***Napomene:***
- *Savjetuje se navedene zadatke riješiti ubrzo nakon predavanja*
- *Savjetuje se ne gledati rješenja prije nego se pokuša samostalno riješiti zadatke*

# 16. vježbe uz predavanja

U svim zadacima u kojima se traži definiranje funkcije, treba napisati odgovarajući glavni program (tj. funkciju `main`) u kojem ćete po potrebi definirati stvarne argumente, s tipkovnice učitati njihove vrijednosti, pozvati funkciju i ispisati rezultat.

1. Napisati funkciju koja za zadani cijeli broj n vraća $n^2$, ali tako da rezultat vraća preko adrese koju je dobila kao argument. Funkcija **ne smije** promijeniti stvarni argument n definiran u pozivajućem programu. Kojeg je tipa funkcija?
2. Napisati funkciju koja sadržaj neke cjelobrojne varijable n iz pozivajućeg programa **mijenja** u $n^2$. Dakle, funkcija treba **promijeniti** vrijednost neke cjelobrojne varijable koja je definirana u pozivajućem programu. Kojeg je tipa funkcija?
3. Napišite funkciju tipa double koja za zadanu vrijednost tipa double vraća zadanu vrijednost (tipa double) uvećanu za 10.0. Hoćete li dobiti ispravan rezultat ako funkciju pozovete sa stvarnim argumentom tipa int?
4. Napišite funkciju koja zadanoj varijabli tipa double vrijednost uvećava za 10.0. Dakle, funkcija treba **promijeniti** vrijednost neke realne (double) varijable koja je definirana u pozivajućem programu. Hoćete li dobiti ispravan rezultat ako funkciju pozovete sa stvarnim argumentom koji je pokazivač na varijablu tipa int?
5. Napišite funkciju koja za dvije zadane vrijednosti tipa int u pozivajući program vraća dvije vrijednosti: prva vraćena vrijednost je veća među zadanim vrijednostima, a druga vraćena vrijednost je manja među zadanim vrijednostima. Npr. ako se funkciji zadaju vrijednosti 2 i 3*2, funkcija u pozivajući program mora vratiti vrijednosti 6 i 2.
6. Napišite funkciju koja vrijednosti u zadanim **varijablama** x, y i z (tipa double) poredava po veličini, od najveće prema najmanjoj. Drugim riječima, očekuje se da će funkcija zamijeniti vrijednosti u varijablama x, y i z tako da vrijednosti budu poredane od najveće prema najmanjoj. Npr. ako se funkcija pozove za varijable x=2.0, y=4.0, z=3.0, nakon izvršavanja funkcije u varijablama x, y, z se moraju nalaziti vrijednosti x=4.0, y=3.0, z=2.0.
7. Napišite funkciju koja prima **pokazivače** na dvije varijable tipa int, te vraća **pokazivač** na onu od njih koja ima veću vrijednost. Ako varijable imaju istu vrijednost, funkcija vraća pokazivač na prvu varijablu.
8. Napišite funkciju koja prima **pokazivače** na dvije varijable tipa int, te vraća **vrijednost** varijable koja ima veću vrijednost.

# Rješenja

## Rješenje 1. zadatka

```
#include <stdio.h>

void kvad2(int n, int *rez) {
   *rez = n*n;
}

int main (void) {
   int n, n2;
   printf ("Upisite n: ");
   scanf("%d", &n);
   kvad2(n, &n2);
   printf("n na kvadrat (preko adrese) je: %d\n", n2);
   printf("Vrijednost varijable n se nije promijenila: %d\n", n);
   return 0;
}
```

Funkcija `kvad2` kao drugi argument dobija **adresu** varijable u koju će zapisati rezultat. Jedina naredba u toj funkciji upravo to i radi: na adresu kamo pokazuje pokazivač `rez`, zapisuje $n^2$. Primijetite da pozivajući program za drugi stvarni argument predaje **adresu** varijable `n2`. Tip funkcije je `void`, jer funkcija pomoću naredbe `return` ne treba vratiti niti jednu vrijednost.

Ipak, uočite da će uvjet iz programa (da funkcija ne smije promijeniti stvarni argument) biti narušen ukoliko se funkcija pozove na sljedeći način: `kvad2(n, &n);`

## Rješenje 2. zadatka

```
#include <stdio.h>

void kvad3(int *n) {
   *n = *n * *n;
}

int main (void) {
   int n;
   printf ("Upisite n: ");
   scanf("%d", &n);
   kvad3(&n);
   printf("n na kvadrat (promjena originalne varijable preko adrese) je: %d\n",
n);
   return 0;
}
```

Funkcija `kvad3` dobija **adresu** varijable u kojoj se nalazi cijeli broj čiji kvadrat treba izračunati, ali se na tu istu adresu također zapisuje i rezultat. Varijabla n iz pozivajućeg programa će biti promijenjena!

