**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №1**

**по дисциплине «Программирование»**

Тема: Типы данных, определяемые пользователем. Структуры

|  |  |  |
| --- | --- | --- |
| Студент гр.2373 | Карпачёв А.С. |  |
| Преподаватель | Глущенко А. Г. |  |

Санкт-Петербург

2022

**Цель работы.**

Изучение и организация структур; получение практических навыков работы со структурами; определение преимуществ и недостатков использования структур.

**Основные теоретические положения.**

Структуры представляют собой группы связанных между собой, как правило, разнотипных переменных, объединенных в единый объект, в отличие от массива, все элементы которого однотипны. В языке C++ структура является видом класса и обладает всеми его свойствами. Чаще всего ограничиваются тем, как структуры представлены в языке С:

struct [имя\_типа] {

тип\_1 элемент\_1;

тип \_2 элемент\_2;

…

тип\_k элемент\_k;

} [ список\_описателей ];

Описание структуры начинается ключевым словом struct. Каждая входящая в структуру переменная называется членом (полем, элементом) структуры и описывается типом данных и именем. Поля структуры могут быть любого типа данных. Их количество не лимитировано.

Вся эта конструкция является инструкцией языка программирования, поэтому после нее всегда должен ставиться символ ‘;’.

При описании структуры память для размещения данных не выделяется. Работать с описанной структурой можно только после того, как будет определена переменная (переменные) этого типа данных, только при этом компилятор выделит необходимую память.

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

struct complex{

float real, im;

} data [2][2] = {

{{1,1}, {2,2}},

{{3,3}, {4,4}}

};

Все поля структурных переменных располагаются в непрерывной области памяти одно за другим. Общий объем памяти, занимаемый структурой, равен сумме размеров всех полей структуры. Для определения размера структуры следует использовать инструкцию sizeof().

Для того чтобы записать данные в структурную переменную, необходимо каждому полю структуры присвоить определенное значение. Для этого необходимо использовать оператор ‘’ («точка»):

struct Stack { // Cтек

float arr[100];

short topIndex;

};

…

Stack stack; // Объявляем переменную типа Stack

Stack.arr[0] = 1;

…

При доступе к определенному полю его следует рассматривать как обычную переменную, тип данных которой соответствует типу этого поля. Поля структур могут участвовать в качестве операндов любых выражений, допускающих использование операндов соответствующего типа данных.

Копирование данных из одной структурной переменной в другую осуществляется простой операцией присваивания, независимо от количества полей и размера структуры (это можно делать только в том случае, когда обе переменные одного и того же типа).

В программировании очень часто используются такие конструкции, как массивы структур. Например, сведения о студентах некоторой учебной группы можно хранить в массиве студентов:

t\_Student Gruppa\_N [30];

Был определен 30-элементный массив, каждый элемент которого предназначен для хранения данных одного студента. Получение доступа к данным некоторого студента из группы *N* осуществляется обычной индексацией переменной массива. Поскольку поля структуры могут быть любого типа данных, то они в свою очередь могут быть другой структурой или массивом других структур:

struct Stud

{

char FN[100];

short listNumber;

};

struct Group

{

int groupNumber;

short students;

Stud stud[30];

};

Но в структуре поля нельзя использовать элемент, тип которого совпадает с типом самой структуры, так как рекурсивное использование структур запрещено.

Любая структурная переменная занимает в памяти определенное положение, характеризующееся конкретным адресом. Для работы с адресами структурных переменных (как и для простых переменных) можно использовать указатели. Указатели на структурные переменные определяются точно так же, как и для обычных переменных. Разыменование указателя (обращение к данным по адресу, хранящемуся в указателе) осуществляется также обычным образом.

Через указатели можно работать с отдельными полями структур. Для доступа к полю структуры через указатель используется оператор ‘’ («стрелка»), а не «точка».

Структуры можно использовать в качестве параметров функций, как и обычные переменные. Для структур поддерживаются все три механизма передачи данных: по значению, через указатели и по ссылке.

