**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. 3373 |  | Буряков П. С. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

Познакомиться с основными типами данных в языке C++. На практике изучить их внутреннее представление в памяти.

**Основные теоретические положения.**

1. Целочисленные. Виды данных в программировании делятся на знаковые и беззнаковые. Уже понятно из наименования: в знаковых могут храниться все действительные числа, а также ноль, а в беззнаковых – только положительные (больше нуля).У беззнаковых данных диапазон больше в 2 раза, чем у знаковых. Это – из-за компьютерного восприятия: в знаковых типах бит отражает знак числа, где 0 является положительным значением, а 1 – отрицательным. Тип short (короткий целый.) Для него в памяти отведено 16 бит, то есть 2 байта (216 = 65 536). Диапазон значений, который может принять тип short со знаком – это [-32 768; 32 767]. Переменный тип long (длинный целый). Этому типу выделено 64 бита, то есть 8 байт. (264 = 1,8 446 744 \* 1 019). Он имеет внушительный диапазон: в случае знакового типа это [-9 223 372 036 854 775 808 9 223 372 036 854 775 807]. Также модификатор long может использоваться в связке с другими типами (long будет указан перед наименованием типа, допустим, long double). Благодаря этому увеличивается диапазон возможных значений.
2. Вещественные. Значения этого типа имеют плавающую запятую. Плавающая запятая — форма представления действительных чисел, в которой число хранится в форме мантиссы и показателя степени. Если говорить на языке программирования, то каждое число может быть представлено в следующей форме: N = M ∗ 10p, где N — записываемое число; M — мантисса; p (целое) — порядок. float —с плавающей запятой; double —с плавающей запятой двойной точности.
3. Символьный тип данных в программировании. В символьном типе переменная имеет только один символ, целое число. В соответствии с кодировкой, он преобразуется в некий символ. Символьному виду данных в программировании присущ только размер выделяемой под них памяти.
4. Логический тип данных в программировании. У этого типа данных могут быть следующие значения: false (ложь) или true (правда).
5. Массивы. Массив – это последовательно выстроенная и имеющая общее имя структура данных, в которой хранятся элементы одного типа. Его можно представить как набор пронумерованных ячеек, в каждую из которых поместили какие-то данные (один элемент данных в конкретную ячейку). Индексом массива является целое число, ссылающее на определенную часть массива. Индекс, как правило, имеет вид int.

**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.**

Код программы представлен в приложении А.

**Описание кода и использованных алгоритмов**

Функция main() – Тело программы представляет собой цикл do while(), в котором вызывается функция get\_info(), которая выводит меню. Далее пользователь вводит число (0-4), которое соответствует определенному действию. Выбор действия реализован через switch. Описание case:

1. – выход из программы
2. Вызывается функция get\_memory\_info(), которая с помощью sizеof() выводит размер каждого типа данных.
3. Пользователю необходимо ввести целочисленное число (value). Далее вызывается функция int\_to\_bin(int value), в которую передается введенное число. В самой функции объявляются две переменные типа unsigned int. Order – количество разрядов, mask – маска побитового сравнения. Циклом for проходим по всем разрядам, выводится результат поразрядной конъюнкции value и mask. После само число (value) побитово сдвигается на 1. Отделение знакого бита и остальных реализовано через условный оператор If.
4. Создается блок union объединяющий две переменные разных типов, но занимающих одинаковый объем памяти (int integerA и float floatB), они имеют одинаковый адрес в памяти. Пользователь вводит вещественное число (floatB), и оно передается в функцию float\_to\_bin(int integerA). Алгоритм работы функции идентичен пункту 2, потому что благодаря union происходит обработка данных типа int. Так же происходит отделение пробелами знакового бита, экспоненты и мантиссы.
5. Чтобы обработать double, создается так же блок union, но уже включающий целочисленный массив из двух элементов и переменную типа float. (int integerA[2], double doubleB). Пользователь вводит число и вызывается функция double\_to\_int(int integerA[]). Алгоритм работы функции схож с пунктами 2 и 3, но отличие в том, что мы используем два вложенных цикла for, в связи с использованием массива. Сначала происходят все необходимые побитовые операции с первым элементом массива, а после со вторым.

