

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования


ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра экономической математики, информатики и статистики (ЭМИС)

ТИПЫ ДАННЫХ, ОПРЕДЕЛЯЕМЫЕ ПОЛЬЗОВАТЕЛЕМ. СТРУКТУРЫ.

Отчёт по лабораторной работе по дисциплине “Программирование”

Студент гр. 543-1

 /А.Е. Мухамеджан

“ ____ ” _____ 2024

Доцент кафедры ЭМИС

_____ Е.А. Шельмина
оценка

“ ____ ” _____ 2023 г.

Томск 2024

Лабораторная работа №2

“ Типы данных, определяемые пользователем. Структуры.”

Цель работы – познакомиться с типами данных, определяемыми пользователем.

Структура – это объединенное в единое целое множество поименованных элементов в общем случае разных типов. В отличие от массива, все элементы которого однотипны, структура может содержать элементы разных типов.

Каждая структура включает в себя один или несколько объектов (переменных, массивов, указателей, структур и т. д.), называемых элементами структуры (компонентами). Элементы структуры также называются полями структуры и могут иметь любой тип, кроме типа этой же структуры, но могут быть указателями на него. Структуры, так же, как и массивы относятся к структурированным типам данных. Они отличаются от массивов тем, что, во-первых, к элементам структуры необходимо обращаться по имени, во-вторых, все поля структуры необязательно должны принадлежать одному типу.

```
struct [ имя_типа ]  
{тип_1 элемент_1; тип_2 элемент_2;  
...  
тип_n элемент_n;  
} [ список_описателей ];
```

Такое определение вводит новый производный тип, который называется структурным типом. Если список описателей отсутствует, описание структуры определяет новый тип, имя которого можно использовать в дальнейшем наряду со стандартными типами

Если отсутствует имя типа, должен быть указан список описателей переменных, указателей или массивов. В этом случае описание структуры служит определением элементов этого списка.

```
struct {  
    char fio[30]; int age, code; double salary;  
} x, staff[100], *ps;
```

Имя структуры можно использовать сразу после его объявления (определение можно дать позднее) в тех случаях, когда компилятору не требуется знать размер структуры.

Для инициализации структуры значения ее элементов перечисляют в фигурных скобках в порядке их описания.

```
struct Worker {  
    char fio[30]; int age, code; double salary;};  
Worker ivanov = {"Иванов И.И.", 31, 215, 5800.35};
```

При инициализации массивов структур следует заключать в фигурные скобки каждый элемент массива.

```
struct complex {  
    float re, im; } compl[3] = { { 1.3, 5.2 }, { 3.0, 1.5 }, { 1.5, 4.1 } };
```

Для переменных одного и того же структурного типа определена операция присваивания, при этом происходит поэлементное копирование. Но присваивание – это и все, что можно делать со структурами целиком. Другие операции, например сравнение на равенство или вывод, не определены.

Структуру можно передавать в функцию и возвращать в качестве значения функции.

Размер структуры не обязательно равен сумме размеров ее элементов, поскольку они могут быть выровнены по границам слова.

Доступ к полям структуры выполняется с помощью операций выбора . (точка) при обращении к полю через имя структуры и -> при обращении через указатель.

Ввод/вывод структур, как и массивов, выполняется поэлементно.

```
Worker worker, staff[100], *ps;  
worker.fio = "Петров С.С."; staff[3] = worker; staff[8].code = 123;  
ps->salary = 4350.00;
```

Переменные структурного типа можно размещать и в динамической области памяти, для этого надо описать указатель на структуру и выделить под нее место

```
Worker *ps = new Worker; //создает переменную структурного типа  
Worker *pms = new Worker[5]; //создает массив структурного типа  
//обращение через операцию косвенного доступа ps->age = 55;  
//обращение через разыменовывание указателя  
(*ps).code = 253;  
//обращение к 0 элементу созданного массива через индекс pms[0].salary  
= 5800;  
//обращение к 1 элементу созданного массива через указатель  
(*(pms + 1)).salary = 4800;  
//очистка занимаемой памяти  
delete ps; delete []pms;
```

Если элементом структуры является другая структура, то доступ к ее элементам выполняется через две операции выбора.

```
struct A {  
    int a; double x;};  
struct B {  
    A a; double x;};  
    B x[2]; x[0].a.a = 1;  
    x[0].a.x = 35.15;  
    x[1].x = 0.1;
```

Как видно из примера, поля разных структур могут иметь одинаковые имена, поскольку у них разная область видимости.

Ход работы:

Задания для самостоятельной работы:

Задание 1. Разработать структуру данных согласно варианту.

Вариант 12. Описать структуру с именем NOTE, содержащую поля:

- фамилия и имя;
- номер телефона;
- дата рождения (массив из трех чисел).

Написать программу, выполняющую следующие действия:

- ввод с клавиатуры данных в массив, состоящий из 9 структур типа NOTE;
- вывод на экран информации о людях, чьи дни рождения приходятся на год, значение которого введено с клавиатуры;
- если таких людей нет, вывести соответствующее сообщение.

Для выполнения задания 1 (приложение А) использовался вариант 12, представленный в лабораторной работе №2 и интегрированная среда разработки Microsoft Visual Studio.

Вывод: в результате выполнения лабораторной работы получены навыки работы со структурами.

Приложение А

(обязательное)

Код программы задания 1

```
#include <iostream>
#include <format>
#include <string>
using namespace std;
struct NOTE
{
    string FName; string phone; int* birthday = new int[3];
};
void Print(NOTE n);
int N = 3;
int main()
{
    setlocale(LC_ALL, "Rus");
    srand(time(NULL));
    NOTE* arrN = new NOTE[N];
    for (int i = 0; i < N; i++) {
        cout << "Введите фамилию и имя: ";
        /*string tempSurname, tempName;
        cin >> tempSurname >> tempName;
        arrN[i].FName = tempSurname + tempName;*/
        arrN[i].FName = "dasdnsna" + to_string(i);
        cout << "Введите дату рождения: ";
        int* temp = new int[3];
        /*cin >> arrN[i].birthday[0] >> arrN[i].birthday[1] >>
        arrN[i].birthday[2];*/
        cout << "Введите номер телефона: ";
        /*cin >> arrN[i].phone;*/
        temp[0] = rand() % (31 - 1 + 1) + 1;
        temp[1] = rand() % (12 - 1 + 1) + 1;
        temp[2] = rand() % (2000 - 1970 + 1) + 1970;
        arrN[i].birthday = temp;
        arrN[i].phone = to_string(rand() % (8999999999 - 89000000000 + 1) +
89000000000);
        Print(arrN[i]);
    }
    cout << "Введите искомый год: ";
    int data;
    cin >> data;
    bool c1 = false;
    for (int i = 0; i < N; i++) {
        if (arrN[i].birthday[2] == data) {
            Print(arrN[i]);
            c1 = true;
        }
    }
    if (!c1) {
        cout << "Людей такого года рождения нет";
    }
}
void Print(NOTE n) {
    cout << "Данные:\nФамилия Имя: " << n.FName;
    cout << "\nДата рождения: ";
    for (int j = 0; j < 3; j++) {
        cout << n.birthday[j] << ".";
    }
    cout << "\nНомер телефона: " << n.phone << "\n";
}
```