

XIV. Struktury danych

14.1. Organizacja danych

Jeśli próbowałeś już pisać jakąś bazę danych, np. przechowującą kontakty telefoniczne do różnych osób to zauważyłeś pewnie, że szukanie nazw tablic i posługiwanie się wieloma tablicami dla jednej osoby nie jest wygodne. Z pomocą przychodzą tu **struktury**, które pozwalają na zorganizowanie danych w wygodniejszy sposób. **Struktury potrafią przechowywać różne typy danych.** Deklaracja **struktury** wygląda następująco:

```
//struktura dla użytkowników forum
struct Uzytkownicy_DDT //deklaracja struktury
{ //opis struktury
    std::string nik;
    std::string haslo;
    short data_zalozenia_konta;
    std::string status;
};
```

Elementy charakteryzujące ten typ danych to:

→ słowo kluczowe **struct**, które informuje kompilator, że będzie definiowana struktura,

→ opis struktury opisujący zawartość struktury,

→ jest to najprostsza postać klasy.

W przykładzie został wykorzystana **biblioteka string**, zapoznaj się z nią nim przejdziesz do analizy kursu!

14.2. Wykorzystanie struktury w praktyce

Ponieważ wiemy już jak utworzyć własny, bardziej złożony typ danych to dobrze by było, gdybyśmy teraz wiedzieli jak go możemy wykorzystać. Zmienną deklarujemy tak samo, jak w przypadku standardowych typów danych.

```
//struktura dla użytkowników forum-----
#include <iostream>
#include <string>
#include <conio.h>
//struktura dla użytkowników forum
struct Uzytkownicy_DDT //deklaracja struktury
{
    //opis struktur
    std::string nik;
    std::string haslo;
    short lp;
    std::string status;
    std::string rodzaj_konta;
};

int main()
{
    using namespace std;
    //Tworzenie obiektu struktury i wypełnianie-----
    Uzytkownicy_DDT PiotrSzawdyski =
```

```

{
    "PiotrSzawdyski",
    "!@#$$%%^^",
    1,
    "Aktywny",
    "Admin"
};
/*WAŻNE dla niektórych kompilatorów należy
użyć zapisu static Uzytkownicy_DDT PiotrSzawdyski =*/
Uzytkownicy_DDT Pietrzuch =
{
    "Pietrzuch",
    "@m@m$@#@@",
    24,
    "Aktywny",
    "Moderator"
};
/*inny sposób wprowadzania danych do
obiektu*/
//tworzenie obiektu
Uzytkownicy_DDT Piotrus_Pan;
/*wypełnianie obiektu wartościami dla
poszczególnych typów danych*/
Piotrus_Pan.haslo = "PanPiotr";
Piotrus_Pan.lp = 927;
Piotrus_Pan.nik = "Piotrus_Pan";
Piotrus_Pan.rodzaj_konta = "Uzytkownik";
Piotrus_Pan.status = "zablokowany";

//wyświetlenie obiektów
cout << "Oto lista uzytkownikow forum DDT"
<< endl << "Nik " << PiotrSzawdyski.nik
<< endl << "haslo " << PiotrSzawdyski.haslo
<< endl << "Nr usera " << PiotrSzawdyski.lp
<< endl << "Status " << PiotrSzawdyski.status
<< endl << "Rodzaj Konta "
<< PiotrSzawdyski.rodzaj_konta
<< endl
<< endl << "Nik " << Pietrzuch.nik
<< endl << "haslo " << Pietrzuch.haslo
<< endl << "Nr usera " << Pietrzuch.lp
<< endl << "Status " << Pietrzuch.status
<< endl << "Rodzaj Konta "
<< Pietrzuch.rodzaj_konta
<< endl
<< endl << "Nik " << Piotrus_Pan.nik
<< endl << "haslo " << Piotrus_Pan.haslo
<< endl << "Nr usera " << Piotrus_Pan.lp
<< endl << "Status " << Piotrus_Pan.status
<< endl << "Rodzaj Konta "
<< Piotrus_Pan.rodzaj_konta;

getch();
return(0);
}

```

14.3. Analiza kodu

Jak pewnie się zorientowałeś **strukturę** można wypełnić podobnie jak tablicę, gdy oczywiście przed uruchomieniem programu znamy wartości jakie ma zawierać. Co warto podkreślić **struktura Uzytkownicy_DDT** posiada trzy obiekty **PiotrSzawdynski**, **Pietrzuch**, **Piotrus_Pan**. By wprowadzić bądź wyświetlić wartości danego obiektu, musimy użyć **kropki (.)!!!**

```
nazwa_obiektu.nazwa_zmiennej = wartość_wprowadzana
```

Większość kompilatorów po wprowadzeniu kropki z klawiatury wyświetla podpowiedź dla danego obiektu (typy danych dla tego obiektu (struktury)).

