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

17. vježbe uz predavanja

1. Napisati prototip (deklaraciju) za svaku funkciju koja se nalazi u rješenjima prethodnih vježbi.
2. U modulu **mat2.c** napisati funkcije čiji su prototipovi navedeni u nastavku

   ```
   int binCoeff(int m, int n); /* izracunava "m povrh n" */
   int factorial(int n);       /* izracunava n! */
   int iabsolute(int n);       /* izracunava apsolutnu vrijednost */
   float fabsolute(float x);   /* izracunava apsolutnu vrijednost */
   ```

   Prototipove navedenih funkcija smjestiti u datoteku s prototipovima **mat2.h**.

   Funkciju main ("glavni program") smjestiti u modul **glavni.c** U glavnom programu treba izračunati i na zaslon ispisati rezultate za:

   ```
   factorial(0)
   factorial(25)
   factorial(26)
   binCoeff(13, 3)
   binCoeff(4, 4)
   iabsolute(-5)
   iabsolute(0)
   iabsolute(-5.7f)
   fabsolute(-5)
   fabsolute(-5.7f)
   ```

   Testirati prevođenje na dva načina:

   - tako da se oba modula prevedu i povežu samo jednim pozivom prevodioca
   - tako da se zasebno prevede svaki modul, a zatim se dobiveni objektni kôd poveže u izvršni kôd

   - koje datoteke su stvorene za vrijeme prevođenja na prvi, odnosno drugi način?

3. Što će se ispisati tijekom izvođenja sljedećeg programa:

   ```
   #include <stdio.h>

   void fun(void) {
      int x = 5;
      printf("%d\n", x);
      x++;
   }

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

4. Što će se ispisati tijekom izvođenja sljedećeg programa:

```
#include <stdio.h>

void fun(void) {
   static int x = 5;
   printf("%d\n", x);
   x++;
}

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

5. Što će se ispisati tijekom izvođenja sljedećeg programa:

```
#include <stdio.h>

void fun(void) {
   static int x = 5;
   int y = 5;
   printf("%d %d\n", ++x, --y);
}

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

6. Što će se ispisati tijekom izvođenja sljedećeg programa:

```
#include <stdio.h>

int main (void) {
   static int i = 5;
   int prviPut = 1;
   labela:
   {
      static int i = 10;
      int j = 15;
      printf("%d %d\n", i, j);
      i++;
      j++;
   }
   i++;
   printf("%d\n", i);
   if (prviPut) {
      prviPut = 0;
      goto labela;
   }
   return 0;

}
```

7. Što će se ispisati tijekom izvođenja sljedećeg programa:

```
#include <stdio.h>

static int x = 25;

void fun1(void) {
   static int x = 5;
   printf("%d\n", ++x);
}

void fun2(void) {
   int x = 10;
   printf("%d\n", ++x);
}

void fun3(void) {
   printf("%d\n", ++x);
}

int main (void) {
   x++;
   {
      static int x = 15;
      {
         int x = 20;
         printf("%d\n", x++);
      }
      printf("%d\n", x++);
   }
   printf("%d\n", x++);
   fun1();
   fun2();
   fun3();

   fun1();
   fun2();
   fun3();
   return 0;
}
```

U zadacima u kojima se traži definiranje funkcije, treba napisati prototipove funkcija, te odgovarajući glavni program (tj. funkciju `main`) u kojem ćete po potrebi definirati stvarne argumente, pozvati funkciju i ispisati rezultat.

8. Napisati funkciju `zbroji` tipa int koja vraća zbroj dvaju zadanih cijelih brojeva i funkciju `mnozi` tipa int koja vraća umnožak dvaju zadanih cijelih brojeva. Svaka od funkcija, osim što izračunava rezultat i vraća ga u pozivajući program, na zaslon ispisuju koliko je puta bila pozvana. Npr. ako se u glavnom programu obave naredbe:

```
printf("2*2=%d\n", mnozi(2,2));
printf("2+3=%d\n", zbroji(2,3));
printf("4+2=%d\n", zbroji(4,2));
printf("2*5=%d\n", mnozi(2,5));
printf("2*3=%d\n", mnozi(2,3));
```

na zaslonu se treba ispisati:

