

INFORMATIČKI KLUB  
**FUTURA**

```
#include<stdio.h> C  
int main()  
{ RADIONICE PROGRAMIRANJA  
  printf("Hello World!");  
  ZA SREDNJE ŠKOLE  
  return 0;  
}
```

# **RADIONICE PROGRAMIRANJA ZA SREDNJE ŠKOLE - 3. RADIONICA**

Nikola Rabrenović, Informatički klub FUTURA  
Dubrovnik, 29. studenog 2014.



# Creative Commons



## slobodno smijete:

- **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
- **remiksirati** — prerađivati djelo



## pod slijedećim uvjetima:

- **imenovanje**. Morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
- **nekomercijalno**. Ovo djelo ne smijete koristiti u komercijalne svrhe.
- **dijeli pod istim uvjetima**. Ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.



U slučaju daljnjeg korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela. Najbolji način da to učinite je linkom na ovu internetsku stranicu.

Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.

Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.

Tekst licence preuzet je s <http://creativecommons.org/>.

# Sadržaj

---

- Funkcije
- Deklaracija i definicija funkcije
- Pozivanje
- Glavna funkcija programa

# Funkcije

---

- ❑ Zasebni odsječci programa koji imaju svoju funkcionalnost.
- ❑ Doprinose modularnosti programa razdvajajući programski kod u logičke cjeline.
- ❑ Omogućuju ponovnu iskoristivost programskog koda.
- ❑ Olakšavaju preglednost programskog koda.

# Funkcije

---



```
main();
```

```
printf("Dobar dan");
```

```
scanf("%d", &a);
```

```
srtcpy(ime, niz);
```

```
sqrt(a);
```

```
pow(a, 2);
```

# Funkcije – definicija funkcije



**Tip funkcije** – određuje kojeg je tipa povratna vrijednost funkcije.

**Ime funkcije** – identifikator kojeg koristimo kod pozivanja funkcije.

**Lista argumenata funkcije** – varijable koje sadrže vrijednosti koje se predaju funkciji kod njenog pozivanja.

**Formalni argumenti.**

Sintaksa zapisa argumenata je: **tip naziv**

Više argumenata u funkciji odvaja se zarezom:

**int funkcija(float arg1, char arg2, float arg3);**

Deklaracija lokalnih varijabli. Lokalne varijable vidljive su samo unutar tijela funkcije u kojoj su deklarirane.

Definicija funkcije predstavlja cijeli programski kod kojim je opisana njezina funkcionalnost.

```
double potenciraj(int broj, int potencija) {  
    int i;  
    double umnozak=1;  
  
    for(i=1; i<=potencija; i++) {  
        umnozak = umnozak * broj;  
    }  
  
    return umnozak;  
}
```

Tijelo funkcije

**return** – naredba za povratak vrijednosti funkcije i prekid njenog izvršavanja.

# Funkcije – definicija funkcije



**Void** – funkcija tipa **void** ne vraća vrijednost.

**Lista argumenata funkcije** – funkcija ne mora primiti argumente ako nije potrebno.

**void** pozdrav() {

printf("Dobar dan.\n");

}

Tijelo funkcije

**return** – primjećujemo da funkcija ne vraća vrijednost. Nema naredbe **return**.

# Funkcije – definicija funkcije



```
double potenciraj(int broj, int potencija) {
    int i;
    double umnozak=1;

    if(potencija==0) {
        return 1;
    }
    else if(potencija>0) {
        for(i=1; i<=potencija; i++) {
            umnozak = umnozak * broj;
        }
    }
    return umnozak;
}
```

```
void pozdrav() {
    printf("Dobar dan.\n");
}
```



# Funkcije – deklaracija funkcije



```
double potenciraj(int broj, int potencija);
```

- ❑ Funkcija mora biti deklarirana prije nego se prvi puta pozove.
- ❑ Definicija funkcije je ujedno i deklaracija funkcije.
- ❑ Ako se funkcija definira prije `main()` funkcije, nije dozvoljeno deklarirati funkciju.

```
#include <stdio.h>
```

