5 Контрола на тек *Switch*

1. Switch структура

Структурата switch има слична употреба како низа на вгнездени if/else наредби. Општиот облик на структурата е следниот:

```
switch ( izraz ) {
case konstanta1 :
     naredbi1;
     . . . .
     break ;
case konstanta2 :
     naredbi2;
     . . . .
   break ;
. . . .
case konstantan :
   naredbin;
   . . . .
   break ;
default:            // ova e opciono
   naredbid;
   . . . .
   break ;
}
```

Со оваа структура, најпрво, се разрешува изразот напишан во заградите веднаш по зборчето **switch**, и потоа се прескокнува кон соодветната **case** константа (лабела) чија вредност е еднаква на вредноста на изразот. Не е дозволено да постојат дупли **case** константи. Вредноста на изразот izraz мора да биде цел број (int), знак (char) или енумерички тип. Константите (лабелите) во **case**, може да бидат наредени по кој било редослед, но **мора** да бидат константи. Доколку во **case** константите не постои вредност еднаква со вредноста на изразот, се извршуваат наредбите по лабелата **default**. Во случај да не постои **default** лабела, тогаш **switch** наредбата не извршува **ништо**. **Default** лабелата може да се наоѓа каде било во листата на лабели од **switch** наредбата, односно не мора да биде на крајот, но почитливо е ако се напише на крај.

Иако **default** не мора да се напише, понекоаш е згодно во секоја switch структура да постои и оваа лабела, дури и доколку наредбите во неа не прават ништо.

```
default:
/* Ne pravi nishto */
break;
```

Доколку по наредбите што одговараат на соодветната **case** лабела не се напише зборчето **break**, тогаш С продолжува со извршување на наредбите од следната **case** лабела, иако тоа не би било логично. **Break** овозможува **завршување** на наредбата **switch** и продолжување со следната наредба од програмата по неа. Затоа, **break** не смее да се заборави, освен доколку програмерот, има таква намера.

```
control = 0;
/* losh primer na programerska praksa */
switch (control) {
case 0:
     printf("Reset\n");
case 1:
     printf("Initializing\n");
     break;
case 2:
     printf("Working\n");
}
```

Доколку, на овој начин се напишат наредбите во switch структурата, тогаш ако вредноста на control == 0, програмата ќе испечати

```
Reset
Initializing
```

затоа што case 0 не завршува со break. По печатењето на Reset, програмата **продолжува** со извршување на наредбите од следната лабела (case 1) и печати Initializing.

2. Задачи:

1. Да се напише програма што ќе работи како едноставен калкулатор: од тастатура ќе чита оператор и два броја и ќе ја изврши наведената операција. Задачата да се реши најпрво со if, а потоа со switch стуктура.

СО IF:

```
#include <stdio.h>
int main()
{
     char operator;
     float br1, br2, rez=0;
     printf("Vnesete operator ('+', '-', '*', '/'):");
     scanf("%c", &operator);
     printf("Vnesete dva broja:"); scanf("%f %f",&br1,&br2);

     if (operator == '+')
     {
          rez = br1 + br2;
          printf("Rezultatot od operacijata: %.2f %c %.2f =
                 %f", br1, operator, br2, rez);
     }
```

```
     else if (operator == '-')
     {
          rez = br1 - br2;
          printf("Rezultatot od operacijata: %.2f %c %.2f =
                 %f", br1, operator, br2, rez);
     }
     else if (operator == '*')
     {
          rez = br1 * br2;
          printf("Rezultatot od operacijata: %.2f %c %.2f =
                 %f", br1, operator, br2, rez);
     }
     else if (operator == '/')
     {
            if (br2 == 0)
            {
               printf("Greshka: Delenje so 0\n");
               printf(" operacijata ke se ignorira\n");
            }
            else
            {
               rez = br1 / br2;
               printf("Rezultatot od operacijata: %.2f %c
                   %.2f = %f", br1, operator, br2, rez);
            }
     }
     else printf("Nepoznat operator %c\n", operator);
return 0;
}
```

