# Лабораторная работа №16. Типы данных, определяемые пользователем. Структуры

## 1 Цель и порядок работы

Цель работы – ознакомиться с типами данных, определяемыми пользователем и их применением в процессе программирования.

Порядок выполнения работы:

* ознакомиться с описанием лабораторной работы;
* получить задание у преподавателя, согласно своему варианту;
* написать программу и отладить ее на ЭВМ;
* оформить отчет.

## 2 Краткая теория

В реальных задачах, обрабатываемая информация может иметь достаточно сложную структуру. Язык программирования C++ позволяет определять свои типы данных и правила работы с ними. Рассмотрим эти возможности подробней.

### 2.1 Определение типов (typedef)

Для того чтобы сделать программу более легкой для восприятия, удобно задавать используемым типам новые мнемонические имена, отражающие их суть в разрабатываемой программе. Кроме того, принято употреблять такие имена для сложных составных типов воспринимаемых, в противном случае, с трудом, например при объявлении указателей на функции.

Директива typedef позволяет задать синоним для встроенного или пользовательского типа данных. Имена, определенные с помощью директивы typedef, можно использовать точно так же, как обычные спецификаторы типов.

Синтаксис директивы typedef имеет два варианта:

typedef тип новое\_имя;

typedef тип новое\_имя [ размерность ];

Второй вариант используется при определении нового имени типа для массива, а квадратные скобки являются элементом синтаксиса.

Например:

typedef unsigned int UINT; //UINT - беззнаковое целое

typedef char msg[100]; //msg - массив из ста символов типа char

Объявленные таким образом имена типов используются так же, как и имена стандартных типов:

UINT i, j; //две переменных типа unsigned int

UINT А[10]; //массив переменных типа unsigned int

msg m; //массив (строка) из ста символов типа char

msg strs[10]; //массив из 10 строк по сто символов каждая

Кроме задания типам более коротких псевдонимов typedef используется для облегчения переносимости программ. При изменении машинно-зависимых типов, при переносе необходимо будет изменить только оператор определения типа без изменения всей остальной программы.

### 2.2 Структуры (struct)

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 злемент\_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

Например:

struct Worker{ // описание нового типа Worker

 char fio[30];

 int age, code;

 double salary;

}; // описание заканчивается точкой с запятой

// определение переменной, массива типа Worker и указателя на тип Worker

Worker y, staff[100], \*ps;

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

// Определение переменной, массива структур и указателя на структуру

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.

struct List; // объявление структуры List

struct Link

{

 List \*р; // указатель на структуру List

 Link \*prev, \*succ; // указатели на структуру Link

}

struct List { /\* определение структуры List \*/};

Это позволяет создавать связные списки структур.

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 А

{

 int а;

 double х;

};

struct В

{

 А а;

 double х;

};

B x[2];

х[0].а.а = 1;

х[0].а.x = 35.15;

х[1].х = 0.1;

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

### 2.3 Битовые поля

Битовые поля – это особый вид полей структуры. Они используются для плотной упаковки данных, например, флажков типа «да/нет». Минимальная адресуемая ячейка памяти – 1 байт, а для хранения флажка достаточно одного бита. При описании битового поля после имени через двоеточие указывается длина поля в битах (целая положительная константа).

struct Options

{

 bool centerX:1;

 bool centerY:1;

 unsigned int border\_type:2;

 unsigned int color\_index:4;

}

Битовые поля могут быть любого целого типа. Имя поля может отсутствовать, такие поля служат для выравнивания на аппаратную границу. Доступ к полю осуществляется обычным способом – по имени. Адрес поля получить нельзя, однако в остальном битовые поля можно использовать точно так же, как обычные поля структуры. Следует учитывать, что операции с отдельными битами реализуются гораздо менее эффективно, чем с байтами и словами, так как компилятор должен генерировать специальные коды, и экономия памяти под переменные оборачивается увеличением объема кода программы. Размещение битовых полей в памяти зависит от компилятора и аппаратуры.