Передачу структур в функции по значению необходимо использовать аккуратно:

void WriteStudent ( t\_Student S )

{

cout << "Фамилия: " << S.Fam << endl;

cout << "Имя: " << S.Name << endl;

cout << "Год рождения: " << S.Year << endl;

if ( S.Sex )

cout << "Пол: " << "М\n";

else

cout << "Пол: " << "Ж\n";

cout << "Средний балл: " << S.Grade << endl;

}

Вызов такой функции сопровождается дополнительным расходом памяти для создания локальной переменной *S*и дополнительными затратами времени на физическое копирование данных из аргумента в параметр *S*. Учитывая то, что объем структур может быть очень большим, эти дополнительные затраты вычислительных ресурсов могут быть чрезмерными.

Предпочтительно использование передачи структуры по указателю или ссылке:

void WriteStudent ( t\_Student \*S )

{

cout << "Фамилия: " << S -> Fam << endl;

cout << "Имя: " << S -> Name << endl;

cout << "Год рождения: " << S -> Year << endl;

if ( S -> Sex )

cout << "Пол: " << "М\n";

else

cout << "Пол: " << "Ж\n";

cout << "Средний балл: " << S -> Grade << endl;

}

Фактической передачи данных в функцию не осуществляется. Дополнительные затраты памяти для создания локальной переменной небольшие – это адрес памяти (4 байта, независимо от размера самой структуры). Вызов такой функции будет происходить быстрее, а расход памяти будет существенно меньше, чем при передаче данных по значению.

Передача по ссылке по эффективности эквивалентна передаче данных через указатель. Однако, поскольку при передаче данных по ссылке все адресные преобразования берет на себя компилятор, существенно упрощается программирование действий со структурами. При использовании ссылочных параметров структурных типов доступ к членам структуры осуществляется обычным способом – с помощью оператора «точка».

Недостатком этих способов является то, что случайные изменения значений полей структуры внутри функции отразятся на значении аргумента после окончания работы функции. Если необходимо предотвратить изменения переданных по адресу аргументов, можно при определении соответствующего параметра объявить его константой (использовать спецификатор const).

**Постановка задачи.**

Разработать алгоритм и написать программу, которая позволяет:

1) Вывести, сколько памяти (в байтах) на вашем компьютере отводится под различные типы данных со спецификаторами и без: int, short int, long int, float, double, long double, char и bool.

2) Вывести на экран двоичное представление в памяти (все разряды) целого числа. При выводе необходимо визуально обозначить знаковый разряд и значащие разряды отступами или цветом.

3) Вывести на экран двоичное представление в памяти (все разряды) типа float. При выводе необходимо визуально обозначить знаковый разряд мантиссы, знаковый разряд порядка (если есть), мантиссу и порядок.

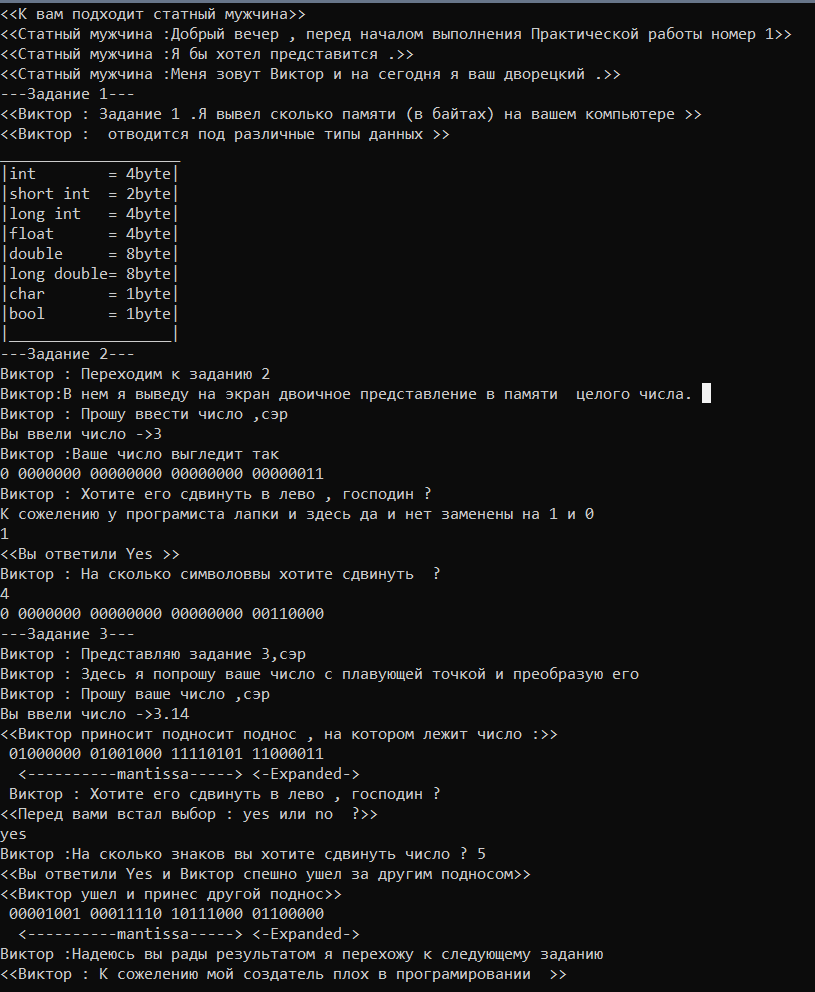
4) Вывести на экран двоичное представление в памяти (все разряды) типа double. При выводе необходимо визуально обозначить знаковый разряд мантиссы, знаковый разряд порядка (если есть), мантиссу и порядок.