Prototip funkcije

```
int potenciraj(int, int);
```

```
double potenciraj(int broj, int potencija);
```

```
int main() {  
    int a, n; double rez;  
    printf("Izracun potencije broja a na n.\n");  
    printf("Unesite a n: ");  
    scanf("%d %d", &a, &n);  
}
```



# Funkcije – deklaracija funkcije



```
#include <stdio.h>
```

```
double potenciraj(int broj, int potencija) {  
    int i; double umnozak=1;  
  
    if(potencija==0) {  
        return 1;  
    }  
    else if(potencija>0) {  
        for(i=1; i<=potencija; i++) {  
            umnozak = umnozak * broj;  
        }  
    }  
    return umnozak;  
}
```

Definicija funkcije u ovom primjeru služi kao i deklaracija jer je definirana prije glavne funkcije.

```
int main() {  
    int a, n;  
    double rez;  
    printf("Izracun potencije broja a na n.\n");  
    printf("Unesite a n: ");  
    scanf("%d %d", &a, &n);
```

# Funkcije – poziv funkcije

---



- ❑ Funkcija se poziva navođenjem njezinog imena i predavanjem potrebnih argumenata.
- ❑ Formalni argumenti su argumenti u definiciji funkcije i funkcija upravlja njihovim vrijednostima unutar tijela funkcije.
- ❑ Kod poziva funkcije formalni argumenti funkcije se inicijaliziraju sa stvarnim argumentima funkcije.
- ❑ Imena formalnih i stvarnih argumenata ne moraju biti isti.
- ❑ Broj argumenata prilikom pozivanja funkcije mora biti jednak broju argumenata u definiciji funkcije.
- ❑ Tipovi stvarnih argumenata koji se predaju funkciji moraju biti istoga tipa kao i formalni argumenti u definiciji funkcije.

# Funkcije – poziv funkcije (primjer)



Napisati funkciju za zbrajanje dva cijela broja.

```
#include <stdio.h>
#include <stdlib.h>

int zbroji(int x, int y) {
    int suma = 0;
    suma = x + y;
    return suma;
}

int main() {
    int a, b, c;
    printf("Unesite dva broja [a b]: ");
    scanf("%d %d", &a, &b);

    c = zbroji(a, b);

    printf("%d + %d = %d\n", a, b, c);
    system("pause");
    return 0;
}
```

**Poziv funkcije** – navodimo ime funkcije i navodimo listu stvarnih argumenata.

Glavni program

```
a=5
b=4
poziv funkcije

c <-- 9
5 + 4 = 9
kraj programa
```

Funkcija

```
x=5
y=4
suma=0
suma=5+4 -> 9
return 9
```

Kod poziva funkcije izvršavanje glavnog programa se nakratko zaustavlja i prepušta izvođenju funkcije.

U posebnom dijelu memorije pohranjuju se sve vrijednosti operacija funkcije. Kada se funkcija završi ona vraća povratnu vrijednost i prepušta izvođenje glavnom programu.

# Funkcije – poziv funkcije (primjer)



Funkcija za zbrajanje dvaju brojeva. Glavni program poziva funkciju.

```
#include <stdio.h>
#include <stdlib.h>

int zbroji(int x, int y) {
    int suma = 0;
    suma = x + y;
    return suma;
}

int main() {
    int a, b, c;
    printf("Unesite dva broja [a b]: ");
    scanf("%d %d", &a, &b);

    c = zbroji(a, b);

    printf("%d + %d = %d\n", a, b, c);
    system("pause");
    return 0;
}
```

```
Unesite dva broja [a b]: 5 4
5 + 4 = 9
Press any key to continue . . . _
```

# 1. zadatak: Jednostavni kalkulator



Na temelju prethodnog primjera napravite jednostavan kalkulator operacija zbrajanja, oduzimanja, množenja i dijeljenja.

Program treba prikazati izbornik s računskim operacijama. Nakon izbora računske operacije u program treba upisati dva broja.

Pozivom funkcija odradite računsku operaciju.

Rezultat ispišite iz glavnog programa.