СО SWITCH:

```
#include <stdio.h>
int main()
{
     char operator;
     float br1, br2, rez=0;
     printf("Vnesete operator ('+', '-', '*', '/'):");
     scanf("%c", &operator);
     printf("Vnesete dva broja:"); scanf("%f %f",&br1,&br2);

     switch (operator) {
     case '+':
          rez = br1 + br2;
          printf("Rezultatot od operacijata: %.2f %c %.2f =
                 %f", br1, operator, br2, rez);
          break;
     case '-':
          rez = br1 - br2;
          printf("Rezultatot od operacijata: %.2f %c %.2f =
                 %f", br1, operator, br2, rez);
```

```
          break;
     case '*':
          rez = br1 * br2;
          printf("Rezultatot od operacijata: %.2f %c %.2f =
                 %f", br1, operator, br2, rez);
          break;
     case '/':
          if (br2 == 0)
          {
               printf("Greshka: Delenje so 0\n");
               printf(" operacijata ke se ignorira\n");
          }
          else
          {
            rez = br1 / br2;
            printf("Rezultatot od operacijata: %.2f %c %.2f =
                   %f", br1, operator, br2, rez);
          }
          break;
     default:
          printf("Nepoznat operator %c\n", operator);
     break;
     }
return (0);
}
```

2. Да се напише програма што ќе овозможи претворање на броевите во зборови на следниот начин: За 89 како излез ќе се добие "osum devet". Дозволено е да се внесат **само двоцифрени** броеви.

```
#include <stdio.h>
int main()
{
     int broj,mala,golema;
     printf("Vnesete dvocifren broj:"); scanf("%d", &broj);
     mala = broj % 10;
     golema = broj/10;

     switch (golema) {
     case 0:
          printf("nula ");
          break;
     case 1:
          printf("eden ");
          break;
     case 2:
          printf("dva ");
          break;
     case 3:
          printf("tri ");
```

```
        break;
    case 4:
        printf("cetiri ");
        break;
    case 5:
        printf("pet ");
        break;
    case 6:
        printf("sest ");
        break;
    case 7:
        printf("sedum ");
        break;
    case 8:
        printf("osum ");
        break;
    case 9:
        printf("devet ");
        break;
    default:
        break;
    }
    switch (mala) {
    case 0:
        printf("nula\n");
        break;
    case 1:
        printf("eden\n");
        break;
    case 2:
        printf("dva\n");
        break;
    case 3:
        printf("tri\n");
        break;
    case 4:
        printf("cetiri\n");
        break;
    case 5:
        printf("pet\n");
        break;
    case 6:
        printf("sest\n");
        break;
    case 7:
        printf("sedum\n");
        break;
    case 8:
        printf("osum\n");
        break;
    case 9:
```

```
          printf("devet\n");
          break;
     default:
          break;
     }
return (0);
}
```

