**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 |  | Чванов М.В. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

Разработать алгоритм и написать программу, которая позволяет:

Научиться выводить сколько памяти (в байтах) на вашем компьютере отводится под различные типы данных со спецификаторами и без; выводить на экран двоичное представление в памяти (все разряды) целого и дробного чисел.

**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):

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:

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

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

putchar(value & mask ? '1' : '0'); // если 1, то возвращается 1, иначе 0

value <<= 1; // побитовый сдвиг влево на 1 бит

Putchar возвращает один символ в консоль. Альтернатива - cout. В представленном способе, маска - то, с чем сравнивается значение. И побитовый сдвиг применяется для value. Таким образом 1 бит будет сравниваться с каждым битом числа. Альтернатива - побитовый сдвиг вправо, но при этом нужно проводить данную операцию не над значением(единицей), а над маской (исходым числом, битовое представление которого нужно получить).

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).

Вещественные типы данных хранятся в памяти компьютера иначе, чем целочисленные. Внутреннее представление вещественного числа состоит из двух частей – мантиссы и порядка.

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:

Увидеть, каким образом вещественные типы данных представляются в компьютере немного сложнее. Логические операции, которые использовались с int, для вещественных типов данных не подходят. Но это ограничение можно легко обойти, использовав объединения.

Объединения – это две или более переменных расположенных по одному адресу (они разделяют одну и ту же память). Объединения определяются с использованием ключевого слова union. Объединения не могут хранить одновременно несколько различных значений, они позволяют интерпретировать несколькими различными способами содержимое одной и той же области памяти.

С объединениями нужно быть острожным. Вся работа с памятью требует грамотного подхода. Более подробно с объединениями можно будет ознакомиться при изучении структур. Пока что объедения будут служить инструментом для работы с float и double.

#include <iostream>

using namespace std;

int main()

{

union {

int tool;

float numb\_f = 3.14;

};

cout << tool << endl; // 1078523331

cout << numb\_f << endl; // 3.14

tool = tool >> 1; // побитовый сдвиг вправо

cout << tool << endl; // 5392261665

cout << numb\_f; // 1.3932e-19

return 0;

}

Подобные манипуляции возможны благодаря тому, что int и float занимают 4 байта. Проводя манипуляции над tool, мы изменяем значение numb\_f. Таким образом, алгоритм, который использовался для представления в памяти int может использоваться и для float.

Алгоритма представления double немного отличается. Под вещественное число с двойной точностью отводиться 8 байт, в то время как под int всего 4 байта. Но и это ограничение можно легко обойти. Так как данные любой линейной структуры в память записываются последовательно (друг за другом), можно использовать массив из двух int, под который будет отведено 8 байт.

#include <iostream>

using namespace std;

int main()

{

int value = -127; // Значение числа

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(' ');

}

}

return 0;

}

В консоль будет выведено: 1 1111111 11111111 11111111 10000001.

**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.

1. 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.

**Выполнение работы.**

Код программы представлен в приложении А.

На запуске программы пользователь наблюдает меню с командами.

Для вывода объёма занимаемой памяти для каждого типа данных используется функция **int** **sizes()**, которая не принимает аргументов и использует встроенные функции, такие как **cout** и **sizeof**.

После определяются функции, представляющие числа из десятичной в двоичную систему счисления: **int bin\_Int()** и **int bin\_Float()**.

**int bin\_Int()** и **int bin\_Float()**

На вход поступает число, определяются переменные для количества разрядов и маски. Далее, в цикле for используется операция для сравнения числа с маской, а также вывод 0 или 1, которые представляют собой биты, и вывод разделителя через putchar. Функции зациклены до того момента, пока не будет введена цифра ноль в конце цикла(или в начале). Также пользователь может поменять биты числа так, как он пожелает, но только один раз.

При вводе числа не из диапазона команд, программа попросит пользователя еще раз ввести число.

Скриншоты представлены далее в таблице

|  |  |
| --- | --- |
| Ввод пользователем и обработка данных | Работа алгоритма и вывод на экран |
| Меню | |
| При запуске программы перед пользователем появляется меню, где он может посмотреть, сколько памяти отводится на каждый тип данных, или выбрать тип данных для представления в двоичном коде. | Меню:    Проверка на ввод символов, которые не входят в диапазон выбора: |

|  |  |
| --- | --- |
| Представление числа в памяти | |
| При вводе пользователем корректного значения пункта меню, пользователь может ввести число (целочисленное или вещественное) | Как только введено число, программа выводит его представление в памяти и пользователь может еще раз ввести число:    Если пользователь хочет поменять биты местами: |

|  |  |
| --- | --- |
| Выход из программы | |
| Пользователь может ввести символ “0”, чтобы выйти из программы | В главном окне при вводе 0: |

**Выводы.**

Я научился выводить количество памяти (в байтах), которое отводится в компьютере под различные типы данных и выводить на экран двоичное представление в памяти (все разряды) целого и дробного чисел.