### 2.4 Примеры использования

Пример 1. Нахождение расстояния между точками в декартовой системе координат

#include "stdafx.h"

#include <iostream>

#include <math.h>

using namespace std;

//объявляем структуру точка

struct point

{

 float X;

 float Y;

};

float dst(point p, point q)

{

 return sqrt((p.X - q.X)\*(p.X - q.X) + (p.Y - q.Y)\*(p.Y - q.Y));

}

int main()

{

 setlocale(LC\_ALL, "Russian");

 point a, b;

 cin >> a.X >> a.Y;

 cin >> b.X >> b.Y;

 float d = dst(a, b);

 cout << "Расстояние между точками: " << d << endl;

 return 0;

}

Пример 2. Работа с каталогом книг

#include "stdafx.h"

#include <iostream>

#include <math.h>

using namespace std;

//объявляем структуру книга

struct book

{

 char author[40];

 char title[100];

 int year;

 int pages;

 int quantity;

 float price;

};

int main()

{

 setlocale(LC\_ALL, "Russian");

 int N;

 cout << "Введите количество книг в каталоге: " << endl;

 cin >> N;

 //созадим динамически каталог книг

 book \*catalog = new book[N];

 //введем данные о книгах с клавиатуры

 for (int i = 0; i < N; i++)

 {

 cout << "Введите имя автора: " << endl;

 cin >> catalog[i].author;

 cout << "Введите название: " << endl;

 cin >> catalog[i].title;

 cout << "Введите год издания: " << endl;

 cin >> catalog[i].year;

 cout << "Введите количество страниц: " << endl;

 cin >> catalog[i].pages;

 cout << "Введите количество книг: " << endl;

 cin >> catalog[i].quantity;

 cout << "Введите цену книги: " << endl;

 cin >> catalog[i].price;

 cout << endl;

 }

 //выведем на экран книги, цена которых меньше 200 руб

 //и найдем сумму всех книг в каталоге

 float s = 0;

 for (int i = 0; i < N; i++)

 {

 if (catalog[i].price < 200)

 {

 cout << "Автор: " << catalog[i].author << endl;

 cout << "Название: " << catalog[i].title << endl;

 cout << "Год издания: " << catalog[i].year << endl;

 cout << "Количество страниц: " << catalog[i].pages << endl;

 cout << "Количество книг: " << catalog[i].quantity << endl;

 cout << "Цена книги: " << catalog[i].price << endl;

 cout << "Сумма: " << catalog[i].price\*catalog[i].quantity << endl;

 cout << endl;

 }

 s += catalog[i].price\*catalog[i].quantity;

 }

 cout << "Общая сумма: " << s << endl;

 delete []catalog;

 return 0;

}

## 3 Контрольные вопросы

1. Как объявить пользовательский тип данных?
2. Что такое перечисления?
3. Как осуществляется описание структуры?
4. Что такое поле структуры?
5. Как обратится к элементу структуры?
6. К каким типам данных относятся структуры?
7. Как проинициализировать переменную структурного типа?
8. Для чего применяются структуры?
9. Что такое битовые поля?
10. Как обратится к элементу битового поля?

## 4 Задание

1. Написать программу в соответствии с вариантом задания из пункта 5.1.
2. Отладить и протестировать программу.
3. Написать программу в соответствии с вариантом задания из пункта 5.2.
4. Отладить и протестировать программу.
5. Оформить отчёт.

## 5 Варианты заданий

### 5.1 Структуры (Часть 1)

Вариант 1.

Описать структуру с именем STUDENT, содержащую поля:

* фамилия и инициалы;
* номер группы;
* успеваемость (массив из пяти элементов).

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 8 структур типа STUDENT;
* вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4.0;
* если таких студентов нет, вывести соответствующее сообщение.

Вариант 2.

Описать структуру с именем STUDENT, содержащую поля:

* фамилия и инициалы;
* номер группы;
* успеваемость (массив из пяти элементов).

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 6 структур типа STUDENT;
* вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если они имеют оценки 4 и 5;
* если таких студентов нет, вывести соответствующее сообщение.

Вариант 3.

Описать структуру с именем STUDENT, содержащую поля:

* фамилия и инициалы;
* номер группы;
* успеваемость (массив из пяти элементов).

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 9 структур типа STUDENT;
* вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, имеющих хотя бы одну оценку 2;
* если таких студентов нет, вывести соответствующее сообщение.