3. Доколку е познато дека бројот 85 се изговара како "osumdeset i pet", да се промени претходната програма така што ќе може за броевите **од 0 до 100**, да печати текст, онака како што се изговараат броевите. На пример: за 13 ќе отпечати "trinaeset", а за 100 ќе отпечати "sto".

```
#include<stdio.h>
int main()
{    int i;

     for (i=0;i<=100;i++)
     {    gotovo=0; okruglo=0;
          if (i>9)
          {    if (i==100) {printf("sto\n"); gotovo=1;}
               else
               {   if ((i/10)==1) /* vo intervalot [10-19] */
                    {  gotovo=1;
                       switch (i)
                       {
                         case 10: printf("deset\n");
                                  break;
                         case 11: printf("edinaeset\n");
                                  break;
                         case 12: printf("dvanaeset\n");
                                  break;
                         case 13: printf("trinaeset\n");
                                  break;
                         case 14: printf("cetirinaeset\n");
                                  break;
                         case 15: printf("petnaeset\n");
                                  break;
                         case 16: printf("sesnaeset\n");
                                  break;
                         case 17: printf("sedumnaeset\n");
                                  break;
                         case 18: printf("osumnaeset\n");
                                  break;
                         case 19: printf("devetnaeset\n");
                                  break;
                       }
                    }
                    else  /* znaci vo intervalot [20-99] */
                    {
```

```
                        switch (i/10)
                        {
                          case 2: printf("dvaeset");
                                  break;
                          case 3: printf("trieset");
                                  break;
                          case 4: printf("cetirieset");
                                  break;
                          case 5: printf("pedeset");
                                  break;
                          case 6: printf("seeset");
                                  break;
                          case 7: printf("sedumdeset");
                                  break;
                          case 8: printf("osumdeset");
                                  break;
                          case 9: printf("devedeset");
                                  break;
                        } /* switch */
                if (i%10==0) { okruglo=1; printf("\n"); }
                } /* kraj na else za if (i/10==1) */
              } /* kraj na else za if (i==100) */
          }/* kraj na if (i>9) */

          if (!gotovo)
          {   if ((i>20) && (!okruglo)) printf(" i ");
              switch (i%10)
              {  case 0: if (!okruglo) printf("nula \n");
                         break;
                 case 1: printf("eden\n");
                         break;
                 case 2: printf("dva\n");
                         break;
                 case 3: printf("tri\n");
                         break;
                 case 4: printf("cetiri\n");
                         break;
                 case 5: printf("pet\n");
                         break;
                 case 6: printf("sest\n");
                         break;
                 case 7: printf("sedum\n");
                         break;
                 case 8: printf("osum\n");
                         break;
                 case 9: printf("devet\n");
                         break;
              } /* switch */
          } /* kraj na if (!gotovo) */
     } /* kraj na for */
}/* kraj na main */
```

6 Функции, дефинирање, употреба и важност на променливите

1. Што се тоа функции?

Од претпоследната задача (задача 2 кај switch) се гледа делумно потребата за дефинирање на делови од програмата како посебни целини наречени функции, кои можат да се повикаат повеќе пати во текот на програмата со различни параметри.

Така на пример можевме делот со switch да го издвоиме во посебна функција која ќе ни ја печати текстуално дадената цифра и потоа двапати да ја повикаме функцијата – еднаш за цифрата mala еднаш за golema.

Оваа функција немаме потреба да ни враќа никаков резултат. Сепак во C функциите вообичаено враќаат некаков резултат, па и ние ќе ја направиме функцијата поопшта со тоа што ќе ставиме да враќа 1 ако отпечатила нешто и 0 во спротивно.

```
#include <stdio.h>
int cifra_vo_tekst(int cifra)
{  int rez=1;
     switch (cifra)
     {  case 0:
          printf("nula "); break;
        case 1:
          printf("eden "); break;
        case 2:
          printf("dva "); break;
        case 3:
          printf("tri "); break;
        case 4:
          printf("cetiri "); break;
        case 5:
          printf("pet "); break;
        case 6:
          printf("sest "); break;
        case 7:
          printf("sedum "); break;
        case 8:
          printf("osum "); break;
        case 9:
          printf("devet "); break;
        default:
          rez=0; /* znaci ne otpechatila nishto */
          break;
     }
     return rez; /* vrakja rezultat tamu od kade shto
}                   e povikana f-ijata */
```

```
int main()
{
     int broj,mala,golema;
     int rez1,rez2;
     printf("Vnesete dvocifren broj:"); scanf("%d", &broj);
     mala = broj % 10;
     golema = broj/10;

     rez1=cifra_vo_tekst(golema);
     rez2=cifra_vo_tekst(mala);

     if (!(rez1 && rez2)) printf("Greshka: trebashe da
     vnesete pozitiven dvocifren broj!");
}
```

