7 – Struktury a uniony

Příklad 7.1
Následující příklad ukazuje, kterak se strukturou pracovat.

```
#include<stdio.h>
typedef struct
{
    char st_jmeno[20], identifikace[10];
    short st_mesic,st_den, st_rok;
}   STUDENT;

STUDENT  Novak, Homola, Chouchela;

main()
{
STUDENT Novak = { "Jan  Novak","123456789", 5, 21, 1973};
STUDENT Chouchela ={" Eman Chouchela","987654321",14,8,1980};
printf("\n Jmeno studenta je %s", Novak.st_jmeno);
printf("\nRok narozeni dalsiho je %d",Chouchela.st_rok);
}
```

Příklad 7.2
Tento příklad ukazuje, kolik místa zabírá námi vyrobený nový typ struktura DIVNOST.

```
#include<stdio.h>
typedef struct
{ char i,j,k;
  int a,b,c;
  float d,e,f;
} DIVNOST;

DIVNOST k,l,m;

main()
{
printf("Velikost k je %d", sizeof(k));
printf("Velikost typu DIVNOST je %d", sizeof(DIVNOST));
}
```

Příklad 7.3
Následující příklad ukazuje použití struktury ve struktuře. Příklad dokončete následovně: pole struktur naplňte hodnotami. Poté vyhledejte a vypište, kdo má nejvyšší plat. Vyzkoušejte si všechny způsoby , jak přistupovat k jednotlivým položkám struktury.

```
#include <stdio.h>
#include <string.h>

#define MAX   7

typedef struct {
  char ulice[30];
  int  cislo;
} ADRESA;

typedef struct {
  char jmeno[20];
  ADRESA adresa;
```

```
  float  plat;
} CLOVEK;

CLOVEK lide[MAX];
```

Příklad 7.4
Vyrobte jednorozměrné pole, které naplňte např. vzestupně čísly 1 až 10. Toto pole překopírujte jediným příkazem do jiného pole. Pro tuto operaci použijte pole ve struktuře, protože jazyk C tuto operaci jinak jediným příkazem neumožňuje.

Příklad 7.5
Zjistěte, jak jsou prvky struktury uloženy v paměti. Předpokládejme následující strukturu:

```
typedef struct {
    char j;
    int k,l,m;
    char  d;
} PRIKLAD;
PRIKLAD muj;
```

Zjistěte, na jakých adresách položky struktury muj leží. Zjistěte velikost dané struktury.

Příklad 7.6
Definujte strukturu s prvky float,char a int. Poté definujte union se stejnými prvky. Zjistěte adresy struktury, unionu a všech položek.

Příklad 7.7
Následující příklad ukazuje, jak pracovat s dynamickou strukturou.

```
#include <stdio.h>
#include <stdlib.h>

typedef struct {
  char jmeno[30];
  int  rocnik;
} ZAK;

main()
{
  ZAK s, *p_s;

  if ((p_s = (ZAK *) malloc(sizeof(ZAK))) == NULL) {
    printf("Malo pameti \n");
    return;
  }

  s.rocnik = 1;
  printf("Rocnik je %d \n", s.rocnik);

  (*p_s).rocnik = 4;
  printf("Rocnik je %d \n", (*p_s).rocnik);

  p_s->rocnik = 2;
  printf("Rocnik je %d \n", p_s->rocnik);
}
```

Příklad 7.8
Příklad ukazuje, kterak přiřazovat struktury.

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

```
typedef struct {     /* definice struktury jejimz  */
  int a[10];         /* jedinym prvkem je pole int */
} STR_POLE;

typedef struct rozmery {
  int   vyska;
  float vaha;
} ROZMERY;

main()
{
  ROZMERY marie, hanka;
  STR_POLE  pole1, pole2;

  marie.vyska = 170;
  pole1.a[1] = 9;
  hanka = marie;
  pole2 = pole1;

  printf("Hanka vyska %d \n", hanka.vyska);
  printf("pole2.a[1] = %d \n", pole2.a[1]);
}
```

Příklad 7.9

Při použití unionů kompilátor zachází s jednotlivými bity vždy předepsaným způsobem, tj. v našem případě jednou dané bity interpretuje jako long a jednou jako float . Při přetypování postupuje kompilátor jiným způsobem. Zde kompilátor provede konverzi do cílové podoby.

```
#include <stdio.h>
#include<conio.h>
int main( void )
{
  union {
    long long_element;
    float float_element;
  } u;
  long lng_var;
  float flt_var;
  clrscr();

  lng_var = u.long_element = 10;
  printf( "Hodnota  lng_var pretypovane na  float je: %f\n",
            (float) lng_var );
  printf( "Hodnota  float elementu po prirazeni\n"
            "do long_elementu je: %f\n\n",
            u.float_element );

  flt_var = u.float_element = 3.555;

  printf( "Hodnota  flt_var pretypovane na long je: %d\n",
            (long) flt_var );
  printf( "Hodnota  long_elementu po \n"
            "prirazeni do float_elementu is: %d\n",
            u.long_element );
}
```

Příklad 7.10

Definujte strukturu, která bude mít položky jmeno a vek. Vytvořte funkci pln( ), která strukturu naplní Vašimi daty. První parametr funkce bude právě pointer na strukturu. Zkuste strukturu vytvořit jak staticky, tak dynamicky. Dále vytvořte funci tiskni ( ), která umí strukturu vytisknout položkách. Vytvořte program, ve kterém vytvoříte postupně dvě proměnné typu struktura, obě naplníte a obě vytiskněte za použití uvedených funkcí.

Příklad 7.11

Vytvořte spojový seznam, který bude obsahovat N celých čísel od jedné do N. N zadá uživatel programu. Program dále rozšiřte o funkci sudá, ta z uvedeného spojového seznamu vymaže všechna sudá čísla a ponechá jen lichá. Poté spojový seznam vypište.

Příklad 7.12

Vytvořte spojový seznam, který bude obsahovat N celých čísel od jedné do N. N zadá uživatel programu. Program dále rozšiřte o funkci lichá, ta z uvedného spojového seznamu vymaže všechna lichá čísla a ponechá jen sudá. Poté spojový seznam vypište.

Příklad 7.13

Struktura má následující deklaraci:

```
struct{
    char jmeno[30];
    enum {zena, muz}  pohlavi;
    union{
        enum {ne,ano} vojak;
        char rozena[20];
        } zdata;
    } zamestnanec;
```

Popište, jak je struktura vytvořena. Poté vytvořte program, ve které bude struktura použita pro dvě proměnné Adam a Eva. Obě naplňte a vypište. Uvědomte si, že na člen union se musíme odvolat např. zamestnanec.zdata.vojak apod. U obou proměnných navíc vypište, na jakých adresách jsou jednotlivé položky.

Od RNDr. Dagmar Brechlerové.