Fibonacci broj (rekurzivno) za n = " << n << " je: "
   << rezultat_rekurzivno << endl;

    // Testiranje iterativne funkcije
```

```
37      long long rezultat_iterativno = fib_iterativno(n);
38      cout << "Fibonacci broj (iterativno) za n = " << n << " je: "
    << rezultat_iterativno << endl;
39
40      return 0;
41  }
```

3. Задатак: Бројеви у релацији

За два цијела броја кажемо да су у релацији ако се састоје од истих цифара, нпр. бројеви 1223 и 321123 су у релацији. Написати функцију која за аргумент узима два цијела броја, а као резултат враћа 1, ако су бројеви у релацији, док у супротном враћа 0. Функцију тестирати у главном дијелу програма на бројевима који се уносе са тастатуре.

Примјер извршавања

```
Unesite prvi broj: 1223
Unesite drugi broj: 321123
Brojevi su u relaciji.
```

Примјер извршавања

```
Unesite prvi broj: 14551
Unesite drugi broj: 1155
Brojevi nisu u relaciji.
```

Рјешење

```cpp
#include <iostream>
using namespace std;

//funkcija koja provjerava da li se sve cifre broja1 nalaze u broju2
bool sveCifre (int broj1, int broj2){

    while (broj1 > 0) {
        bool found = false;  // Fleg koja oznacava da li je cifra pronadjena u drugom broju
        int cifra1 = broj1 % 10;  // Uzimamo poslednju cifru broja1
        broj1 /= 10;

        int temp_2 = broj2;
        while (temp_2 > 0) {
            int cifra2 = temp_2 % 10;  // Uzimamo poslednju cifru broja2
            if (cifra1 == cifra2) {
                found = true;  // Ako cifru pronadjemo u broju 2
                temp_2 /= 10;  // Uklanjamo tu cifru iz broja 2
                break;  // Prekidamo unutrasnju petlju
            }
            temp_2 /= 10;  // Uklanjamo poslednju cifru iz broja 2
        }

        if (!found) {
            return false;  // Ako cifra iz broja 1 nije pronadjena u broju 2, vracamo false
```

```
25          }
26      }
27      return true;
28
29  }
30
31  bool uRelaciji(int broj1, int broj2) {
32      return sveCifre(broj1,broj2) && sveCifre(broj2,broj1);
33  }
34
35  int main() {
36      int broj1, broj2;
37
38      // Unos brojeva
39      cout << "Unesite prvi broj: ";
40      cin >> broj1;
41      cout << "Unesite drugi broj: ";
42      cin >> broj2;
43
44
45      // Testiranje funkcije
46      if (uRelaciji(broj1, broj2)) {
47          cout << "Brojevi su u relaciji." << endl;
48      } else {
49          cout << "Brojevi nisu u relaciji." << endl;
50      }
51
52      return 0;
53  }
```

4. Задатак: Најмањи број k

Нека је број n_1 производ цифара датог броја n, број n_2 производ цифара броја n_1,..., број n_k производ цифара броја n_{k-1}, при чему је k најмањи природан број за који је n_k једноцифрен. Написати функцију која за дато n израчунава k. На примјер, вриједности ове функције за 10, 25, 39 су редом 1, 2, 3.

Примјер извршавања

```
Unesite broj n: 39
k je: 3
```

Рјешење

```cpp
#include <iostream>
using namespace std;

// Funkcija koja racuna proizvod cifara broja
int proizvodCifara(int broj) {

    int proizvod = 1;
    while (broj > 0) {
        proizvod *= broj % 10;
        broj /= 10;
    }
    return proizvod;
}

// Funkcija koja racuna k prema zadatom broju n
int izracunajK(int n) {

    int k = 0;
    while (n >= 10) {  // Ako broj ima vise od jedne cifre
        n = proizvodCifara(n);  // Izracunaj proizvod cifara
        k++;  // Povecaj k za svaki korak
    }
    return k;
}

int main() {
    int n;

    cout << "Unesite broj n: ";
    cin >> n;

    int k = izracunajK(n);

    cout << "k je: " << k << endl;

    return 0;
}
```

5. Задатак: Занимљиви бројеви

Са тастатуре се уносе два броја. Потребно је пронаћи укупан број занимљивих бројева између та два унесена броја. Број је занимљив ако су све цифре, не рачунајући водеће нуле, исте осим једне цифре која је различита. На примјер, бројеви 3323 и 110 су занимљиви, док бројеви 9779 и 55555 нису.

Примјер извршавања

```
Unesite pocetni broj: 100
Unesite krajnji broj: 150
Zanimljivi brojevi izmedju 100 i 150
100
101
110
112
113
114
115
116
117
118
119
121
122
131
133
141
144
Broj zanimljivih brojeva je: 17
```

Рјешење