Може пред main да се дефинира само заглавјето на функцијата, а потоа било каде после main да се дефинира и телото на функцијата:

```
int cifra_vo_tekst(int cifra); /* se najavuva samo zaglavjeto */

int main()
{
   ...
}
   ...

int cifra_vo_tekst(int cifra)
{
   ...          /* definicija na samata f-ija */
}
   ...
/* drugi funkcii */
```

Дефиницијата на една функција е од обликот:

```
tip_na_rezultatot ime_na_funkcijata(deklaracija_na_parametri)
{
   deklaracii na promenlivi, konstanti…
   naredbi;
}
```

Пример:
Типичен пример за функција е функцијата степен (која не е стандардна во C).

```
#include <stdio.h>
int stepen(int osnova, int n); /* najavuvanje na f-jata */

main()
{
     int i;
```

```
     for (i = 0; i < 10; ++i)
     printf("%d %d %d\n", i, stepen(2,i), stepen(-3,i));
     return 0;
}

/* stepen: podigni ja osnovata na n-ti stepen; n >= 0 */
int stepen(int osnova, int n)
{    int i, st=1;
     for (i = 1; i <= n; ++i)
        st = st * osnova;
     return st;
}
```

Дури и main е функција! И ако не се дефинира поинаку, сите функции се сметаат дека враќаат резултат од типот int.

Во C нема вгнездени функции една во друга како што има во Pascal на пример.

Во примерот се гледа дека функцијата stepen е повикана двапати со различни параметри (2 и -3). Веднаш се гледа предноста во дефинирањето на функциите: поедноставни програми за читање и најчесто пократки.

Понекогаш дел од програмата се прави функција дури и само еднаш да се повикува – заради полесна читливост. Ако е добро напишана функцијата, може да се употребува и без да се знае како точно работи – доволно е да се знае кои параметри ги прима на влез и што враќа на излез од функцијата.

Еве како би изгледала задачата која за даден природен број m ќе ги испише сите броеви помеѓу 1 и 1000 за кои збирот на цифрите изнесува m.

```
#include <stdio.h>
int zbir_na_cifri(int broj);
{
     int suma=0, cifra;

     while (broj > 0) {
          cifra = broj % 10;

          suma += cifra;
          broj /= 10;
     }
     return suma;
}
```

пример како може и без посебна променлива `cifra`:

```
int zbir_na_cifri(int broj);
{
     int suma=0;

     while (broj > 0) {
          suma += broj % 10;
          broj /= 10;
     }
     return suma;
}
```

```
int main ()
{
     int  m, i;
     printf("Vnesete go brojot m za sporedba: ");
     scanf("%d", &m);
     if (m > 27)
```

```
      printf("Ne postoi broj od 1 do 1000 so suma %d.\n", m);
      else
      {
            printf("Broevi cij zbir na cifri e %d se: \n", m);
            i = 1;
            while (i<=1000)
            {
                  if (zbir_na_cifri(i) == m) printf("%d\t", i);
                  i++;
            }
      }
   return 0;
}
```

Се гледа дека се елиминира потребата од многу променливи во главната програма – повеќето од нив се префрлаат во функцијата. Но треба да се запомни дека во C параметрите се пренесуваат според вредноста и не постои можност да се менува содржината на една променлива од главната програма во рамки на функцијата, ако таа се пренесе како параметар во функцијата.

Името на променливите нема значење при повикот. Тука во функцијата се вика broj, а во програмата се повикува со i. Дури и исто да се викаа, не се менува содржината на i во главната програма.

C-компајлерот создава нова променлива со важност само во рамки на функцијата и тоа додека таа се извршува. По завршувањето на функцијата вредноста на таа променлива се губи, а исто и на сите други променливи декларирани во рамки на функцијата – наречени **локални**.

Наспорти нив, **глобални** променливи се дефинираат надвор од main и тие важат во сите функции вклучувајќи ја и main.