

15. 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 **tipa int** koja za zadani cijeli broj n (formalni argument je tipa `int`) vraća n^2 .
2. Provjerite hoćete li dobiti ispravan rezultat kada pomoću funkcije iz 1. zadatka pokušate izračunati 50000^2 . Objasnite što se dogodilo.
3. Provjerite hoćete li dobiti ispravan rezultat kada pomoću funkcije iz 1. zadatka pokušate izračunati 2.0^2 i 3.5^2 . Objasnite što se dogodilo.
4. Napisati funkciju **tipa double** koja za zadani cijeli broj n (formalni argument je tipa `int`) vraća n^2 . Provjerite hoćete li dobiti ispravan rezultat kada s tom funkcijom pokušate izračunati 2^2 , 50000^2 .
5. Koji je tip funkcije i što vraća funkcija `f`:

```
f (void) {  
    ;  
    ;  
}
```

6. Napisati funkciju koja na ekran ispisuje sve pozitivne parne brojeve između 2 i zadanog cijelog broja n (u obliku 2 4 6 8 ...). Kojeg je ta funkcija tipa?
7. Napisati funkciju koja na ekran ispisuje tablicu množenja za zadanih m redaka i n stupaca. Za ispis brojeva koristite format `%5d`. Npr. ispis za tablicu množenja od 3 retka i 4 stupca je:

	1	2	3	4
1	1	2	3	4
2	2	4	6	8
3	3	6	9	12

8. Napisati funkciju **tipa double** naziva `nfakt` za računanje $n!$. Napisati funkciju tipa `double` naziva `mpovrh` za računanje m povrh n koja će za izračunavanje koristiti funkciju `nfakt`. U glavnom programu (`main` funkciji) učitavati s tipkovnice cijele brojeve m i n dok god su ispravno zadani, te izračunavati i ispisivati m povrh n . Prekinuti program kad se zadaju pogrešne vrijednosti za m i n .
9. Napisati funkciju koja na zaslon ispisuje prvih 20 Fibonaccijevih brojeva (svaki član u novi redak na zaslonu).
10. Napisati funkciju tipa `int` koja vraća broj bajtova koji se koriste za pohranu podatka tipa `int`.
Napomena: različiti prevodioci koriste različiti broj bajtova, te se funkcija koja koristi sljedeću `return` naredbu ne može smatrati ispravnom:

```
return 4;
```

11. 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?

12. 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?
13. 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?
14. 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?
15. 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.
16. 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 $x=4.0$, $y=3.0$, $z=2.0$.
17. 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.
18. Napišite funkciju koja prima **pokazivače** na dvije varijable tipa int, te vraća **vrijednost** varijable koja ima veću vrijednost.

Rješenja svih zadataka provjeriti prevođenjem i testiranjem vlastitih programa!

Rješenja:

Rješenje 1. zadatka

```
#include <stdio.h>

int kvadrat(int n) {
    int kv;
    kv = n*n;
    return kv;
}

int main() {
    int arg, rez;
    printf("Upisite cijeli broj: ");
    scanf("%d", &arg);
    rez = kvadrat(arg);
    printf("%d na kvadrat jest %d\n", arg, rez);
    return 0;
}
```

Rješenje 2. zadatka

Ukoliko korisnik unese 50000, u varijablu `kv` neće se pohraniti ispravan rezultat (2500000000 se ne može pohraniti u varijablu `kv` jer se radi o broju koji prelazi dopušteni raspon za tip `int`). Funkcija će vratiti broj `-1794967296`

Rješenje 3. zadatka

Koristi se ista funkcija, ali drugačiji glavni program, kojim se s tipkovnice učitava **realni** broj.

```
int main() {
    float arg;
    int rez;
    printf("Upisite realni broj: ");
    scanf("%f", &arg);
    rez = kvadrat(arg);
    printf("%f na kvadrat jest %d\n", arg, rez);

    return 0;
}
```

Ukoliko korisnik unese 2.0, prilikom prijenosa stvarnog argumenta u formalni, obaviti će se konverzija u cijeli broj 2. Funkcija će vratiti cijeli broj 4.

Ukoliko korisnik unese 3.5, prilikom prijenosa stvarnog argumenta u formalni, obaviti će se konverzija u cijeli broj 3. Funkcija će vratiti cijeli broj 9.

Rješenje 4. zadatka

```
#include <stdio.h>

double kvadrat(int n) {
    double kv;
    kv = (double)n*n;
    return kv;
}

int main() {
    int arg;
    double rez;
    printf("Upisite cijeli broj: ");
    scanf("%d", &arg);
    rez = kvadrat(arg);
    printf("%d na kvadrat jest %f\n", arg, rez);
    return 0;
}
```

Ovdje je eksplicitna konverzija u tip `double` stavljena radi toga da se množenje obavi u `double` domeni. Inače, opet bi se dogodilo da se pri računanju 50000^2 dobije negativan cijeli broj, koji bi se kod pridruživanja varijabli `kv` pretvorio u realni broj (ali prekasno, jer bi se kao rezultat dobio negativan realni broj). Testirajte: izbacite cast operator (`double`) iz funkcije `kvadrat`.