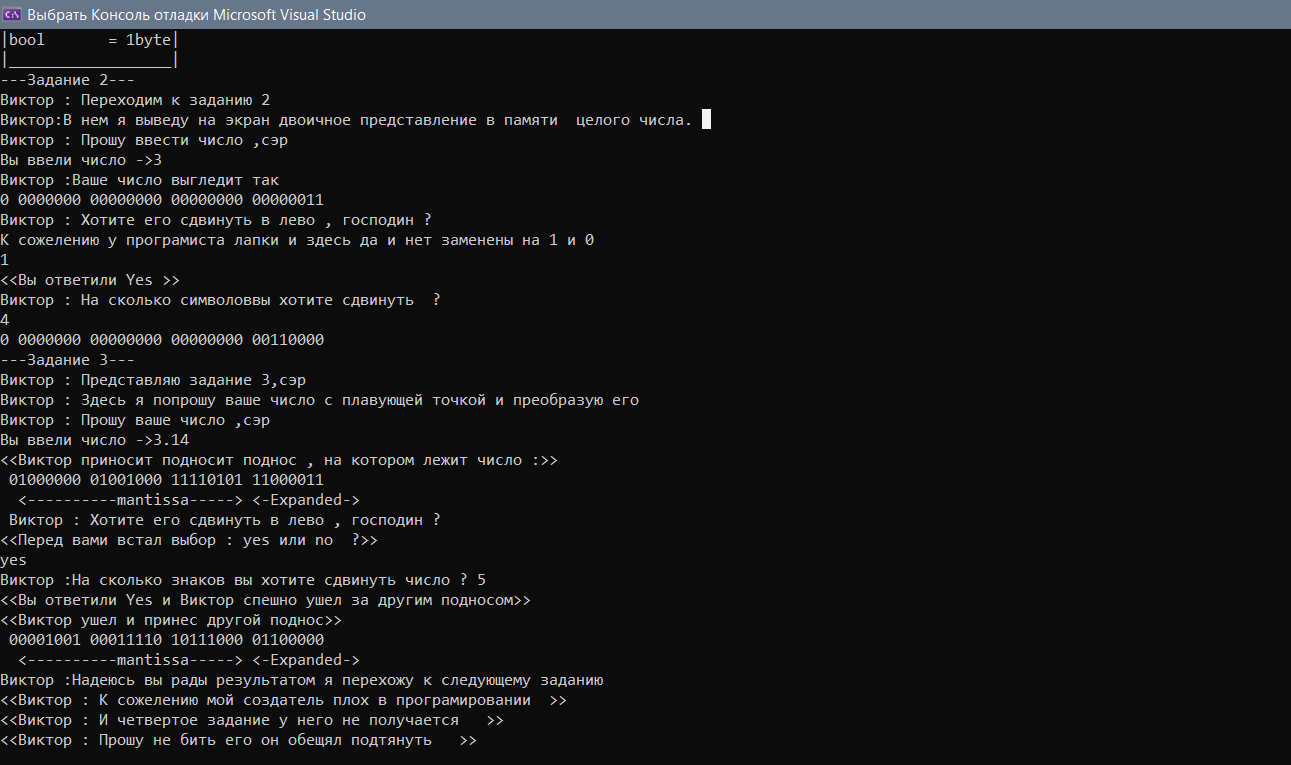
**Выполнение работы.**

Код программы представлен в приложении А.

