**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 и их внутреннее представление в памяти**

|  |  |  |
| --- | --- | --- |
| Студент гр. 4373 |  | Простакишин Е.А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2024

**Цель работы.**

Изучение типов данных, получение практических навыков работы с памятью и типами данных.

**Основные теоретические положения.**

Основные (предопределенные) типы данных часто называют арифметическими, поскольку их можно использовать в арифметических операциях. 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).

Спецификаторы добавляются слева к названию типа, например, так:

**short int** – короткое целое;

**unsigned short int** - короткое целое без знака.

Спецификаторы могут в произвольном порядке. Например, **unsigned short int** эквивалентно **short unsigned int**.

Описание переменной можно совместить с ее **инициализацией**:

int а = 1213; // переменная a инициализирована значением 1213  
  
double х = 0.003; // переменная х инициализирована значением 0.003

setlocale (0, ""); // русский язык

int a; // инициализация переменной

cout << "Введите целое число:\t"; // вывод текста в консоль

cin >> a; // ввод текста и запись в переменную

Список необходимых операций и выражений

|  |  |
| --- | --- |
| sizeof | размер объекта или типа данных в байтах |
| % | остаток от деления |
| << | сдвиг влево |
| >> | сдвиг вправо |
| < | меньше |
| <= | меньше или равно |
| == | равно |
| != | не равно |
| & | поразрядная конъюнкция (И) |
| || | логическое ИЛИ |
| <<= | сдвиг влево с присваиванием |
| >>= | сдвиг вправо с присваиванием |
| ?: | условная операция(тернарная) |

Внутреннее представление величин целого типа – целое число в двоичном коде. При использовании спецификатора signed старший бит числа интерпретируется как знаковый (0 – положительное число, 1 – отрицательное). Для кодирования целых чисел со знаком применяется прямой, обратный и дополнительный коды.

Представление положительных и отрицательных чисел в прямом, обратном и дополнительном кодах отличается. В прямом коде в знаковый разряд помещается цифра 1, а в разряды цифровой части числа – двоичный код его абсолютной величины.



Обратный код получается инвертированием всех цифр двоичного кода абсолютной величины, включая разряд знака: нули заменяются единицами, единицы – нулями. Прямой код можно преобразовать в обратный, инвертировав все значения всех битов (кроме знакового). Обратный код числа −3:



Дополнительный код получается образованием обратного кода с последующим прибавлением единицы к его младшему разряду. Дополнительный код числа −3:



Увидеть, каким образом тип данных представляется на компьютере, можно при помощи логических операций: побитового сдвига (<<) и поразрядной конъюнкции (&).

putchar(value & mask ? '1' : '0'); // если 1, то возвращается 1, иначе 0  
value <<= 1; // побитовый сдвиг влево на 1 бит

**Putchar** возвращает один символ в консоль. Альтернатива - **cout**. В представленном способе, маска - то, с чем сравнивается значение. И побитовый сдвиг применяется для **value.** Таким образом 1 бит будет сравниваться с каждым битом числа. Альтернатива - побитовый сдвиг вправо, но при этом нужно проводить данную операцию не над значением(единицей), а над маской (исходым числом, битовое представление которого нужно получить).

При сдвиге вправо для чисел без знака позиции битов, освобожденные при операции сдвига, заполняются нулями. Для чисел со знаком бит знака используется для заполнения освобожденных позиций битов. Другими словами, если число 25 является положительным, используется 0, если число является отрицательным, используется 1. При сдвиге влево позиции битов, освобожденных при операции сдвига, заполняются нулями. Сдвиг влево является логическим сдвигом (биты, сдвигаемые с конца, отбрасываются, включая бит знака).

Вещественные типы данных хранятся в памяти компьютера иначе, чем целочисленные. Внутреннее представление вещественного числа состоит из двух частей – мантиссы и порядка.

Для 32-разрядного процессора для float под мантиссу отводится 23 бита, под экспоненту – 8, под знак – 1. Для double под мантиссу отводится 52 бита, под экспоненту – 11, под знак – 1.

**Объединения** – это две или более переменных расположенных по одному адресу (они разделяют одну и ту же память). Объединения определяются с использованием ключевого слова union. Объединения не могут хранить одновременно несколько различных значений, они позволяют интерпретировать несколькими различными способами содержимое одной и той же области памяти.

**Постановка задачи.**

Необходимо создать программу для работы с типами данных и их представлением во внутренней памяти.

Написать функции, реализующие операции со структурами (ввод данных с клавиатуры):

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.