```cpp

#include <iostream>
using namespace std;


bool jeZanimljiv(int broj) {

    int prvaRazlicita = broj % 10;  // Uzmi poslednju cifru
    int brojacPR = 1;
    broj /= 10;  // Ukloni poslednju cifru

    bool imaRazlicitih = false;
    int drugaRazlicita = -1;
    int brojacDR = 0;

    while (broj > 0) {
```

```
17          int trenutnaCifra = broj % 10;  // Uzmi trenutnu cifru
18
19          if(!imaRazlicitih && trenutnaCifra == prvaRazlicita){
20              brojacPR++;
21              broj/=10;
22              continue;
23          }
24
25          if(!imaRazlicitih && trenutnaCifra != prvaRazlicita){
26              drugaRazlicita = trenutnaCifra;
27              brojacDR++;
28              imaRazlicitih = true;
29              broj/=10;
30              continue;
31          }
32
33          if(imaRazlicitih && trenutnaCifra != prvaRazlicita &&
    trenutnaCifra != drugaRazlicita){
34              return false;
35          }
36
37
38          if(imaRazlicitih && trenutnaCifra == prvaRazlicita &&
    trenutnaCifra != drugaRazlicita){
39              brojacPR++;
40              broj/=10;
41              continue;
42          }
43
44          if(imaRazlicitih && trenutnaCifra != prvaRazlicita &&
    trenutnaCifra == drugaRazlicita){
45              brojacDR++;
46              broj/=10;
47              continue;
48          }
49
50      }
51
52      // Broj treba imati bas jednu razlicitu cifru
53
54      if(brojacPR >1 && brojacDR == 1)
55      return true;
56      else if(brojacPR == 1 && brojacDR>1)
57      return true;
58      else
59      return false;
60  }
61
62  int main() {
63      int pocetniBroj, krajnjiBroj;
64      int brojZanimljivih = 0;
```

```
65
66      // Unos pocetnog i krajnjeg broja
67      cout << "Unesite pocetni broj: ";
68      cin >> pocetniBroj;
69      cout << "Unesite krajnji broj: ";
70      cin >> krajnjiBroj;
71
72
73
74      // Prolazimo kroz sve brojeve izmedju pocetnog i krajnjeg
    broja
75      cout<<"Zanimljivi brojevi izmedju "<<pocetniBroj<<" i
    "<<krajnjiBroj<<endl;
76      for (int i = pocetniBroj; i <= krajnjiBroj; i++) {
77          if (jeZanimljiv(i)) {
78              cout<<i<<endl;
79              brojZanimljivih++;  // Povecaj broj zanimljivih
    brojeva
80          }
81      }
82
83      // Ispis rezultata
84      cout << "Broj zanimljivih brojeva je: " << brojZanimljivih <<
    endl;
85
86
87      return 0;
88  }
```

6. Задатак: Велики бинарни бројеви

Драган је данас у школи учио бинарни бројни систем – систем бројева са само двије различите цифре (0 и 1). Претварање бројева у овај систем Драган је са лакоћом савладао па му је учитељица поставила тежи задатак. На табли је написала један број N у декадном систему и рекла Драгану да испише све бројеве од 1 до N у бинарном облику, један за другим (на примјер за N=3 то су бројеви 1, 10, 11). Затим је рекла да споји све ове бројеве у један велики бинарни број (11011 за N=3). Задатак који је поставила Драгану је да преброји укупан број јединица и нула у овом великом бинарном броју (за постављени претходни примјер број јединица је 4, а број нула је 1).

Примјер извршавања

```
Unesite broj N: 3
Broj jedinica: 4
Broj nula: 1
```

Рјешење

```

#include <iostream>
using namespace std;

// Funkcija koja broji broj jedinica u binarnom broju
int brojiJedinice(int broj) {
    int brojJedinica = 0;

    // Brojanje jedinica u binarnom broju
    while (broj > 0) {
        if (broj % 2 == 1) {
            brojJedinica++;
        }
        broj /= 2;  // Skidamo posljednju cifru
    }

    return brojJedinica;
}

// Funkcija koja broji broj nula u binarnom broju
int brojiNule(int broj) {
    int brojNula = 0;

    // Brojanje nula u binarnom broju
    while (broj > 0) {
        if (broj % 2 == 0) {
            brojNula++;
        }
        broj /= 2;  // Skidamo posljednju cifru
    }

```

```
    return brojNula;
}

int main() {
    int N;
    int ukupnoJedinica = 0;
    int ukupnoNula = 0;

    // Unos broja N
    cout << "Unesite broj N: ";
    cin >> N;

    // Prolazimo kroz sve brojeve od 1 do N i brojimo jedinice i
   nule
    for (int i = 1; i <= N; i++) {
        ukupnoJedinica += brojiJedinice(i);
        ukupnoNula += brojiNule(i);
    }

