**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Вычислительной техники**

отчет

**по лабораторной работе № 1**

**по дисциплине «Организация ЭВМ и систем»**

Тема: Исследование внутреннего представления различных форматов данных

|  |  |  |
| --- | --- | --- |
| Студент гр. 3311 |  | Пасечный Л.В. |
| Преподаватель |  | Гречухин М. Н. |

Санкт-Петербург

2024

**Цель работы.**

Знакомство с внутренним представлением различных типов данных, используемых компьютером при их обработке.

**Задание (вариант 16)**

1. Разработать алгоритм ввода с клавиатуры unsinged long и double число и показать на экране его внутреннее представление в двоичной системе счисления
2. Написать и отладить программу на языке C++, реализующую разработанный алгоритм. Программа должна

* Иметь дружественный интерфейс
* Выводить на экран информативное сообщение при вводе некорректных данных
* Предложить повторный ввод пока не будут введены корректные данные

3. Выполнить путём сдвига вправо все биты, значение которых равно нулю и влево все биты, значение которых равно единице.

**Постановка задачи и описание решения**

Для вывода двоичного представления unsigned long используются побитовые операции. Задаётся переменная, имеющая значение 1, а затем происходит побитовое сравнение с unsigned long и при его равенстве выводится единица, иначе ноль. Далее при помощи битовых операции производим сдвиг переменной, имеющей значение 1, влево, и так до тех пор, пока не будут проверены все биты числа.

Для вывода двоичного представления числа типа double также используются побитовые операции. Однако, в отличие от простого целого типа, для работы с битами числа с плавающей запятой требуется доступ к его внутреннему представлению, так как оно хранится в формате IEEE 754 (1 бит знака, 8 бит экспоненты, 52 бита мантиссы).

Для этого используется union, который позволяет хранить одно и то же значение в разных типах данных. Мы записываем значение типа double в переменную union и можем получить доступ к его битам через эквивалентное представление типа long long. Далее происходит побитовое сравнение с каждой позицией бита, начиная с самого старшего, с помощью сдвига битов влево. Если бит установлен, выводится единица, если нет — ноль. Пробелы вставляются для разделения различных частей числа (знак, экспонента и мантисса).

Для сдвига битов равных 1 влево, а 0 вправо. используется побитовая маска.

Программа считает количество битов в числе, равных 1. Далее запускается алгоритм, в котором с помощью побитового или (|) выставляется 1 в маске. В итоге все биты равные 1 оказываются слева, а все нули справа.

Пример маски:

Входные данные: число 10 типа unsigned long и маска = 0.

Маска будет: 11000000... , так как 0000..0001010 = 10 в десятичной.

Хранение данных вещественного типа описывается IEEE 754 - стандартом в виде значения мантиссы (M) со знаком (S) и значения порядка (P). Число бит для хранения мантиссы и порядка зависит от типа данных с плавающей запятой.

float

|  |  |  |
| --- | --- | --- |
| S | P | M |

31 30 23 22 0

double

|  |  |  |
| --- | --- | --- |
| S | P | M |

63 62 52 51 0

long double

|  |  |  |
| --- | --- | --- |
| S | P | M |

79 78 64 63 0

Вещественное число в памяти хранится с нормализованной мантиссой, значение которой в десятичном эквиваленте лежит в диапазоне от 1 до 2. Причём 2 не входит в границу диапазона. Если в процессе выполнения какой-либо операции над данными с плавающей запятой значение мантиссы выходит из указанного диапазона, то в конце операции выполняется нормализация результата путем приведения значения мантиссы к указанному диапазону с соответствующим изменением значения порядка.

