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

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 zaslon 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 zaslon 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 mpovrhn 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 niza u novi redak na zaslonu).
10. Napisati funkciju koja na zaslon ispisuje prvih *n* (*n* se zadaje kao argument funkcije) Fibonaccijevih brojeva (svaki član niza u novi redak na zaslonu).
11. 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 naredbu return ne može smatrati ispravnom:

```
return 4;
```

Rješenja

Rješenje 1. zadatka

```
#include <stdio.h>

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

int main(void) {
   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(void) {
   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, obavit će se konverzija u cijeli broj 2. Funkcija će vratiti cijeli broj 4.

Ukoliko korisnik unese 3.5, prilikom prijenosa stvarnog argumenta u formalni, obavit 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(void) {
   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(void) {
   int arg;
   printf("Upisite cijeli broj: ");
   scanf("%d", &arg);
   ispisiParne(arg);
   return 0;
}
```

Rješenje 7. zadatka

```c
#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(void) {
   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 mpovrhn (int m, int n) {
   return nfakt(m) / ( nfakt(n) * nfakt(m-n) );
}

int main(void) {
   int m, n, mn;
   do {
      printf ("Upisite m i n: ");
      scanf("%d %d", &m, &n);
      if (m >= 0 && n >= 0 && m >= n) {
         mn = mpovrhn(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(void) {
   fibonacci();
   return 0;
}
```

Rješenje 9. zadatka - alternativno

U funkciji je definirano polje veličine 20 članova. Tako se moglo postupiti zato jer je zadano da treba ispisati točno 20 članova niza.

```
#include <stdio.h>

void fibonacci (void) {
   int i, fbroj[20];
   fbroj[0] = fbroj[1] = 1;

   for (i = 2; i < 20; ++i)
      fbroj[i] = fbroj[i-1] + fbroj[i-2];
   for (i = 0; i < 20; ++i)
      printf ("%d\n", fbroj[i]);
}

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

Rješenje 10. zadatka

```
#include <stdio.h>

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

int main(void) {
   fibonacci(30);
   return 0;
}
```

Uočiti: zadatak se ne može riješiti pomoću polja, kao prethodni zadatak, jer se ne zna unaprijed koliko bi polje u funkciji trebalo biti veliko. Argument *n* se ne može pri definiciji polja koristiti kao dimenzija polja jer dimenzija polja pri definiciji mora biti cjelobrojni **konstantni** izraz. Dakle, nije dopušteno sljedeće:

```
void fibonacci (int n) {
   int i, fbroj[n];
```

Rješenje 11. zadatka

```
#include <stdio.h>

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

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