Приложение А

рабочий код

#include <iostream>

#include <cmath>

#include <ctype.h>

#include <iomanip>

using namespace std;

int sizes(int number\_choise) {

cout << "Выбрана команда: " << number\_choise << "\n\n";

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

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

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

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

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

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

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

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

return 0;

}

int bin\_Int(int number\_choise) {

cout << "Выбрана команда: " << number\_choise << "\n\n";

unsigned int bytes = 32, masks = 1 << bytes - 1, int\_num;

int A1[32] = {};

cout << "Введите целое число\n";

while (!(cin >> int\_num)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

while (int\_num != 0) {

for (int i = 0; i < bytes; i++)

{

putchar(int\_num & masks ? '1' : '0');

A1[i] = (int\_num & masks ? 1 : 0);

int\_num <<= 1;

cout << (i == 0 || i % 8 == 0 ? " " : "");

}

cout << "\nПоменять местами заданные пользователем биты\n"

"[1]ДА\n"

"[2]НЕТ\n";

int choises, a1, a2, temp, s = 0;

while (!(cin >> choises)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

if (choises == 1) {

cout << "Какие элементы поменять местами? ";

cin >> a1 >> a2;

if (a1 < a2) {

temp = A1[a1 - 1];

A1[a1 - 1] = A1[a2 - 1];

A1[a2 - 1] = temp;

}

else {

temp = A1[a2 - 1];

A1[a2 - 1] = A1[a1 - 1];

A1[a1 - 1] = temp;

}

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

cout << A1[i];

cout << (i == 0 || i % 8 == 0 ? " " : "");

}

cout << "\n";

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

s = s + (A1[i] \* pow(2, (bytes - 1 - i)));

}

cout << "Измененное число: " << s << "\n";

}

cout << "Введите число: ";

cout << "\n";

while (!(cin >> int\_num)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

}

return 0;

}

int bin\_Float(int number\_choise) {

cout << "Выбрана команда: " << number\_choise << "\n\n";

union

{

int int\_Float;

float float\_Num;

};

unsigned int bites = sizeof(int) \* 8, mask = 1 << bites - 1;

int A1[32] = {};

cout << "Введите вещественное число\n";

cin >> float\_Num;

while (!(cin >> float\_Num)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

while (float\_Num != 0) {

for (int i = 0; i < bites; i++)

{

putchar(int\_Float & mask ? '1' : '0');

A1[i] = (int\_Float & mask ? 1 : 0);

int\_Float <<= 1;

if (i == 0 || i == 1 || i == 8)

putchar(' ');

}

cout << "\nПоменять местами заданные пользователем биты\n"

"[1]ДА\n"

"[2]НЕТ\n";

int choises, a1, a2, temp;

float s = 0;

while (!(cin >> choises)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

if (choises == 1) {

cout << "Какие элементы поменять местами? ";

cin >> a1 >> a2;

if (a1 < a2) {

temp = A1[a1 - 1];

A1[a1 - 1] = A1[a2 - 1];

A1[a2 - 1] = temp;

}

else {

temp = A1[a2 - 1];

A1[a2 - 1] = A1[a1 - 1];

A1[a1 - 1] = temp;

}

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

cout << A1[i];

cout << (i == 0 || i == 1 || i == 8 ? " " : "");

}

cout << "\n";

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

s = s + (A1[i] \* pow(2, (bites - 1 - i)));

}

cout << "Измененное число: " << s << "\n";

}

cout << "Введите число: ";

cout << "\n";

while (!(cin >> float\_Num)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

}

return 0;

}

int main()

{

double number = 1;

setlocale(LC\_ALL, "RU");

while (number != 0) {

cout << "Введите номер команды:\n"

"[1]Вывод размерности типов данных.\n"

"[2]Вывод двоичной записи целого числа в представлении памяти.\n"

"[3]Вывод двоичной записи вещественного числа в представлении памяти.\n"

"[0]Выход из программы.\n";

cout << "Команда №: ";

while (!(cin >> number)) {

cout << "Ошибка, введен неподходящий тип данных\n";

cin.clear();

std::cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

int number\_choise = floor(number);

cout << "Принята команда: " << number\_choise;

cout << "\n";

if (number\_choise == 1 || number\_choise == 2 || number\_choise == 3 || number\_choise == 0) {

switch (number\_choise) {

case(1):

{

system("cls");

sizes(number\_choise);

system("pause");

break;

}

case(2):

{

system("cls");

bin\_Int(number\_choise);

system("pause");

break;

}

case(3):

{

system("cls");

bin\_Float(number\_choise);

system("pause");

break;

}

}

}

else

{

cout << "Введите число из диапазона цифр 0 - 3\n\n";

system("pause");

}

system("cls");

}

}