```
Funkcija mnozi do sada je pozvana 1 puta      ← ispisano u funkciji mnozi
2*2=4                                         ← ispisano u glavnom programu
Funkcija zbroji do sada je pozvana 1 puta     ← ispisano u funkciji zbroji
2+3=5                                         ← ispisano u glavnom programu
Funkcija zbroji do sada je pozvana 2 puta     ← ispisano u funkciji zbroji
4+2=6                                         ← ispisano u glavnom programu
Funkcija mnozi do sada je pozvana 2 puta      ← ispisano u funkciji mnozi
2*5=10                                        ← ispisano u glavnom programu
Funkcija mnozi do sada je pozvana 3 puta      ← ispisano u funkciji mnozi
2*3=6                                         ← ispisano u glavnom programu
```

9. Slično kao prethodni zadatak, uz dodatak: svaka od funkcija mora ispisati ne samo koliko je puta bila pozvana ona sama, nego i koliko puta je bila pozvana bilo koja od funkcija `zbroji` i `mnozi`. Npr. ako se u glavnom programu obave naredbe:

```
printf("2*2=%d\n", mnozi(2,2));
printf("2+3=%d\n", zbroji(2,3));
printf("4+2=%d\n", zbroji(4,2));
printf("2*5=%d\n", mnozi(2,5));
printf("2*3=%d\n", mnozi(2,3));
```

na zaslonu se treba ispisati:

```
Funkcija mnozi do sada je pozvana 1 puta          ← ispisano u funkciji mnozi
Funkcije zbroji i mnozi do sada su pozvane 1 puta ← ispisano u funkciji mnozi
2*2=4                                             ← ispisano u glavnom programu
Funkcija zbroji do sada je pozvana 1 puta         ← ispisano u funkciji zbroji
Funkcije zbroji i mnozi do sada su pozvane 2 puta ← ispisano u funkciji zbroji
2+3=5                                             ← ispisano u glavnom programu
Funkcija zbroji do sada je pozvana 2 puta         ← ispisano u funkciji zbroji
Funkcije zbroji i mnozi do sada su pozvane 3 puta ← ispisano u funkciji zbroji
4+2=6                                             ← ispisano u glavnom programu
Funkcija mnozi do sada je pozvana 2 puta          ← ispisano u funkciji mnozi
Funkcije zbroji i mnozi do sada su pozvane 4 puta ← ispisano u funkciji mnozi
2*5=10                                            ← ispisano u glavnom programu
Funkcija mnozi do sada je pozvana 3 puta          ← ispisano u funkciji mnozi
Funkcije zbroji i mnozi do sada su pozvane 5 puta ← ispisano u funkciji mnozi
2*3=6                                             ← ispisano u glavnom programu
```

10. Što će se ispisati tijekom izvođenja sljedećeg programa:

datoteka s prototipovima `proto.h`

```
void fun1(void);
void fun2(void);
void fun3(void);
void fun4(void);
```

modul `glavni.c`

```
#include <stdio.h>
#include "proto.h"
extern int x;

int main(void) {
   int x = 30;
   x += 2;
   printf("%d\n", x);
   fun1();
   fun2();
   fun3();
   fun4();
   fun3();
   return 0;
}

void fun1(void) {
   x += 3;
   printf("%d\n", x);
}
```

modul `modulA.c`

```
#include <stdio.h>
#include "proto.h"
extern int x = 20;

void fun2(void) {
   x += 4;
   printf("%d\n", x);
}
```

modul `modulB.c`

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

void fun3(void) {
   static int x = 5;
   x += 5;
   printf("%d\n", x);
}

void fun4(void) {
   extern int x;
   x += 6;
   printf("%d\n", x);
}
```

Provjeriti rješenje izvođenjem programa na vlastitom računalu i pri tome testirati prevođenje na dva načina:

- tako da se svi moduli prevedu i povežu samo jednim pozivom prevodioca
- tako da se zasebno prevede svaki modul, a zatim se dobiveni objektni kôd poveže u izvršni kôd

