**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. 2372 |  | Сарайкин А. А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2021

**Цель работы.**

Понять, как числа разных типов представляются в памяти компьютера.

Написать программу, визуализирующую представление чисел в памяти компьютера.

**Основные теоретические положения.**

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). Дополнительный код получается образованием обратного кода с последующим прибавлением единицы к его младшему разряду.

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).

**Постановка задачи.**

Разбить программу на несколько блоков, выполняющих разные функции.

**Первый блок** программы выводит количество байт, которое выделяет компьютер на хранение переменных разных типов.

**Второй блок** выводит двоичное представление числа типа «int» с визуальным отображением каждого байта и знакового бита в памяти компьютера.

**Третий блок** выводит двоичное представление числа типа «float» с визуальным отображением знакового бита, экспоненты и мантиссы в памяти компьютера.

**Четвертый блок** выводит двоичное представление числа типа «double» с визуальным отображением знакового бита, экспоненты и мантиссы в памяти компьютера.

Для удобства надо зациклить программу (с возможностью выхода из цикла) на 2, 3 и 4 блоках .

**Выполнение работы.**

Код программы представлен в приложении А.

1. При запуске программы пользователь получает объемы памяти, необходимые для хранения чисел в определенных типах.
2. Пользователю предлагается ввести целое число для дальнейшего отображения двоичного представления данного числа в памяти компьютера. (есть возможность сразу перейти к пункту 3, введя любой символ с клавиатуры)
3. Пользователю предлагается ввести дробное число для дальнейшего отображения двоичного представления данного числа типа «float» в памяти компьютера. (есть возможность сразу перейти к пункту 4, введя любой символ с клавиатуры)
4. Пользователю предлагается ввести дробное число для дальнейшего отображения двоичного представления данного числа типа «double» в памяти компьютера. (есть возможность сразу перейти к пункту 3, введя любой символ с клавиатуры)

**Выводы.**

**Код написан, и работает корректно.**

Приложение А

рабочий код

#include <iostream>  
using namespace std;  
int main()  
{  
 cout << "int" << " " << sizeof(int) << endl << "short int" << " " << sizeof(short int) << endl << "long int" << " " << sizeof(long int) << endl  
 << "float" << " " << sizeof(float) << endl << "double" << " " << sizeof(double) << endl << "long double" << " " << sizeof(long double) << endl  
 << "char" << " " << sizeof(char) << endl << "bool" << " " << sizeof(bool) << endl;  
 bool g = true;  
 while (g == 1)  
 {  
 int mask , value\_1, count = 2;  
 cout << "enter a number to get a binary representation in memory of the \"int\" type" << endl ;  
 cin >> value\_1;  
 mask = 1 << (sizeof(int) \* 8 - 1);  
 if (cin)  
 {  
 for (int i = 0; i < sizeof(int) \* 8; i++ )  
 {  
 if (i == 1 || i % 8 == 0 && i != 0)  
 {  
 putchar(' ');  
 }  
 putchar(value\_1 & mask ? '1' : '0');  
 mask = 1 << (sizeof(int) \* 8 - count) ;  
 count ++;  
 }  
 cout << endl;  
 }  
 else  
 {  
 g = false;  
 }  
 }  
 union  
 {  
 float value\_2\_1;  
 int value\_2;  
 };  
 g = true;  
 cin.clear();  
 while(cin.get() != '\n');  
 while (g == 1)  
 {  
 int mask, count = 2;  
 cout << "enter a number to get a binary representation in memory of the \"float\" type" << endl;  
 cin >> value\_2\_1 ;  
 mask = 1 << (sizeof(float) \* 8 - 1);  
 if (cin)  
 {  
 for (int i = 0; i < sizeof(float) \* 8; i++)  
 {  
 if (i == 1 || i == 9)  
 {  
 putchar(' ');  
 }  
 putchar(value\_2 & mask ? '1' : '0');  
 mask = 1 << (sizeof(float) \* 8 - count);  
 count++;  
 }  
 cout << endl;  
 }  
 else  
 {  
 g = false;  
 }  
 }  
 union  
 {  
 double value\_3\_1;  
 int value\_3[2] ;  
 };  
 g = true;  
 cin.clear();  
 while(cin.get() != '\n');  
 while (g == 1)  
 {  
 int count = 2;  
 int mask ;  
 cout << "enter a number to get a binary representation in memory of the \"double\" type" << endl;  
 cin >> value\_3\_1 ;  
 mask = 1 << (sizeof(int) \* 8 - 1);  
 if (cin)  
 {  
 for (int j = 1; j >= 0 ; j--)  
 {  
 for (int i = 0; i < sizeof(float) \* 8; i++)  
 {  
 if ((i == 1 && j == 1) || (i == 12 && j == 1))  
 {  
 putchar(' ');  
 }  
 putchar(value\_3[j] & mask ? '1' : '0');  
 mask = 1 << (sizeof(float) \* 16 - count);  
 count++;  
 }  
 }  
 cout << endl;  
 }  
 else  
 {  
 g = false;  
 }  
 }  
 return 0;  
 }