    // Ispis rezultata
    cout << "Broj jedinica: " << ukupnoJedinica << endl;
    cout << "Broj nula: " << ukupnoNula << endl;

    return 0;
}
```

7. Задатак: Ана и троуглови

Мала Ана је дјевојчица која обожава математику. Данас су на часу математике учили о троугловима. Учитељица је разреду објашњавала шта је троугао, као и које врсте троуглова постоје и све је то вјешто скицирала на табли. Ана се највише заинтересовала за правоугли троугао. Научила је тог дана у школи да је то троугао у којем један угао износи тачно 90 и назива се прави угао. Страница наспрам правог угла назива се хипотенуза, а двије преостале странице су катете. Ђаци су такође научили да се збир дужина свих страница троугла назива обим троугла. Учитељица је на крају часа оставила ђацима један занимљив задатак који гласи овако: За задати обим троугла O, израчунати колико постоји различитих правоуглих троуглова који имају тај обим. Два троугла се сматрају идентичним ако су им и мање и веће катете једнаке. На примјер, два правоугла троугла $(a = 3, b = 4, c = 5)$ и $(a = 4, b = 3, c = 5)$ се сматрају истим и броје се као један. Са тастатуре се уноси цијели број $(1 \leqslant O \leqslant 106)$ који представља обим троугла. На излазу исписати цијели број који представља број различитих правоуглих троуглова са обимом .

Примјер извршавања

```
Unesite obim: 60
Pravougli trougao: a = 10, b = 24, c = 26
Pravougli trougao: a = 15, b = 20, c = 25
Broj razlicitih pravouglih trouglova sa obimom 60 je: 2
```

Рјешење 1

```

#include <iostream>
#include <cmath>
using namespace std;

// Funkcija koja provjerava da li a i b mogu biti katete
   pravouglog trougla sa datim obimom O
bool daLiSuKatetePT(int a, int b, int O) {
    // Izracunavamo hipotenuzu c
    int c = O - a - b;
    // Ako je c manji ili jednak 0, onda nije validan trougao
    if (c <= 0) return false;

    // Provjera da li a, b i c mogu biti katete i hipotenuza
   pravouglog trougla
    return (a * a + b * b == c * c);
}

int main() {
    int O;
    int brojT = 0;

    // Unos obima
    cout << "Unesite obim: ";
```

```
    cin >> O;

    // Prolazimo kroz sve moguce vrijednosti kateta
    for (int a = 1; a < O / 2; a++) {
        for (int b = a; b < O / 2; b++) {
            // Pozivamo funkciju da provjeri da li a i b mogu biti katete pravouglog trougla
            if (daLiSuKateretePT(a, b, O)) {
                brojT++;
                cout << "Pravougli trougao: a = " << a << ", b = " << b << ", c = " << O - a - b << endl;
            }
        }
    }

    // Ispisujemo ukupan broj trouglova
    cout << "Broj razlicitih pravouglih trouglova sa obimom " << O << " je: " << brojT << endl;

    return 0;
}
```

Pješење 2

```

#include <iostream>
#include <cmath>
using namespace std;


int main() {
    int O;
    int brojT = 0;

    // Unos obima
    cout << "Unesite obim: ";
    cin >> O;

    // Prolazimo kroz sve moguce vrijednosti katete a
    for (int a = 1; a < O/2; a++) {
        int zbirBC = O - a;
        int razlikaBC = (a*a)/zbirBC;
        int c = (zbirBC + razlikaBC)/2;
        int b = zbirBC - c;
        if(a<b && a+b+c == O && a*a+b*b == c*c){
            cout<<"**********"<<endl;
            cout<<a<<endl;
            cout<<b<<endl;
            cout<<c<<endl;
            cout<<"**********"<<endl;
```

```
27                brojT++;
28            }
29
30        }
31
32        // Ispisujemo ukupan broj trouglova
33        cout << "Broj razlicitih pravouglih trouglova sa obimom " <<
      O << " je: " << brojT << endl;
34
35        return 0;
36    }
```

Домаћи рад

1. Задатак: Степен броја

Написати рекурзивну функцију и итеративну функцију које рачунају степен броја. Функције тестирати у главном дијелу програма на бројевима који се уносе са тастатуре.

2. Задатак: Линеарна једначина

Напиши програм који одређује број рјешења линеарне једначине $a * x + b = 0$ за реалне вриједности коефицијената a и b и ријешава је ако постоји јединствено рјешење. Улаз: У првом реду коефицијент a, у другом коефицијент b. Излаз: Ако има једно рјешење испис рјешења на двије децимале, ако нема рјешења порука: NEMA RJESENJA, ако има бесконачно много рјешења порука: BESKONACNO RJESENJA.

3. Задатак

За сваки од задатака провјерити да ли постоји ефикасније рјешење и ако постоји написати га. Напомена: Задатак 6 (велики бинарни бројеви) урадити помоћу битског оператора шифт >>.