Rješenja

Rješenje 2. zadatka

modul **glavni.c**

```
#include <stdio.h>
#include "mat2.h"

int main(void) {
   printf("%d\n", factorial(0));
   printf("%d\n", factorial(25));
   printf("%d\n", factorial(26)); /* zasto ovdje rezultat nece biti dobar?*/
   printf("%d\n", binCoeff(13, 3));
   printf("%d\n", binCoeff(4, 4));
   printf("%d\n", iabsolute(-5));
   printf("%d\n", iabsolute(0));
   printf("%d\n", iabsolute(-5.7f));
   printf("%3.1f\n", fabsolute(-5));
   printf("%3.1f\n", fabsolute(-5.7f));
   return 0;
}
```

modul **mat2.c**

```
#include "mat2.h"

int binCoeff(int m, int n) {
   return factorial(m) / ( factorial(n) * factorial(m - n) );
}

int factorial(int n) {
   int i, f = 1;
   for (i = 2; i <= n; i++) {
      f = f * i;
   }
   return f;
}

int iabsolute(int n) {
   return n >= 0 ? n : -n;
}

float fabsolute(float x) {
   return x >= 0.0f ? x : -x;
}
```

datoteka s prototipovima funkcija **mat2.h**

```
int binCoeff(int m, int n);
int factorial(int n);
int iabsolute(int n);
float fabsolute(float x);
```

Rješenje 3. zadatka

Pri svakom pozivu funkcije varijabla x se ponovno inicijalizira. Ispisat će se:

```
5
5
```

Rješenje 4. zadatka

Varijabla x se inicijalizira samo jednom, na početku izvođenja programa, a njezina vrijednost ostaje sačuvana do kraja izvođenja programa (ne gubi se završetkom funkcije). Ispisat će se:

```
5
6
```

Rješenje 5. zadatka

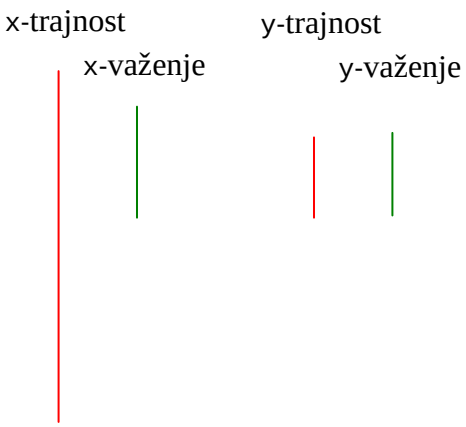
Potrebno je uočiti koje su varijable definirane u programu, te na temelju smještajnog razreda kojem pripadaju odrediti njihovo područje važenja i trajnost.

Varijabla x je statička vrijabla. To znači da se njezina trajnost proteže od početka do završetka programa. Varijabla se inicijalizira **samo jednom**, na početku izvođenja programa (čak i prije nego se prvi puta pozove funkcija), te njena vrijednost ostaje sačuvana do završetka programa. Varijabla x je definirana unutar funkcije, stoga se njezino područje važenja (tj. "područje programa u kojem je vidljiva") proteže od mjesta u funkciji na kojem je definirana do kraja funkcije.