Rješenje 5. zadatka

Funkcija je tipa `int`, a rezultat funkcije je nedefiniran, odnosno vraća "smeće" (vrijednost koju nije moguće unaprijed odrediti).

Rješenje 6. zadatka

```
#include <stdio.h>

void ispisiParne (int n) {
    int i;
    for (i = 2; i <= n; i += 2)
        printf("%d ", i);
}

int main() {
    int arg;
    printf("Upisite cijeli broj: ");
    scanf("%d", &arg);
    ispisiParne(arg);
    return 0;
}
```

Rješenje 7. zadatka

```
#include <stdio.h>

void ispisiTablicuMnozenja (int redaka, int stupaca) {
    int i, j;
    /* ispisi prvi red: "zaglavlje" tablice */
    printf("      ");
    for (j = 1; j <= stupaca; j++)
        printf("%5d", j);
    printf("\n");

    /* ispisi tablicu */
    for (i = 1; i <= redaka; i++) {
        /* na pocetku svakog retka ispisi redni broj retka */
        printf("%5d", i);

        for (j = 1; j <= stupaca; j++)
            printf("%5d", i*j);
        /* na kraju svakog retka tablice, skoci u novi redak na zaslonu */
        printf("\n");
    }
}

int main() {
    int m, n;
    printf("Upisite broj redaka: ");
    scanf("%d", &m);
    printf("Upisite broj stupaca: ");
    scanf("%d", &n);
    printf("\nTABLICA MNOZENJA:\n");
    ispisiTablicuMnozenja(m, n);
    return 0;
}
```

Rješenje 8. zadatka

```
#include <stdio.h>

double nfakt (int n) {
    int i;
    double f;
    for (f = 1, i = 1; i <= n; i++) {
        f *= i;
    }
    return f;
}

double mpovrh (int m, int n) {
    return nfakt(m) / ( nfakt(n) * nfakt(m-n) );
}

int main() {
    int m, n, mn;
}
```

```
do {
    printf ("Upisite m i n: ");
    scanf("%d %d", &m, &n);
    if (m >= 0 && n >= 0 && m >= n) {
        mn = mpovrh(m, n);
        printf("%d povrh %d je: %d\n\n", m, n, mn);
    }
} while (m >= 0 && n >= 0 && m >= n);
return 0;
}
```

Rješenje 9. zadatka

```
#include <stdio.h>

void fibonacci (void) {
    int i, f0 = 1, f1 = 1, f = 1;
    for (i = 0; i < 20; i++) {
        if (i > 1) {
            f = f1 + f0;
            f0 = f1;
            f1 = f;
        }
        printf ("%d\n", f);
    }
}

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

Rješenje 10. zadatka

```
#include <stdio.h>

int kolikoInt(void) {
    return sizeof(int);
}

int main() {
    int brojBajtovaZaInt;
    brojBajtovaZaInt = kolikoInt();
    printf("Ovaj prevodilac za tip int koristi bajtova: %d\n", brojBajtovaZaInt);
    return 0;
}
```

Rješenje 11. zadatka

```
#include <stdio.h>

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

```
int main () {
    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 12. zadatka

```
#include <stdio.h>

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

int main () {
    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 13. zadatka

```
#include <stdio.h>

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

int main () {
    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 () {
    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;
}
```

Ukoliko 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 14. zadatka

```
#include <stdio.h>

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

int main () {
    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 () {
    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 ipak prevesti program. **Rezultat 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 15. 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 () {
    int veci, manji;
    poredaj(2, 3*2, &veci, &manji);
    printf("veci i manji su: %d %d\n", veci, manji);
    return 0;
}
```

Rješenje 16. 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 () {  
    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 17. zadatka

```
#include <stdio.h>  
  
int *vratiAdresuVeceg(int *x, int *y) {  
    if (*x >= *y)  
        return x;  
    else  
        return y;  
}  
  
int main () {  
    int a = 5, b = 2;  
    int *veci;  
    veci = vratiAdresuVeceg(&a, &b);  
    printf("veci od zadana dva broja je: %d\n", *veci);  
    return 0;  
}
```

Rješenje 18. zadatka

```
#include <stdio.h>  
  
int vratiVrijednostVeceg(int *x, int *y) {  
    if (*x > *y)  
        return *x;  
    else  
        return *y;  
}  
  
int main () {  
    int a = 5, b = 2;  
    int veci;  
    veci = vratiVrijednostVeceg(&a, &b);  
    printf("veci od zadana dva broja je: %d\n", veci);  
    return 0;  
}
```