**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. 3372 |  | Веселов А.С. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

Знакомство с внутренним представлением различных типов данных, используемых компьютером при их обработке.

**Основные теоретические положения.**

При программировании на языке С++ используются 11 стандартных типов данных. Среди них можно выделить 3 группы:

1. Данные символьные и целого типа беззнаковые (с фиксированной запятой) ;

2. Данные символьные и целого типа со знаком (с фиксированной запятой), значения которых хранятся в двоичном дополнительном коде;

3. Данные вещественного типа (с плавающей запятой (точкой)). Вещественное число в памяти хранится с нормализованной мантиссой, значение которой в десятичном эквиваленте лежит в диапазоне от 1 до 2. Причём 2 не входит в границу диапазона. Если в процессе выполнения какой-­либо операции над данными с плавающей запятой значение мантиссы выходит из указанного диапазона, то в конце операции выполняется нормализация результата путем приведения значения мантиссы к указанному диапазону с соответствующим изменением значения порядка. Если значение мантиссы равно нулю или в процессе выполнения операции значение порядка становится меньше допустимой величины, то в результате выполнения операции сформируется так называемый «машинный ноль», то есть код, у которого значение всех бит равно нулю. Но если мантисса всегда нормализована, то старший её бит, то есть единицу, можно и не хранить в памяти. Стандартом предложено это бит не хранить в памяти и тем самым увеличить точность представления вещественных чисел в 2 раза. Эта единица присутствует неявно, то есть, скрыта от глаз наблюдателя и называется неявной единицей (implicit one). Отбрасывание старшей цифры мантиссы выполняется для форматов float и double, но не выполняется для long double. Порядок числа в соответствии с указанным форматом хранится сдвинутым», то есть к его действительному значению добавляется в зависимости от формата такое число, чтобы порядок Р был всегда неотрицательным. Для формата float прибавляется 127, для чисел формата double прибавляется 1023, а для формата long double добавляется 16383. Всегда неотрицательный порядок упрощает выполнение операции сравнения порядков и арифметических операций над ними, а также избавляет от необходимости выделять один бит для хранения знака порядка.

**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.

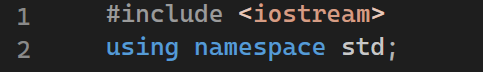
2) Вывести на экран двоичное представление в памяти (все разряды) целого числа. При выводе необходимо визуально обозначить знаковый разряд и значащие разряды отступами или цветом.

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.

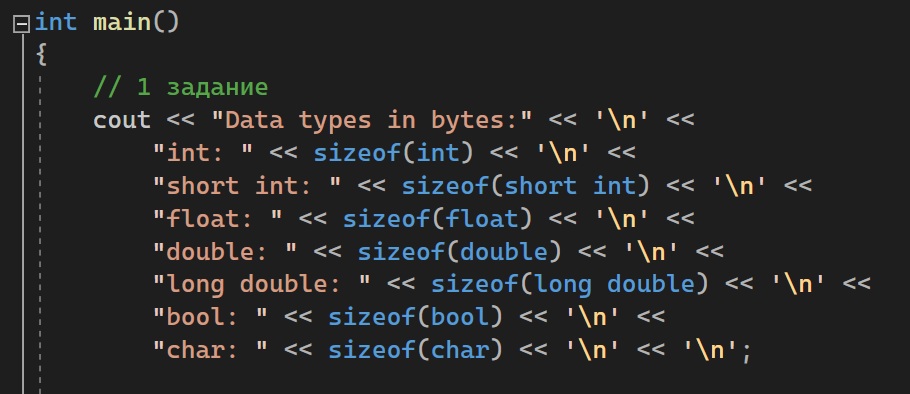
4) Вывести на экран двоичное представление в памяти (все разряды) типа double. При выводе необходимо визуально обозначить знаковый разряд мантиссы, знаковый разряд порядка (если есть), мантиссу и порядок. (\*)

**Выполнение работы.**

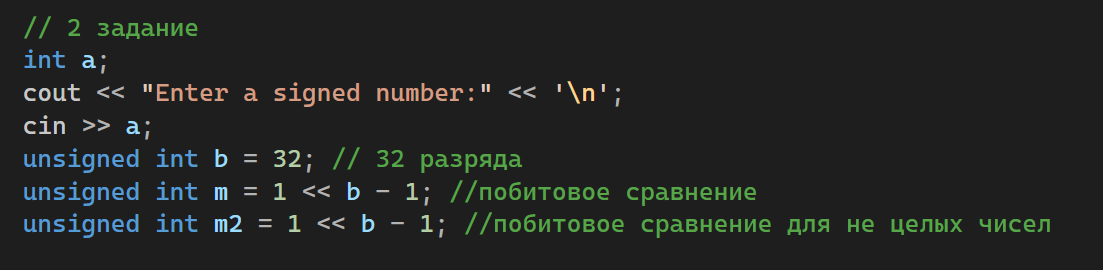
1. Вводим начальную характеристику:



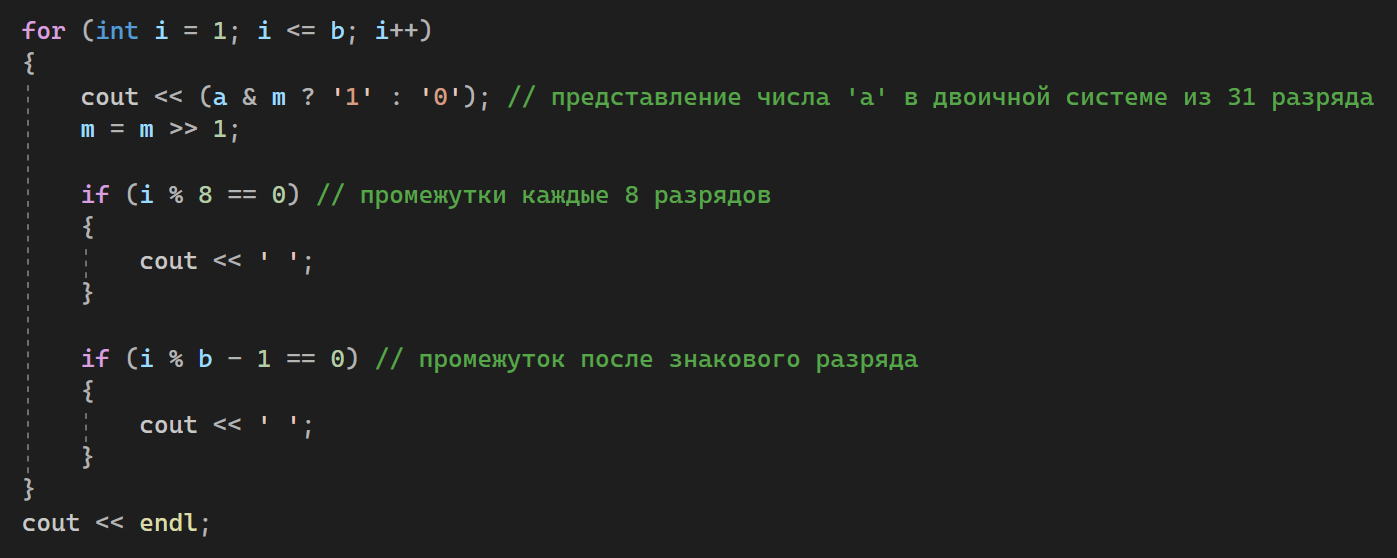
1. Выполняем первое задание, используя тип sizeof, который обозначает размер типа данных в байтах. Через cout выводим значения на экран:



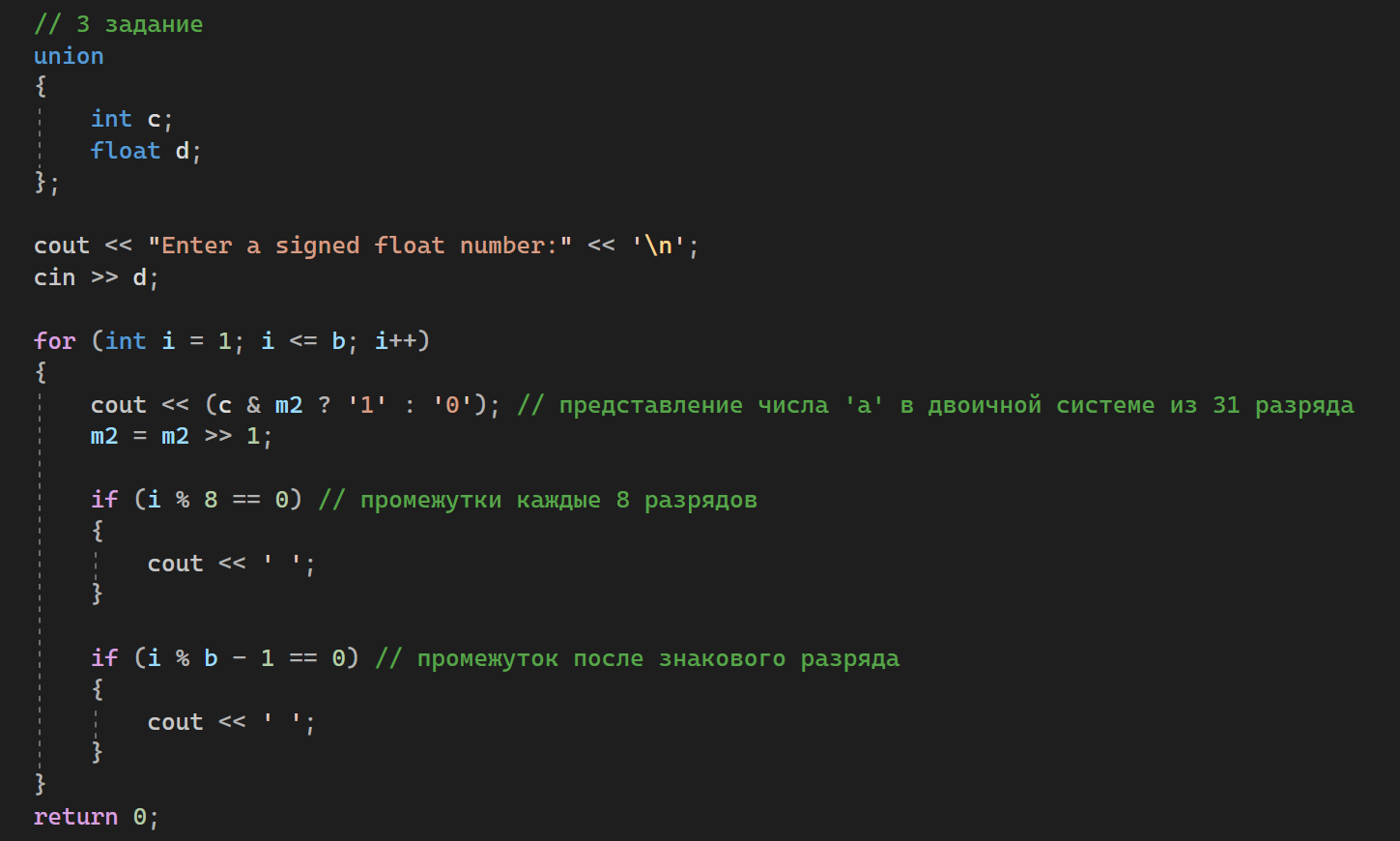
1. Вводим переменные целого числа, количества разрядов двоичного кода, маску, с помощью которой на экран будут выводиться «1» и «0»:



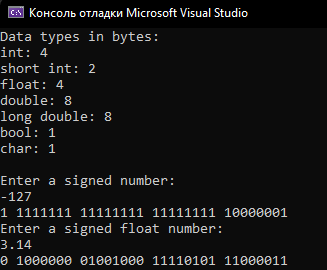
1. Задаем цикл исходя из количества разрядов, и тип cout, который будет выводить на экран число, представленное в двоичной системе исчисления. Далее добавляем промежутки каждые 8 разрядов обозначающие 8 битов в объединении 1 байт, и промежуток между знаковым разрядом:



1. Выполняем туже функцию, только используем union для объединения типа int и float, так как программа изначально не переводит числа типа float в двоичное представление:



1. В итоге получаем:



**Выводы.**

В ходе работы мы научились представлять разные числа в двоичной системе, так как они выглядят в памяти. Как выглядят в памяти разные типы данных в байтах.

Приложение А

рабочий код

#include <iostream>

using namespace std;

int main()

{

// 1 задание

cout << "Data types in bytes:" << '\n' <<

"int: " << sizeof(int) << '\n' <<

"short int: " << sizeof(short int) << '\n' <<

"float: " << sizeof(float) << '\n' <<

"double: " << sizeof(double) << '\n' <<

"long double: " << sizeof(long double) << '\n' <<

"bool: " << sizeof(bool) << '\n' <<

"char: " << sizeof(char) << '\n' << '\n';