Varijabla y je automatska varijabla. Varijabla se inicijalizira svaki puta kad se pozove funkcija, a njezina vrijednost se gubi u trenutku završetka funkcije.

```c
void fun(void) {
   static int x = 5;
   int y = 5;
   printf("%d %d\n", ++x, --y);
}

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

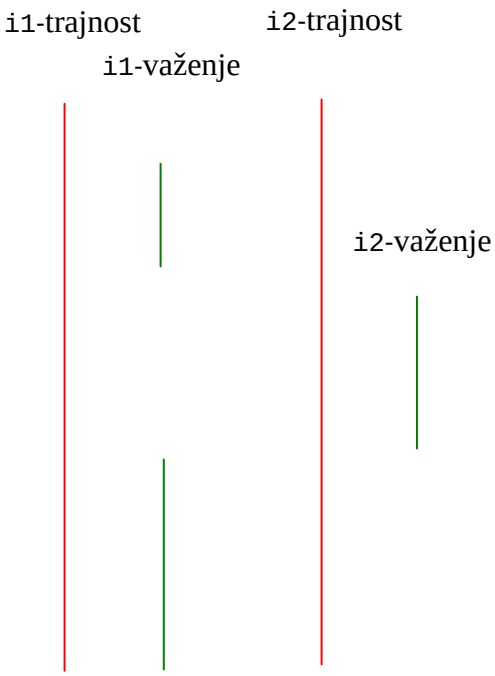


Rješenje 6. zadatka

Ovdje je važno uočiti da postoje dvije varijable naziva `i`. Na slici koja prikazuje područja važenja i trajnosti, prva varijabla `i` (definirana na početku glavnog programa) označena je oznakom `i1`, a druga oznakom `i2`.

```c
#include <stdio.h>

int main (void) {
   static int i = 5;
   int prviPut = 1;
   labela:
   {
      static int i = 10;
      int j = 15;
      printf("%d %d\n", i, j);
      i++;
      j++;
   }
   i++;
   printf("%d\n", i);
   if (prviPut) {
      prviPut = 0;
      goto labela;
   }
   return 0;
}
```



Rješenje 7. zadatka

Ovdje je važno uočiti da postoji 5 različitih varijabli naziva `x`. Kad se odredi trajnost i područje važenja svake od tih varijabli, zadatak je lako riješiti.

Rješenje 8. zadatka

```c
#include <stdio.h>

int zbroji(int x, int y);
int mnozi(int x, int y);

int main (void) {
   printf("2*2=%d\n", mnozi(2,2));
   printf("2+3=%d\n", zbroji(2,3));
   printf("4+2=%d\n", zbroji(4,2));
   printf("2*5=%d\n", mnozi(2,5));
   printf("2*3=%d\n", mnozi(2,3));
   return 0;
}

int zbroji(int x, int y) {
   static int brojPoziva = 0;
   brojPoziva++;
   printf("Funkcija zbroji do sada je pozvana %d puta\n", brojPoziva);
   return x+y;
}

int mnozi(int x, int y) {
   static int brojPoziva = 0;
   brojPoziva++;
   printf("Funkcija mnozi do sada je pozvana %d puta\n", brojPoziva);
   return x*y;
}
```

Rješenje 9. zadatka

```
#include <stdio.h>

int zbroji(int x, int y);
int mnozi(int x, int y);

int ukupniBrojPoziva = 0;


int main (void) {
   printf("2*2=%d\n", mnozi(2,2));
   printf("2+3=%d\n", zbroji(2,3));
   printf("4+2=%d\n", zbroji(4,2));
   printf("2*5=%d\n", mnozi(2,5));
   printf("2*3=%d\n", mnozi(2,3));

   return 0;
}

int zbroji(int x, int y) {
   static int brojPoziva = 0;
   brojPoziva++;
   ukupniBrojPoziva++;
   printf("Funkcija zbroji do sada je pozvana %d puta\n", brojPoziva);
   printf("Funkcije zbroji i mnozi do sada su pozvane %d puta\n", ukupniBrojPoziva);
   return x+y;
}

int mnozi(int x, int y) {
   static int brojPoziva = 0;
   brojPoziva++;
   ukupniBrojPoziva++;
   printf("Funkcija mnozi do sada je pozvana %d puta\n", brojPoziva);
   printf("Funkcije zbroji i mnozi do sada su pozvane %d puta\n", ukupniBrojPoziva);
   return x*y;
}
```

Rješenje 10. zadatka

Ovdje je važno uočiti postojanje **definicija** triju **različitih** varijabli x.

- na početku modula modulA.c **definirana** je eksterna varijabla x
- na početku funkcije main **definirana** je automatska varijabla x
- na početku funkcije fun3 **definirana** je statička varijabla x

Nakon što se odredi trajnost i područje važenja svake pojedine varijable, zadatak je lako riješiti.