**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 |  | Кипень В. А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

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. (\*)

**Основные теоретические положения.**

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. Прямой код числа −3 (для 16- разрядного процессора):



Рисунок 1 – Представление числа 3 в прямом коде

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:



Рисунок 2 – Представление числа -3 в обратном коде

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



Рисунок 3 – Представление числа -3 в дополнительном коде

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

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.**

Необходимо 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.**

|  |  |
| --- | --- |
| 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 пункт, который требуется выполнить | Меню:    Проверка на ввод символов, которые не входят в диапазон выбора: |

Продолжение Таблицы

|  |  |
| --- | --- |
| Двоичное представление числа | |
| При вводе пользователем корректного значения пункта меню и корректного значения числа в пункте, на экран выводится его двоичное представление. | Как только все данные заданы, на экран выводится двоичное представление введённого десятичного числа, а также появляется возможность завершить или продолжить выполнение программы. |

**Выводы.**

1) Выведены значения памяти (в байтах), отведенные под различные типы данных со спецификаторами и без:

- int: 4 байта

- short int: 2 байта

- long int: 4 байта

- float: 4 байта

- double: 8 байт

- long double: 16 байт

- char: 1 байт

- bool: 1 байт

2) Выведено двоичное представление целого числа с обозначением знакового разряда и значащих разрядов. Обозначены разряды отступами или цветом.

3) Выведено двоичное представление числа типа float с обозначением знакового разряда мантиссы, знакового разряда порядка, мантиссы и порядка.

4) Выведено двоичное представление числа типа double с обозначением знакового разряда мантиссы, знакового разряда порядка, мантиссы и порядка.

Приложение А

рабочий код

#include <iostream>

using namespace std;

int main () {

    setlocale(0, "");

*// Базовые переменные для всех типов данных*

    short int order = sizeof(int) \* 8 - 1;

    unsigned int mask = 1 << order;

    while(true) {

        cout << "Выберите пункт работы (число от 1 до 4): " << "\n";

        short int workPoint;

        cin >> workPoint;

        switch (workPoint)

        {

            case 1: {

                cout <<

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

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

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

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

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

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

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

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

                break;

            }

            case 2: {

                int number;

                cout << "\n" << "Введите целое число: ";

                cin >> number;

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

                    cout << ((number & mask) ? 1: 0);

                    mask >>= 1;

                    if (!i) {

                        cout << " ";

                    }

                    if ((i + 1) % 8 == 0) {

                        cout << " ";

                    }

                }

                break;

            }

            case 3: {

                union {

                    int number;

                    float floatNumber;

                };

                cout << "\n" << "Введите вещественное число: ";

                cin >> floatNumber;

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

                    cout << ((number & mask) ? 1: 0);

                    mask >>= 1;

                    if (!i) {

                        cout << " ";

                    }

                    if (i == 8) {

                        cout << " ";

                    }

                }

                break;

            }

            case 4: {

                union {

                    int numberArray [2] = {};

                    double doubleNumber;

                };

                cout << "\n" << "Введите вещественное число двойной точности: ";

                cin >> doubleNumber;

*// Меняем элементы массива местами для корректного двоичного представления*

                int tempElement = numberArray[1];

                numberArray[0] = numberArray[1];

                numberArray[1] = tempElement;

                for (int j = 0; j < 2; j++) {

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

                        cout << ((numberArray[j] & mask) ? 1: 0);

                        mask >>= 1;

                        if (!j && !i) {

                            cout << " ";

                        }

                        if (!j && i == 11) {

                            cout << " ";

                        }

                    }

                }

                break;

            }

            default: {

                cout << "\n" << "Вы ввели не число в диапазоне от 1 до 4";

                break;

            }

        }

        cin.clear(); *// Очищаем поток ввода от возможных ошибок*

        cout << "\n" << "Продолжить работу программы? (Y/N) ";

        char stopFlag;

        cin >> stopFlag;

        if (stopFlag != 'Y' && stopFlag != 'y') {

            break;

        }

    }

    return 0;

}