**Скрины работы программы.**

|  |  |
| --- | --- |
| 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 определенные действия. | Меню:    Проверка на ввод символов, которые не входят в диапазон выбора: |

Продолжение Таблицы

|  |  |
| --- | --- |
| Пункт 1 | |
| При выборе пользователем действия 1. На экран выводится информация о том, сколько памяти (в байтах) отводится под различные типы данных со спецификаторами и без. |  |

|  |  |
| --- | --- |
| Пункт 2 | |
| При выборе 2 пункта, пользователю необходимо ввести целочисленное (int) число, которое в последствии будет обработано с помощью функции int\_to\_bin(). | Выводится число в двоичной форме. Знаковый бит и каждые следующие 8 отделены пробелом, для корректного отображения. |

|  |  |
| --- | --- |
| Пункт 3 | |
| При выборе пункта 3, пользователю необходимо ввести вещественное число (float). Оно будет передано и обработано функцией float\_to\_bin(). | Выводится заданное число в двоичной форме. Пробелами отделены бит знака, мантисса и экспонента. Если число отрицательное, то первый бит будет равен 1, иначе 0. |

|  |  |
| --- | --- |
| Пункт 4 | |
| При выборе пункта 4, пользователю необходимо ввести вещественное число (double). Оно будет передано и обработано функцией float\_to\_double(). | Выводится заданное число в двоичной форме. Пробелами отделены бит знака, мантисса и экспонента. Если число отрицательное, то первый бит будет равен 1, иначе 0. |

Продолжение Таблицы

|  |  |
| --- | --- |
| Выход из программы | |
| Реализована возможность выхода из программы, пользователю необходимо ввести 0, тогда цикличное повторения программы завершится |  |

**Тестирование**

|  |  |
| --- | --- |
| Ввод некорректного значения |  |
| Ввод целого числа |  |
| Вещественные числа float |  |
| Вещественные числа double |  |

**Выводы.**

В результате выполнения практической работы на практике была изучена работа и представление типов данных в языке C++. Так же была реализована программа, позволяющая наглядно отобразить представление конкретных типов данных.

Приложение А

рабочий код

#include <iostream>

using namespace std;

void get\_info() {

cout << "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." << '\n';

cout << "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." << '\n';

cout << "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." << '\n';

cout << "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." << '\n';

cout << "0) Выход" << '\n';

cout << "Выберите действие: ";

}

void get\_memory\_info() {

cout << "\n";

cout << "int: " << sizeof(int) << endl;

cout << "short int: " << sizeof(short int) << endl;

cout << "long int: " << sizeof(long int) << endl;

cout << "float: " << sizeof(float) << endl;

cout << "double: " << sizeof(double) << endl;

cout << "long double: " << sizeof(long double) << endl;

cout << "char: " << sizeof(char) << endl;

cout << "bool: " << sizeof(bool) << endl;

cout << "\n";

}

void int\_to\_bin(int value) {

unsigned int order = 32; // Количество разрядов

unsigned int mask = 1 << order - 1; // Маска побитового сравнения

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

{

putchar(value & mask ? '1' : '0');

value <<= 1; // Побитовый сдвиг числа

if (i % 8 == 0)

{

putchar(' ');

}

if (i % order - 1 == 0)

{

putchar(' ');

}

}

}

void float\_to\_bin(int integerA) {

unsigned int order = sizeof(int) \* 8;

unsigned int mask = 1 << order - 1;

bool f = true;

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

{

putchar(integerA & mask ? '1' : '0');

integerA <<= 1;

if ((i % 9 == 0) && f)

{

putchar(' ');

f = false;

}

if (i % order - 1 == 0)

{

putchar(' ');

}

}

}

void double\_to\_bin(int integerA[2]) {

unsigned int order = sizeof(int) \* 8 \* 2;

unsigned int mask = 1 << order - 1;

bool fSign = true, fOrder = true;

for (int i = 1; i >= 0; i--) {

for (int j = 0; j < 32; j++) {

putchar(integerA[i] & mask ? '1' : '0');

integerA[i] <<= 1;

if ((j % 11 == 0) && fOrder && (j != 0))

{

putchar(' ');

fOrder = false;

}

if ((j % order == 0) && fSign)

{

putchar(' ');

fSign = false;

}

}

}

}

int main()

{

int action;

do

{

setlocale(0, "");

get\_info();

cin >> action;

switch (action)

{

case 0:

exit(0);

break;

case 1:

get\_memory\_info();

break;

case 2:

int value;

cout << "Введите число (int): ";

cin >> value;

int\_to\_bin(value);

cout << "\n\n";

break;

case 3:

{

union {

int integerA;

float floatB;

};

cout << "Введите число (float): ";

cin >> floatB;

float\_to\_bin(integerA);

cout << "\n\n";

break;

}

case 4:

{

union

{

int integerA[2];

double doubleB;

};

cout << "Введите число (double): ";

cin >> doubleB;

double\_to\_bin(integerA);

cout << "\n\n";

break;

}

default:

cout << "Такого вариата нет!\n\n";

break;

}

cout << "\n";

system("pause");

system("cls");

} while (action != 0);

}