## Rješenje 3. zadatka

```
#include <stdio.h>

double uvecajZa10(double x) {
   return x + 10.;
}

int main (void) {
   double arg, rez;
   printf("Upisite realni broj: ");
   scanf("%lf", &arg);
   rez = uvecajZa10(arg);
   printf("%f uvecan za 10.0 jest %f\n", arg, rez);
   return 0;
}
```

Sada treba testirati što će se dogoditi ako se funkcija pozove s cjelobrojnim argumentom?

```
int main (void) {
   int arg, rez;
   printf("Upisite cijeli broj: ");
   scanf("%d", &arg);
   rez = uvecajZa10(arg);
   printf("%d uvecan za 10.0 jest %d\n", arg, rez);
   return 0;
}
```

Ako se funkcija pozove s cjelobrojnim stvarnim argumentom, dobit će se ispravan rezultat jer se pri prijenosu stvarnog u formalni argument obavlja implicitna konverzija (int→double), a pri pridruživanju rezultata funkcije varijabli `rez` obavlja se implicitna konverzija (double→int).

## Rješenje 4. zadatka

```
#include <stdio.h>

void uvecajZa10(double *x) {
   *x = *x + 10.;
}

int main (void) {
   double arg;
   printf("Upisite realni broj: ");
   scanf("%lf", &arg);
   uvecajZa10(&arg);
   printf("Uvecana varijabla jest %f\n", arg);
   return 0;
}
```

Sada treba testirati što će se dogoditi ako se funkciji umjesto pokazivača na varijablu tipa `double` preda pokazivač na varijablu tipa `int`?

```
int main (void) {
   int arg;
   printf("Upisite cijeli broj: ");
   scanf("%d", &arg);
   uvecajZa10(&arg);
   printf("Uvecana varijabla jest %d\n", arg);
   return 0;
}
```

Prevodilac će dojaviti upozorenje, ali će program ipak uspjeti prevesti (ako se radi s gcc prevodiocem, da bi prevođenje uspjelo u ovom primjeru, treba ispustiti opciju -pedantic-errors) .

**Rezultat izvršavanja programa neće biti ispravan.** To se moglo očekivati: funkcija je dobila adresu `int` varijable (pokazuje na neko područje u memoriji od 4 bajta), a "misli" da je dobila adresu `double` varijable (adresu koja pokazuje na područje memorije veličine 8 bajta).

Prekršili smo pravilo koje smo definirali na predavanjima: pokazivač na objekte tipa x smije se koristiti isključivo za pohranu adresa objekata tipa x.

Za vježbu, provjerite kakve ćete rezultate dobiti ako u funkciji, umjesto tipa double koristite tip float.

## Rješenje 5. zadatka

```
#include <stdio.h>

void poredaj(int a, int b, int *veci, int *manji) {
   if (a > b) {
      *veci = a;
      *manji = b;
   }
   else {
      *veci = b;
      *manji = a;
   }
}

int main (void) {
   int veci, manji;
   poredaj(2, 3*2, &veci, &manji);
   printf("veci i manji su: %d %d\n", veci, manji);
   return 0;
}
```

## Rješenje 6. zadatka

```
#include <stdio.h>

void poredaj(double *x, double *y, double *z) {
   double pom;
   if (*x < *y) {
      pom = *x;
      *x = *y;
      *y = pom;
   }
   if (*x < *z) {
      pom = *x;
      *x = *z;
      *z = pom;
   }
   if (*y < *z) {
      pom = *y;
      *y = *z;
      *z = pom;
   }
}

int main (void) {
   double a = 1.0, b = 2.0, c = 3.0;
   poredaj(&a, &b, &c);
   printf("poredani su: %f %f %f\n", a, b, c);
   return 0;
}
```

## Rješenje 7. zadatka

```
#include <stdio.h>

int *vratiAdresuVeceg(int *x, int *y) {
   if (*x >= *y)
      return x;
   else
      return y;
}

int main (void) {
   int a = 5, b = 2;
   int *veci;
   veci = vratiAdresuVeceg(&a, &b);
   printf("veci od zadana dva broja je: %d\n", *veci);
   return 0;
}
```

## Rješenje 8. zadatka

```
#include <stdio.h>

int vratiVrijednostVeceg(int *x, int *y) {
   if (*x > *y)
      return *x;
   else
      return *y;
}

int main (void) {
   int a = 5, b = 2;
   int veci;
   veci = vratiVrijednostVeceg(&a, &b);
   printf("veci od zadana dva broja je: %d\n", veci);
   return 0;
}
```