**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. 4373 |  | Лакалин С.Н. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2024

**Цель работы.**

Получение объёма и двоичного представления различных типов данных в памяти компьютера.

**Основные теоретические положения.**

Память компьютера можно представить в виде непрерывной последовательности двоичных ячеек, каждая из которых может находиться в двух состояниях условно обозначаемых 0 и 1. Каждая такая двоичная ячейка называется битом. Вся эта последовательность ячеек условно разбита на порции из 8 бит, называемые байтами. Таким образом, 1 байт = 8 битам. Байт является основной единицей измерения объема памяти.

С каждым байтом памяти связано понятие **адреса**, который, по сути, является номером байта в непрерывной последовательности байтов памяти компьютера. То есть каждый байт памяти имеет свой адрес. По этому адресу и осуществляется доступ к данным, хранящимся в памяти.

Программе для правильной обработки этого байта необходимо «знать» что это – число или буква. Другими словами, программе необходимо точно представлять какие данные хранятся в этом байте памяти.

Тип данных для каждого программного объекта, представляющего данные, определяет:

* характер данных (число, со знаком или без знака, целое или с дробной частью, одиночный символ или текст, представляющий последовательность символов и т.д.);
* объем памяти, который занимают в памяти эти данные;
* диапазон или множество возможных значений;
* правила обработки этих данных (например, допустимые операции).

Для описания основных типов определены следующие ключевые слова:

* **int** (целый);
* **float** (вещественный);
* **double** (вещественный тип с двойной точностью);
* **bool** (логический);
* **char** (символьный).

Типы **int**,**bool**и**char**  относят к группе целочисленных (целых) типов, а **float**и**double** - к группе вещественных типов - типов с плавающей точкой. Код, который формирует компилятор для обработки целых величин, отличается от кода для величин с плавающей точкой.

Существует четыре спецификатора типа, уточняющих внутреннее представление и диапазон значений стандартных типов:

* **short**(короткий);
* **long** (длинный);
* **signed** (знаковый);
* **unsigned** (без знаковый).

Для определения размера типа данных следует использовать инструкцию sizeof().

Размер типа **int** не определяется стандартом, а зависит от компьютера и компилятора. Для 16-разрядного процессора под величины этого типа отводится 2 байта – в этом случае диапазон возможных значений составляет -32 768 ... 32 767 (2 в степени 16 различных значений). Для 32-разрядного - 4 байта – диапазон значений -2 147 483 648 ... 2 147 483 647 (2 в степени 32 различных значений).

Спецификатор **short** перед именем типа указывает компилятору, что под число требуется отвести 2 байта независимо от разрядности процессора.

Спецификатор **long** означает, что целая величина будет занимать 4 байта. Таким образом, на 16-разрядном компьютере эквиваленты **int** и **short int**, а на 32-разрядном — **int** и **long int**.

Стандарт C++ определяет три типа данных для хранения вещественных значений: **float**, **double** и **long double**. Все эти типы предназначены для представления отрицательных и положительных значений (спецификатор **unsigned**к ним не применим) в разных диапазонах:

* тип **float**занимает в памяти 4 байта с диапазоном абсолютных значений от 3.4е-38 до 3.4е+38;
* тип **double** занимает в памяти 8 байт с диапазоном абсолютных значений от 1.7е-308 до 1.7е+308;
* тип **long double**занимает в памяти 10 байт с диапазоном абсолютных значений от 3.4e-4932 до 3.4e+4932.

Константы вещественных типов задаются двумя способами:

·         нормальный формат: **123.456** или **-3.14**;

·         экспоненциальный формат: **1.23456e2** (**1.23456е+2**).

Тип **float** занимает 4 байта, из которых один двоичный разряд отводится под знак мантиссы, 8 разрядов под порядок и 23 под мантиссу.

Для величин типа **double**, занимающих 8 байт, под порядок и мантиссу отводится 11 и 52 разряда соответственно. Длина мантиссы определяет точность числа, а длина порядка — его диапазон.

Вывод на экран двоичного представления типа int происходит при помощи побитового и маски и числа.

Изначально маска равна mask = 1 << sizeof(int) \* 8 - 1;

Но затем она каждый раз сдвигается на 1 бит.

Тип Float и Double сами по себе не могут проводить побитовое и. Однако при помощи Union, можно выделить те же ячейки памяти для другой переменной.

union {

float f;

int fi;

};

cin >> f;

Переменная fi типа int будет хранить в себе те же ячейки, что и f типа float. А дальше можно провести побитовое и маски и переменной f.

**Постановка задачи.**

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.

**Выводы:**

Успешно удалось вывести двоичное представление различных типов данных, а также узнать их объем.

Приложение А

рабочий код

#include <iostream>

using namespace std;

int main()

{

// Задание 1

{

cout << sizeof(int) << " "

<< sizeof(short int) << " "

<< sizeof(long int) << " "

<< sizeof(float) << " "

<< sizeof(double) << " "

<< sizeof(long double) << " "

<< sizeof(char) << " "

<< sizeof(bool) << "\n";

}

// Задание 2

setlocale(0, "");

int a;

cin >> a;

unsigned int mask = 1 << sizeof(int) \* 8 - 1;

for (int i = 0; i <= (sizeof(int) \* 8 - 1); i++){

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

mask >>= 1;

}

cout << endl;

// Задание 3

union {

float f;

int fi;

};

cin >> f;

mask = 1 << sizeof(float) \* 8 - 1;

for (int i = 0; i <= (sizeof(float)\*8 - 1); i++)

{

putchar(mask& fi ? '1' : '0');

if (i == 0 or i == 8) {

cout << " ";

}

mask >>= 1;

}

cout << endl;

// Задание 4

union {

double number;

unsigned char bytes[sizeof(double)];

};

cin >> number;

for (int i = sizeof(double) - 1; i >= 0; --i){

mask = 1 << sizeof(char) \* 8 - 1;

for (int j = 0; j <= (sizeof(char) \* 8 - 1); j++)

{

putchar(mask & bytes[i] ? '1' : '0');

if ((i == 6 and j == 3) or (i == 7 and j == 0)){

cout << " ";

}

mask >>= 1;

}

}

cout << endl;

return 0;

}

|  |  |
| --- | --- |
| Ввод с клавиатуры | Работа алгоритма |
|  | |
| Пользователем вводятся 3 числа,  Типа int, типа float и типа double. | Идет вывод размеров типов данных, затем после каждого ввода числа пользователем с клавиатуры идет вывод представления числа в соответствующем типе данных. |