Вариант 4.

Описать структуру с именем AEROFLOT, содержащую поля:

* название пункта назначения рейса;
* номер рейса;
* тип самолета.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 9 элементов типа AEROFLOT;
* вывод на экран информации о рейсе, номер которого введен с клавиатуры;
* если таких рейсов нет, вывести соответствующее сообщение.

Вариант 5.

Описать структуру с именем AEROFLOT, содержащую поля:

* название пункта назначения рейса;
* номер рейса;
* тип самолета.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 7 элементов типа AEROFLOT;
* вывод на экран пунктов назначения и номеров рейсов, обслуживаемых самолетом, тип которого введен с клавиатуры;
* если таких рейсов нет, вывести соответствующее сообщение.

Вариант 6.

Описать структуру с именем WORKER, содержащую поля:

* фамилия и инициалы работника;
* название занимаемой должности;
* зарплату;
* год поступления на работу.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 5 структур типа WORKER;
* вывод на дисплей фамилий работников, чей стаж работы в организации превышает значение, введенное с клавиатуры;
* если таких работников нет, вывести соответствующее сообщение.

Вариант 7.

Описать структуру с именем TRAIN, содержащую поля:

* название пункта назначения;
* номер поезда;
* время отправления.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 6 структур типа TRAIN;
* вывод на экран информации о поездах, отправляющихся после введенного с клавиатуры времени;
* если таких поездов нет, вывести соответствующее сообщение.

Вариант 8.

Описать структуру с именем TRAIN, содержащую поля:

* название пункта назначения;
* номер поезда;
* время отправления.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 9 структур типа TRAIN;
* вывод на экран информации о пункте назначения, в который отправляется поезд, номер которого введен с клавиатуры;
* если таких поездов нет, вывести соответствующее сообщение.

Вариант 9.

Описать структуру с именем MARSH, содержащую поля:

* название начального пункта маршрута;
* название конечного пункта маршрута;
* номер маршрута.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 7 структур типа MARSH;
* вывод на экран информации о маршруте, номер которого введен с клавиатуры;
* если таких студентов нет, вывести соответствующее сообщение.

Вариант 10.

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 6 структур типа NOTE;
* вывод на экран информации о человеке, номер телефона которого введен с клавиатуры;
* если таких людей нет, вывести соответствующее сообщение.

Вариант 11.

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 месяц, значение которого введено с клавиатуры;
* если таких людей нет, вывести соответствующее сообщение.

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.

Вариант 13.

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 день, значение которого введено с клавиатуры;
* если таких людей нет, вывести соответствующее сообщение.

Вариант 14.

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 5 структур типа NOTE;
* вывод на экран информации о людях, чьи дни рождения совпадают с введенными с клавиатуры;
* если таких людей нет, вывести соответствующее сообщение.

Вариант 15.

Описать структуру с именем ORDER, содержащую поля:

* расчетный счет плательщика;
* расчетный счет получателя;
* перечисляемая сумма в руб.

Написать программу, выполняющую следующие действия:

* ввод с клавиатуры данных в массив, состоящий из 5 структур типа ZNAK;
* вывод на экран информации о сумме, снятой с расчетного счета плательщика, введенного с клавиатуры;
* если таких людей нет, вывести соответствующее сообщение.