// 2 задание

int a;

cout << "Enter a signed number:" << '\n';

cin >> a;

unsigned int b = 32; // 32 разряда

unsigned int m = 1 << b - 1; //побитовое сравнение

unsigned int m2 = 1 << b - 1; //побитовое сравнение для не целых чисел

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

{

cout << (a & m ? '1' : '0'); // представление числа 'a' в двоичной системе из 31 разряда

m = m >> 1;

if (i % 8 == 0) // промежутки каждые 8 разрядов

{

cout << ' ';

}

if (i % b - 1 == 0) // промежуток после знакового разряда

{

cout << ' ';

}

}

cout << endl;

// 3 задание

union

{

int c;

float d;

};

cout << "Enter a signed float number:" << '\n';

cin >> d;

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

{

cout << (c & m2 ? '1' : '0'); // представление числа 'a' в двоичной системе из 31 разряда

m2 = m2 >> 1;

if (i % 8 == 0) // промежутки каждые 8 разрядов

{

cout << ' ';

}

if (i % b - 1 == 0) // промежуток после знакового разряда

{

cout << ' ';

}

}

return 0;

}

Условный пример содержания пункта «Выполеннеие работы»

|  |  |
| --- | --- |
| Ввод пользователем и обработка данных | Работа алгоритма и вывод на экран |
| Меню | |
| При запуске программы перед пользователем появляется окно с меню, где он может выбрать тип будущей вводимой последовательности | Меню:    Проверка на ввод символов, которые не входят в диапазон выбора: |

Продолжение Таблицы

|  |  |
| --- | --- |
| Создание новой записи | |
| При вводе пользователем корректного значения пункта меню и выбора создания новой записи, пользователь может добавить данные о студенте, но только на русском языке. | Для начала идёт проверка на правильный ввод пользователем:    Как только все данные заданы, пользователь может еще раз просмотреть все данные о студенте и в случае, если видит ошибку, сразу же ее исправить без дополнительного вызова меню: |

Продолжение Таблицы

|  |  |
| --- | --- |
| Внесение изменений в имеющуюся запись | |
| Если пользователь допустил ошибку в создании карточки студента или данные изменились, он может внести изменения по пунктам, которые были созданы ранее. Это второе меню, с которым встречается пользователь. | При необходимости внесения изменений в запись о студенте, перед пользователем появляется меню:    Как только пользователь вводит корректное значение (в данном случае от 1 до 10), его ответ обрабатывается и предлагается изменить существующие данные на новые. После пользователь возвращается (при необходимости) в главное меню. |

Продолжение Таблицы

|  |  |
| --- | --- |
| Информация о студентах в количестве | |
| Если пользователь хочет классифицировать студентов по полу, по группам, по номеру в списке, то он может вывести список о всех студентах | Пример, когда пользователь выбрал список данных по конкретной группе, ввел номер группы, а студентов в этой группе пока что не имеется:    Пример, когда пользователь выбрал студентов по их номеру в списке соответственно для двух групп: |

Продолжение Таблицы

|  |  |
| --- | --- |
| Рейтинг студентов | |
| Пользователь в главном меню может выбрать такую опцию как рейтинг студентов, чтобы наглядно увидеть список студентов с минимальными данными и рейтингу их оценок. | Выводится номер в списке по группам, ФИО, сама группа и рейтинг. Рейтинг складывается из всех оценок за сессию (диф зачеты экзамены):    В данном примере студентов в базе было занесено один, то есть, информация и средний бал вывелся только по одному человеку. |

Продолжение Таблицы

|  |  |
| --- | --- |
| Стипендия | |
| Стипендию пользователь может узнать в двух случаях: он может выбрать непосредственно пункт в меню, где ему выйдет список студентов, которые будут получать стипендию. Или пользователь может выбрать список с данными о студентах, которые могли бы получать стипендию, то есть они учатся не на очной форме обучения при оценках не ниже хорошо. | Пример вывода данных о студенте, который не будет получать стипендию при положительных оценках: |

Окончание Таблицы

|  |  |
| --- | --- |
| Очередь в общежитие | |
| Пользователь может узнать очередь в общежитие, занося данные о место происхождении студента и о его заработке. | Пример очереди из студентов, которые получают разных доход (в тыс. руб.), являются и иногородними, и местными:  ../Desktop/очередь.png |
| Полный список о студентах | |
| Если пользователь не собирается получать данные о студентах по конкретному критерию, он может вывести полный список студентов в консоли или же получить отдельный файл. | Вывод в консоли:  ../Desktop/Снимок%20экрана%202020-02-29%20в%2010.22.33.png  Вывод в файл:  ../Desktop/Снимок%20экрана%202020-02-29%20в%2010.22.07.png |