Lekcja 23 – struktury (zbiór elementów różnego typu)

Temat: Struktura (rekord)

Kod programu źródłowego:

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

using namespace std;

int main(int argc, char *argv[])
{
  int i,j,k;
  struct adres             //deklaracja struktury
  {
    char imie[20];         //zbiór danych struktury
    char nazwisko [30];
    int wiek;
    char plec;
    bool obywatelstwo;
  } adresik;               //konkretny przedstawiciel
  do
  {
   cout <<"Co chcesz zrobic\n";
   cout <<"1-zapisz dane adresowe\n";
   cout <<"2-odczytaj dane zapisane w komorce adresowej\n";
   cout <<"3-zakoncz\n";
   cin >> i;
   switch (i)
   {
     case  1:
     {
        cout <<"Podaj imie\n";
        cin >> adresik.imie;
        cout <<"Podaj nazwisko\n";
        cin >> adresik.nazwisko;
        cout <<"Ile masz lat\n";
        cin >> adresik.wiek;
        cout <<"Podaj plec: k - kobieta, m - meżczyzna\n";
        cin >> adresik.plec;
        cout <<"Czy masz obywatelstwo polskie? 1 - tak, 0 - nie\n";
        cin >> adresik.obywatelstwo;
     };
     break;
     case  2:
     {
        cout <<"Imie "<< adresik.imie << endl;
        cout <<"Nazwisko "<< adresik.nazwisko << endl;
        cout <<"Wiek "<< adresik.wiek << endl;
        cout <<"Plec "<< adresik.plec << endl;
        cout <<"Obywatelstwo "<< adresik.obywatelstwo<< endl;
```

```
      };
      break;
    }
  } while (i!=3);
  return 0;
}
```

Opis struktury: Struktura (rekord) to rodzaj danych, w którym pogrupowano elementy różnego typu w jednym obszarze pamięci.

Taka struktura może obejmować dane całkowite, zmiennoprzecinkowe, łańcuchy tekstowe, znaki, typy logiczne.

Dzięki temu dane różnego typu, ale powiązane ze sobą logicznie są zebrane razem w ramach jednej struktury.

Deklaracja struktury:

`struct adres`	deklaracja struktury

`{`
`char imie[20];`
`char nazwisko [30];`
`int wiek;`
`char plec;`
`bool obywatelstwo;`
`}`
	zbiór składowych struktury

`adresik;`	Zmienna typu struktury

Nowe zmienne tego typu strukturalnego:

1. Możemy je zadeklarować zaraz po zdefiniowaniu struktury

```
struct adres
{
    …
} adresik1, adresik2;
```

2. Można to zrobić tak jak definiuje się inne zmienne. Wcześniej jednak należy definiować daną strukturę.

```
struct adres
  {
```

```
    …
}

adres adresik1, adresik2;
```

Unikalna nazwa struktury i składowych:

Struktura musi mieć unikalną nazwę. Jej składowe również muszą mieć swoje unikatowe (w obrębie struktury) nazwy.

```
struct adres
{
  char imie[20];
  char imie[30];
  char plec;
  bool obywatelstwo;
  int wiek;
  float wiek;
} adresik;
```

Zdublowane składowe `imie`

Zdublowane składowe `wiek`. Jedna typu całkowitego, inna zmiennoprzecinkowego.

Dostęp do pól struktury:

Składowe struktury — pola — są etykietowane. Mają swoje unikatowe nazwy. Poprzez podanie nazwy otrzymuje się dostęp do danego pola.

```
adresik.imie
```

Nazwa struktury	Kropka rozdzielająca nazwę struktury od nazwy składowej	Nazwa składowej
`adresik`	`.`	`imie`

Przykłady użycia:

cout << `adresik.imie;`

```
cin >> adresik.imie;

adresik.imie = ”Jan”;

tekst = adresik.imie;
```

Tablica struktur:

Zastosowanie:

Struktury są powszechnie stosowane w programowaniu. Pozwalają w przejrzysty sposób opisywać złożone obiekty. Przykładem struktury może być informacja o książce, której pola będą zawierały: imię i nazwisko autora (typ łańcuchowy), tytuł (typ łańcuchowy), rok wydania (liczba całkowita), liczba stron (liczba całkowita), nazwa wydawnictwa, numer ISBN itp.