Z - zbrajanje  
O - oduzimanje  
M - množenje  
D - dijeljenje

ULAZ: z  
ULAZ: 5 4  
IZLAZ: 5 + 4 = 9

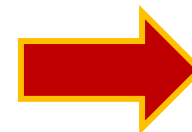
Z - zbrajanje  
O - oduzimanje  
M - množenje  
D - dijeljenje

ULAZ: o  
ULAZ: 10 4  
IZLAZ: 10 - 4 = 6

ULAZ: d  
ULAZ: 5 2  
IZLAZ: 5 / 2 = **2.50**

Pazite na tip funkcije za dijeljenje.

**Vrijeme**



# Funkcije – glavna funkcija main()



Osnovna funkcija s kojom počinje izvršavanje programa.

**Tip funkcije** – uvijek mora biti tipa `int` jer vraća cjelobrojnu vrijednost operacijskom sustavu.

**Ime funkcije** – `main()` označava glavnu funkciju koja se pokreće pozivanjem programa. Može biti samo jedna `main()` funkcija.

```
int main() {  
    ...  
    return 0;  
}
```

A red arrow points from the `int` to the pink box. A green arrow points from the `main()` to the green box. A purple arrow points from the `return 0;` to the purple box.

```
void main() {  
    ...  
}
```

**return** – povratna vrijednost se proslijeđuje operacijskom sustavu prilikom završetka rada programa. Može označavati uredan kraj programa, a može predstavljati kôd greške.

## 2. zadatak: Igraonica



Vlasnik igraonice svake subote u podne organizira nagradnu igru u kojoj prva osoba čije ime počinje sa slučajno izabranim slovom dobiva besplatno igranje ostatak dana. Ako postoji više osoba čije ime počinje zadanim slovom, biraju se sve osobe. Pomozite vlasniku i napišite program koji će izabrati prav(u/e) osob(u/e).

### ULAZNI PODACI:

- jedno slovo **S** [A-Z]
- prirodan broj **N** ( $0 < N < 101$ ), broj osoba u igraonici;
- u sljedećih **N** redaka se nalaze imena osoba, svako maksimalne duljine 20 znakova.

### IZLAZNI PODACI:

- imena izabranih osoba ili tekst „\*\*\*\*\*”

ULAZ	IZLAZ
P	Pero
3	
Ivo	
Pero	
Ana	

ULAZ	IZLAZ
Z	*****
3	
Ana	
Bero	
Katica	

ULAZ	IZLAZ
V	Vedran
4	Vinko
Ana	
Bero	
Vedran	
Vinko	

**Vrijeme**





# 3. zadatak: Zbroj intervala



Napišite program za računanje zbroja brojeva u zadanom intervalu. Zbrajanje brojeva u zadanom intervalu napravite u obliku funkcije, a unos brojeva i ispis rezultata napravite u glavnom programu.

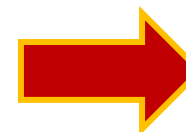
INTERVAL: [5 8]  
ALGORITAM:  $5+6+7+8 = 26$   
PRAVILO:  $A \leq B$

ULAZ:      IZLAZ:  
2 8        35

ULAZ:      IZLAZ:  
8 8        8

ULAZ:      IZLAZ:  
9 3        Greška!

**Vrijeme**



# 4. zadatak: Prosti (prim) broj



Napišite program koji provjerava je li uneseni broj prost (prim) ili ne. Ispitivanje napravite u obliku funkcije, a unos broja i ispis rezultata napravite u glavnom programu.

Prosti (prim) broj je prirodan broj veći od 1, a koji nije dijeljiv (bez ostatka) s nekim drugim prirodnim brojem osim sa samim sobom i brojem 1.

Primjeri prostih brojeva: 2, 3, 5, 7...

Modulo, mod, % je operacija koja za rezultat ima ostatak cjelobrojnog dijeljena.

17:3=5  
2



17 % 3 = 2  
rez = broj % 3;

7:2=3  
1



7 % 2 = 1  
rez = broj % 2;

ULAZ:        IZLAZ:  
47            Broj je prost.

ULAZ:        IZLAZ:  
26            Broj nije prost.

Dijeljenjem nekog broja s brojem dva bez ostatka možemo testirati je li izabrani broj paran ili ne 😊

**Vrijeme**