WAŻNE !!! dla niektórych kompilatorów należy użyć **słowa kluczowego static** **Uzytkownicy_DDT PiotrSzawdynski**.

```
static Uzytkownicy_DDT PiotrSzawdynski =
{
    "PiotrSzawdynski",
    "!@#$$%%^^",
    1,
    "Aktywny",
    "Admin"
};
```

14.4. Tablice struktur

```
//Tablice struktur-----
#include <iostream>
#include <string>
#include <conio.h>
//struktura dla użytkowników forum
struct DDT //deklaracja struktury
{
    //opis struktury
    std::string nik;
    std::string haslo;
    std::string status;
    std::string rodzaj_konta;
};

int main()
{
    using namespace std;
    //Tworzenie obiektu struktury-----
    DDT Uzytkownicy[3];
    cout << "Wprowadz dane.\n";
    //wprowadzenie 3 użytkowników
    for(short licz = 0; licz < 3; licz++){
        cout << "Podaj nik ";
        cin >> Uzytkownicy[licz].nik;
        cout << "Podaj haslo ";
        cin >> Uzytkownicy[licz].haslo;
    }
    //wyswietlanie wprowadzonych danych
    cout << endl
```

```

<< "Oto rezultat wprowadzonych danych:"
<< endl;
for(short licz = 0; licz < 3; licz++){
    cout << "Uzytkownik " << licz + 1
    << " Nik " << Uzytkownicy[licz].nik
    << " haslo " << Uzytkownicy[licz].haslo
    << endl;
}

getch();
return(0);
}
//-----

```

Właściwie nie ma różnic między użyciem tablicy zmiennych a tablicy struktur. Oczywiście możemy też wypełnić tablice w ten sposób:

```

DDT Uzytkownicy[3] =
{
    //Uzytkownik[0]
    {"PiotrSzawdyski", "!@#$$%%^"},
    //Uzytkownik[1]
    {"Pietrzuch", "@m@m$@#@@"},
    //Uzytkownik[2]
    {"Piotrus_Pan", "PanPiotr"}
};

```

14.5. Przypisanie struktury do struktury

Tak jak zmienną można przypisać do innej zmiennej tak samo można postąpić z strukturą danych. Wystarczy użyć operatora „=”.

```

//Przypisanie struktur-----
#include <iostream>
#include <string>
#include <conio.h>
//struktura dla użytkowników forum
struct DDT //deklaracja struktury
{
    //opis struktury
    std::string nik;
    short wiek;
    char plec;
};

int main()
{
    using namespace std;
    DDT Monika, Mariola, Kasia;

    Monika.nik = "Mika";
    Monika.plec = 'K';
    Monika.wiek = 20;

    //przypisanie struktur

```

Kasia = Mariola = Monika;

```
cout << "Wyświetla uzytkownikow plci pieknej"  
<< endl  
<< "Nik - " << Kasia.nik << endl  
<< "Nik - " << Mariola.nik << " wiek "  
<< Mariola.wiek << endl  
<< "Nik - " << Monika.nik << " plec "  
<< Monika.plec << endl;  
getch();  
return(0);  
}  
//-----
```

14.6. Ćwiczenia

1. Farmer zwrócił się do Ciebie z prośbą byś napisał program, który będzie przechowywał informacje o rodzajach zwierząt jakie posiada. Zwierzęta to krowa, koza, kura, pies i świnia. Cechy tych zwierząt to nazwa, waga, wiek(podany w tygodniach), oraz data nabycia(zakupu). Farmer dodał, iż krowę i świnie kupił 24-04-08r od znajomego, pies przybłąkał się 6.05.02r, natomiast kozę i kurę dostał w prezencie od wójta 23.02.09. Niestety resztę cech musisz określić sam, na podstawie przyjętych średnich(google) dla tych zwierząt. Dane mają być zawarte w strukturze. Rezultatem programu ma być:

a) pierwszy program→ wyświetlenie wszystkich danych w przejrzysty i zrozumiały dla użytkownika sposób, użytkownik nic nie podaje, program wyświetla tylko dane.

b) drugi program→ program ma poprosić o wprowadzenie nazwy i wagi(w kilogramach) zwierzęcia(mają być wprowadzone trzy zwierzęta) , oraz wyświetleniu danych w odwrotny sposób do tego w jaki zostały wprowadzone, oraz zachowaniu takiej kompozycji

Trzoda zawiera m.in.:

1.→ nazwa_zwierzęcia jego waga to waga_zwierzęcia(podana w gramach) gram.

2.→

3.→