1. На экран выводится, сколько памяти (в байтах) на компьютере отводится под различные типы данных со спецификаторами и без в таком порядке: int, short int, long int, float, double, long double, char, bool.
2. Программа просит ввести целое число и считывает введенное путём функции cin.
3. После определения маски и количества разрядов, программа проходит через цикл for, сравнивая биты и разделяя их
4. Объединение в памяти переменных формата float и int с помощью union, что поможет обойти ограничение с невозможностью применения операций сдвига к типу float.
5. Программа просит ввести вещественное число и считывает введенное.
6. Программа выводит число, записанное в типе данных float.
7. Программа просит ввести вещественное число и считывает введенное.
8. Программа выводит число, записанное в типе данных double.

**Результаты работы программы.**

1. 

****

**Выводы.**

В этой практической работе изучили типы данных и их внутреннее представления в памяти; получение практических навыков работы с типами данных; определили, каким образом типы данных представляются на компьютере.

Приложение А

рабочий код

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

cout << "<<К вам подходит статный мужчина>> "<< '\n';

cout << "<<Статный мужчина :Добрый вечер , перед началом выполнения Практической работы номер 1>> " << '\n';

cout << "<<Статный мужчина :Я бы хотел представится .>> " << '\n';

cout << "<<Статный мужчина :Меня зовут Виктор и на сегодня я ваш дворецкий .>> " << '\n';

//Задание 1 начало

cout << "---Задание 1---" << '\n';

cout << "<<Виктор : Задание 1 .Я вывел сколько памяти (в байтах) на вашем компьютере >> " << '\n';

cout << "<<Виктор : отводится под различные типы данных >> " << '\n';

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "|"<<"int =" << " " << sizeof(int) << "byte"<<"|" << endl;

cout << "|"<<"short int =" << " " << sizeof(short int) << "byte" << "|" << endl;

cout << "|"<<"long int =" << " " << sizeof(long int) <<"byte" << "|" << endl;

cout << "|"<<"float =" << " " << sizeof(float) <<"byte" << "|" << endl;

cout << "|"<<"double =" << " " << sizeof(double) << "byte" << "|" << endl;

cout << "|"<<"long double=" << " " << sizeof(long double)<< "byte" << "|" << endl;

cout << "|"<<"char =" << " " << sizeof(char) << "byte" << "|" << endl;

cout << "|"<<"bool =" << " " << sizeof(bool) << "byte" << "|" << endl;

cout << "|"<< "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << "| " << endl;

//Задание 1 законченно

{//Задание 2 начало

int value;

setlocale(LC\_ALL, "Russian");

cout << "---Задание 2--- " << '\n';

cout << "Виктор : Переходим к заданию 2 " << '\n';

cout << "Виктор:В нем я выведу на экран двоичное представление в памяти целого числа. " << '\n';

cout << "Виктор : Прошу ввести число ,сэр " << '\n';

cout << "Вы ввели число ->";

cin >> value;

int a = value;

cout << "Виктор :Ваше число выгледит так " << '\n';

unsigned int order1 = 32;

unsigned int mask0 = 1 << order1 - 1;

for (int i = 1; i <= order1; i++)

{

putchar(value & mask0 ? '1' : '0');

value <<= 1;

if (i % 8 == 0)

{

putchar(' ');

}

if (i % order1 - 1 == 0)

{

putchar(' ');

}

}

cout << '\n' << "Виктор : Хотите его сдвинуть в лево , господин ? " << '\n';

int vac ;

cout << "К сожелению у програмиста лапки и здесь да и нет заменены на 1 и 0" << '\n';

cin >> vac ;

if (int(vac) == 1) {

std::cout << "<<Вы ответили Yes >>" << '\n';

cout << "Виктор : На сколько символоввы хотите сдвинуть ? " << '\n';

int art;

cin >> art;

unsigned int order0 = 32;

unsigned int mask1 = 1 << order0 - (1 + art);

for (int i = 1; i <= order0; i++)