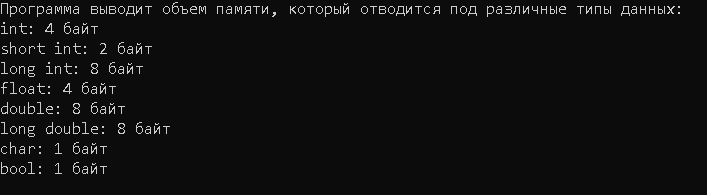
3) Вывести на экран двоичное представление в памяти (все разряды) типа float. При выводе необходимо визуально обозначить знаковый разряд мантиссы, знаковый разряд порядка (если есть), мантиссу и порядок.

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.**

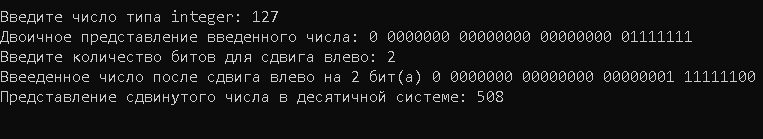
***Задание №1***

Код программ представлен в приложении A.

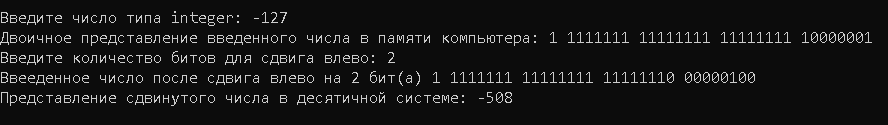
Вывод количества памяти (в байтах) под различные типы данных:

*Программа 1*

***Задание №2***

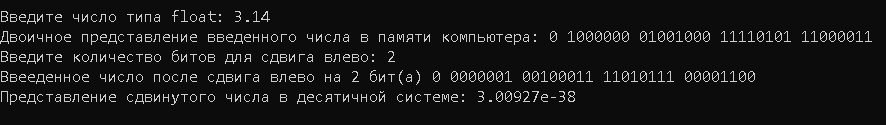
Ввод с клавиатуры целого числа типа integer и вывод его двоичного представления.

*Программа 2.1 (ввод положительного числа)*

В программе реализовано ИДЗ № 11. Пользователь может ввести количество битов для сдвига влево.

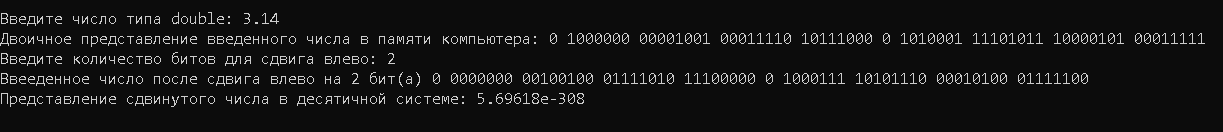
*Программа 2.2 (ввод отрицательного числа)*

***Задание №3***

Ввод с клавиатуры вещественного числа типа float и вывод его двоичного представления.

*Программа 3*

В программе так же реализовано ИДЗ №11.

 ***Задание №4***

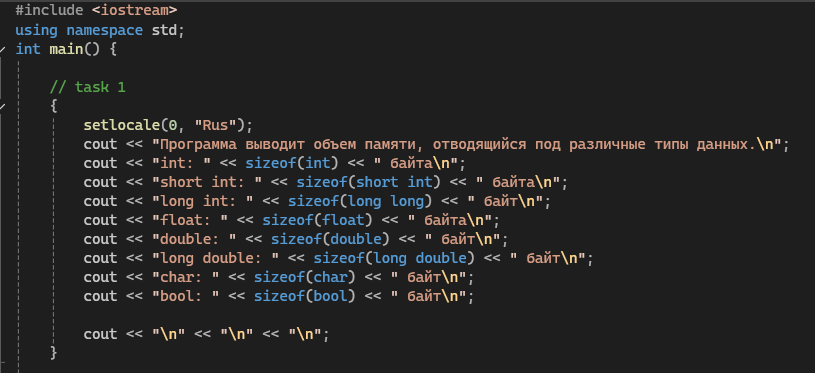
*Программа 4*

В программе так же реализовано ИДЗ № 11.

**Выводы.**

Изучив типы данных, алгоритм работы памяти компьютера и основы языка C++, я разработал 4 программы, 3 из которых позволяют просматривать, как выглядит число, относящееся к тому или иному типу данных, в памяти компьютера. Первая же программа несет образовательный характер и несет в себе информацию о том, сколько байт памяти занимает тот или иной тип данныйх.

Приложение А

рабочий код