### 5.2 Структуры (Часть 2)

1. Описать структуру «дата» (год, месяц, день). Определить функцию «дней до конца месяца» вычисляющую количество дней до конца месяца.
2. Описать структуры для декартовых (x, y) и полярных (r, pi) координат. Определить функцию для перевода из декартовых в полярные координаты.
3. Описать структуру «время» (часы, минуты, секунды). Определить функцию «предыдущая минута» уменьшающая передаваемое в нее время на 1 минуту (учесть, что в сутках 24 часа).
4. Описать структуру «комплексное число» (действительная часть (re), мнимая часть (im)). Определить функции, выполняющие сложение, вычитание и умножение на действительную константу.
5. Описать структуру «дата» (год, месяц, день). Определить функцию «недель с начала года» вычисляющую количество недель с начала года.
6. Описать структуру «время» (часы, минуты, секунды). Определить функцию «прошедшее время» определяющую интервал времени между t1 и t2 в минутах (округление производить в меньшую сторону).
7. Описать структуру «комплексное число» (действительная часть (re), мнимая часть (im)). Определить функцию, выполняющую произведение двух комплексных чисел.
8. Описать структуры для декартовых (x, y) и полярных (r, pi) координат. Определить функцию для перевода из полярных в декартовы координаты.
9. Описать структуру «дата» (год, месяц, день). Определить функцию «дней с начала года» вычисляющую количество дней с начала года.
10. Описать структуру «время» (часы, минуты, секунды). Определить функцию «следующая минута» увеличивающую передаваемое в нее время на 1 секунду (учесть, что в сутках 24 часа).
11. Описать структуру «комплексное число» (действительная часть (re), мнимая часть (im)). Определить функцию, вычисляющую значение квадратного трехчлена a\*x\*x+b\*x+c в комплексной точке x .
12. Описать структуру «дата» (год, месяц, день). Определить функцию «верна ли дата» проверяющую корректность введенной даты.
13. Описать структуру «время» (часы, минуты, секунды). Определить функцию «раньше» для проверки, предшествует ли время t1 времени t2 ( в рамках суток).
14. Описать структуру «комплексное число» (действительная часть (re), мнимая часть (im)). Определить функцию, выполняющую деление двух комплексных чисел.
15. Описать структуру «дата» (год, месяц, день). Определить функцию «дней до конца года» вычисляющую количество дней до конца года.
16. Описать структуру «время» (часы, минуты, секунды). Определить функцию «прошедшее время» определяющую интервал времени между t1 и t2 в секундах.
17. Описать структуру «дата» (год, месяц, день). Определить функцию «дней с начала месяца» вычисляющую количество дней с начала месяца.
18. Описать структуру «комплексное число» (действительная часть (re), мнимая часть (im)). Определить функцию, вычисляющую значение кубического многочлена a\*x\*x\*x +b\*x\*x +c\*x+d в комплексной точке x .
19. Описать структуру «время» (часы, минуты, секунды). Определить функцию «следующая секунда» увеличивающую передаваемое в нее время на 1 секунду (учесть, что в сутках 24 часа).
20. Описать структуру «дата» (год, месяц, день). Определить функцию «день недели» для определения дня недели, на который приходится дата (учитывая, что 1 января 1-го года нашей эры было понедельником).
21. Описать структуру «время» (часы, минуты, секунды). Определить функцию «предыдущая секунда» уменьшающая передаваемое в нее время на 1 секунду (учесть, что в сутках 24 часа).
22. Описать структуру «дата» (год, месяц, день). Определить функцию «дней до конца месяца» вычисляющую количество дней до конца месяца.
23. Описать структуру «время» (часы, минуты, секунды). Определить функцию «позже» для проверки, предшествует ли время t2 времени t1 (в рамках суток).
24. Описать структуру для декартовых координат (x, y). Определить функцию для нахождения расстояния между двумя точками.
25. Описать структуру «дата» (год, месяц, день). Определить функцию «недель до конца года» вычисляющую количество недель до конца года.

*Замечание.*

Полярная система координат — система координат, ставящая в соответствие каждой точке на плоскости пару чисел .



Рисунок 10.1 Полярная система координат

Основными понятиями этой системы являются точка отсчёта (полюс) и луч, начинающийся в этой точке (полярная ось).

Координата определяет расстояние от точки до полюса, координата — угол между полярной осью и отрезком, соединяющим полюс и рассматриваемую точку. Координата берётся со знаком «+», если угол от оси до отрезка вычисляется против часовой стрелки, и со знаком «-» в противоположном случае. Любая точка в этой системе имеет бесконечное число координат вида , которым соответствует одна и та же точка при любых натуральных. Для полюса  = 0, а угол произвольный.

Формулы перехода:

- от полярной системы координат к декартовой:



- от декартовой системы координат к полярной:



## 6 Содержание отчета

1. Титульный лист.
2. Наименование и цель работы.
3. Краткое теоретическое описание.
4. Задание на лабораторную работу.
5. Схема алгоритма.
6. Листинг программы.
7. Результаты выполнения программы.