{

putchar(a & mask1 ? '1' : '0');

a <<= 1;

if (i % 8 == 0)

{

putchar(' ');

}

if (i % order0 - 1 == 0)

{

putchar(' ');

}

}

}

else if (int(vac) == 0) {

std::cout << "<<Вы ответили No >>" << '\n';

cout << "Виктор :На нет и суда нет" << '\n';

cout << "Виктор :Я перехожу к следующему заданию " << '\n';

}

else {

cout << "Виктор :я вас не понимаю" << '\n';

cout << "Виктор огорченно перешол к следующему заданию " << '\n';

}

}//Задание 2 конец

{

//Задание 3 начало

setlocale(LC\_ALL, "Russian");

cout << '\n' <<"---Задание 3---" << '\n';

cout << "Виктор : Представляю задание 3,сэр" << '\n';

cout << "Виктор : Здесь я попрошу ваше число c плавующей точкой и преобразую его" << '\n';

union {

int tool;

float nunf;

};

setlocale(LC\_ALL, "Russian");

cout << "Виктор : Прошу ваше число ,сэр" << '\n';

cout << "Вы ввели число ->";

cin >> nunf;

cout << "<<Виктор приносит подносит поднос , на котором лежит число :>>" << '\n';

int order = 32; //количество разрядов

int mask = 1 << order - 1;//Маска побитого сравнения

int v = tool;

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

{

if (i % 8 == 0)

cout << (" ");

cout << (tool & mask ? '1' : '0'); //вывод самого числа

tool <<= 1;

}

cout << "\n " << " <----------mantissa-----> <-Expanded->" << "\n ";

setlocale(LC\_ALL, "Russian");

cout << "Виктор : Хотите его сдвинуть в лево , господин ? " << '\n';

string c;

cout << "<<Перед вами встал выбор : yes или no ?>>" << '\n';

setlocale(LC\_ALL, "Russian");

std:cin >> c;

if (string(c) == "Yes" or string(c) == "yes") {

cout << "Виктор :На сколько знаков вы хотите сдвинуть число ? ";

int vol;

cin >> vol;

v <<= vol;

std::cout << "<<Вы ответили Yes и Виктор спешно ушел за другим подносом>>" << '\n';

cout << "<<Виктор ушел и принес другой поднос>> " << '\n';

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

{

if (i % 8 == 0)

cout << (" ");

cout << (v & mask ? '1' : '0'); //вывод самого числа

v <<= 1;

}

cout << "\n " << " <----------mantissa-----> <-Expanded->" << '\n';

cout << "Виктор :Надеюсь вы рады результатом я перехожу к следующему заданию " << '\n';

}

else if (string(c) == "No" or string(c) == "no") {

std::cout << "<<Вы ответили No >>" << '\n';

cout << "Виктор :На нет и суда нет" << '\n';

cout << "Виктор :Я перехожу к следующему заданию " << '\n';

}

else {

cout << "Виктор :я вас не понимаю" << '\n';

cout << "Виктор огорченно перешол к следующему заданию " << '\n';

}

}//Задание 3 законченно

cout << "<<Виктор : К сожелению мой создатель плох в програмировании >> " << '\n';

cout << "<<Виктор : И четвертое задание у него долго не получалось >> " << '\n';

cout << "<<Виктор : Прошу не бить его он обещял подтянуть >> " << '\n';

{

setlocale(LC\_ALL, "Russian");

cout << " Задание 4\n";

union {

double value3;

int b[2];

};

unsigned int order3 = sizeof(double) \* 8;

unsigned int mask3 = 1 << (order3 - 1);

cout << "Виктор : Введите вещественное число: ";

cin >> value3;

for (int l = 0; l < order3 / 2; ++l)

{

if (l == 1 || 1 % 8 == 0 || l == 12)

{

putchar(' ');

}

putchar(b[1] & mask3 ? '1' : '0');

b[1] <<= 1;

}

for (int j = 0; j < order3 / 2; ++j)

{

if (j % 8 == 0)

{

putchar(' ');

}

putchar(b[0] & mask3 ? '1' : '0');

b[0] <<= 1;

}

}

}