**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. 5376 |  | Фуфачев Г. К. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2025

**Цель работы.**

Закрепить знания о типах данных в языке программирования и их внутреннем представлении в памяти компьютера, определить размеры различных типов данных, анализировать двоичное представление чисел различных форматов и отработать навыки работы с битами с использованием логических операций.

**Основные теоретические положения.**

Типы данных – это множество значений и операций над ними, определяющих способ хранения и обработки информации в программе. Каждый тип данных характеризуется:

* диапазоном возможных значений;
* объемом памяти, отводимым для хранения;
* правилами интерпретации битов в памяти.

Целые типы данных (int, short int, long int, char, bool) хранятся в виде двоичного кода. Наиболее распространённый способ представления отрицательных чисел – дополнительный код, при котором старший бит интерпретируется как знаковый разряд.

Числа с плавающей точкой (float, double, long double) Внутреннее представление вещественного числа состоит из двух частей – мантиссы и порядка:

* знаковый бит – определяет знак числа;
* порядок – задаёт степень числа с основанием 2
* мантисса – хранит значащие цифры числа.

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:

Размер типов данных зависит от разрядности архитектуры (32-битная или 64-битная система) и особенностей компилятора. Например, в 64-битных системах int обычно занимает 4 байта, long int – 8 байт, float – 4 байта, double – 8 байт.

Операции над битами (логические: AND, OR, XOR, NOT, а также операции сдвига) позволяют управлять отдельными битами числа: проверять, устанавливать, сбрасывать или инвертировать их значения. Это используется для изменения двоичного представления чисел.

В ходе выполнения практической работы потребовалось не только определить размеры различных типов данных, но и получить их внутреннее представление в памяти, а также реализовать универсальные функции для работы с разными типами чисел и организовать проверку корректности пользовательского ввода.

Для этого в программе были использованы следующие конструкции языка C++:

1. Union

Union в C++ - специальный тип данных, который позволяет нескольким переменным различных типов занимать один участок памяти. В данной работе union используется, чтобы интерпретировать число типа float или double как набор байтов (int или long long), и затем просматривать его внутреннее двоичное представление.

1. Template

Шаблоны позволяют писать универсальные функции, работающие с разными типами данных (int, float, double и т. д.) без дублирования кода.

1. cin.clear() и cin.ignore(...)

Используется для обработки ошибок ввода.

* cin.clear() - сбрасывает флаг ошибки, если пользователь ввёл некорректные данные (например, букву вместо числа).
* cin.ignore(numeric\_limits<streamsize>::max(),'\n') - очищает «очередь» ввода, удаляя неверные символы до конца строки.

**Постановка задачи.**

1. Определить и вывести на экран количество памяти (в байтах), занимаемое различными типами данных: int, short int, long int, float, double, long double, char, bool.
2. Реализовать функцию, которая отображает двоичное представление целого числа, выделяя знаковый разряд и значащие разряды.
3. Реализовать функцию, которая отображает двоичное представление числа типа float с указанием полей: знак, порядок, мантисса.
4. Реализовать функцию, которая отображает двоичное представление числа типа double с указанием полей: знак, порядок, мантисса.
5. Реализовать механизм изменения произвольного бита числа (для всех рассматриваемых типов данных) с использованием логических операций.
6. Обеспечить возможность неоднократного использования вышеуказанных функций для работы с различными входными данными (через организацию повторяющегося цикла и меню выбора).

**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 необходимую операцию | | Меню:    Проверка на корректность ввода пункта меню: |
| Вывести объем памяти, отведенный под различные типы данных. | | |
| При вводе пользователем корректного значения пункта меню на экран выведется информация о объеме памяти под различные типы данных | Вывод информации о том, сколько памяти (в байтах) на вашем компьютере отводится под различные типы данных со спецификаторами и без: | |

|  |  |
| --- | --- |
| Вывод представления целого числа в памяти компьютера. | |
| При вводе пользователем корректного значения пункта меню пользователь должен будет ввести целое число, затем выведется его представление в памяти. | После ввода пользователем целого числа на экран выведется представление числа в памяти компьютера по байтам. |
| Вывод представления вещественного числа тип float в памяти компьютера. | |
| При вводе пользователем корректного значения пункта меню пользователь должен будет ввести вещественное число float, затем выведется его представление в памяти. | После ввода пользователем вещественного числа float на экран выведется представление числа в памяти компьютера с разделением на порядок и мантиссу. |

|  |  |
| --- | --- |
| Вывод представления вещественного числа тип double в памяти компьютера. | |
| При вводе пользователем корректного значения пункта меню пользователь должен будет ввести вещественное число double, затем выведется его представление в памяти. | После ввода пользователем вещественного числа double на экран выведется представление числа в памяти компьютера с разделением на порядок и мантиссу. |
| Замена произвольного бита в числа, введенном пользователем. | |
| После вывода представления числа в памяти компьютера пользователю предлагается заменить бит в числе, затем пользователя просят ввести порядковый номер бита и на что его заменить (1 или 0)  В конце выводится новое число и его представление в памяти. | После ввода пользователем всех данных происходит замена бита числа и на экран выводится новое число и его представление в памяти. |

**Выводы.**

В ходе выполнения практической работы была разработана программа на языке **C++**, которая позволяет исследовать внутреннее представление различных типов данных в памяти компьютера.