Порядок числа в соответствии с указанным форматом хранится «сдвинутым», то есть к его действительному значению добавляется в зависимости от формата такое число, чтобы порядок Р был всегда неотрицательным. Для формата double прибавляется 1023 Всегда неотрицательный порядок упрощает выполнение операции сравнения порядков и арифметических операций над ними, а также избавляет от необходимости выделять один бит для хранения знака порядка

**Контрольные примеры**

**Пример 1:**

**Исходные данные:**

Enter the unsigned long digit: 10

Результат:

Binary representstion of unsigned long is 00000000000000000000000000001010

Decimal represintation of unsigned long is 10

---------------------

Move bits 1 to the left in unsigned long: 11000000000000000000000000000000

Binary sistem : 3221225472

-----------DOUBLE-----------------

Исходные данные:

Enter the double digit: 10.5

Результат:

Binary representation of double is 0 10000000010 0101000000000000000000000000000000000000000000000000

Decimal representation of double is 10.5

---------------------

Move bits 1 to the left in double:

1 11100000000 0000000000000000000000000000000000000000000000000000

Binary system: -1.15292e+18

**Пример 2:**

-----------LONG-----------------

Исходные данные:

Enter the unsigned long digit: 15

Результат:

Binary representstion of unsigned long is 00000000000000000000000000001111

Decimal represintation of unsigned long is 15

---------------------

Move bits 1 to the left in unsigned long: 11110000000000000000000000000000

Binary sistem : 4026531840

-----------DOUBLE-----------------

Исходные данные:

Enter the double digit: 15.375

Результат:

Binary representation of double is 0 10000000010 1110110000000000000000000000000000000000000000000000

Decimal representation of double is 15.375

---------------------

Move bits 1 to the left in double:

1 11111100000 0000000000000000000000000000000000000000000000000000

Binary system: -1.44115e+17

**Пример 3:**

-----------LONG-----------------

Исходные данные:

Enter the unsigned long digit: 20

Результат:

Binary representstion of unsigned long is 00000000000000000000000000010100

Decimal represintation of unsigned long is 20

---------------------

Move bits 1 to the left in unsigned long: 11000000000000000000000000000000

Binary sistem : 3221225472

-----------DOUBLE-----------------

Исходные данные:

Enter the double digit: 20.15

Результат:

Binary representation of double is 0 10000000011 0100001001100110011001100110011001100110011001100110

Decimal representation of double is 20.15

---------------------

Move bits 1 to the left in double:

1 11111111111 1111111111111110000000000000000000000000000000000000

Binary system: -1.37439e+11

**Текст программы**

#include <iostream>

#include <limits>

using namespace std;

/\*\*

\* @brief Вывод числа типа unsigned long в двоичном представлении.

\* @param ul - Число типа unsigned long.

\*/

void print\_binar\_long(unsigned long ul)

{

for (int i = (sizeof(unsigned long) \* 8 - 1); i >= 0; i--)

{

unsigned long u1 = 1 << i;

cout << ((ul & u1) ? '1' : '0');

}

cout << endl;

}

/\*\*

\* @brief Вывод числа типа long long в двоичном представлении.

\* @param ul - число типа long long.

\*/

void print\_binar\_long\_long(long long ul)

{

for (int i = (sizeof(long long) \* 8 - 1); i >= 0; i--)

{

long long u1 = (ul >> i) & 1;

cout << u1;

if (i == 63 || i == 52)

cout << " ";

}

cout << endl;

}

/\*\*

\* @brief Вывод числа double в двоичном представлении.

\* @param d - число типа double.

\*/

void print\_binar\_double(double d)

{

union

{

double d;

long long ul;

} u;

u.d = d;

print\_binar\_long\_long(u.ul);

}

/\*\*

\* @brief Подсчёт битов равных 1 в числе типа unsigned long.

\* @param ul - число типа unsigned long.

\*/

int find\_1(unsigned long ul)

{

int num\_1 = 0;

for (int i = 0; i < (sizeof(unsigned long) \* 8); i++)

{

if (ul & 1 << i)

{

num\_1++;

}

}

return num\_1;

}

/\*\*

\* @brief Подсчёт битов равных 1 в числе типа long long.

\* @param ul - число типа long long.

\*/

int find\_1\_l(long long ul)

{

int num\_1 = 0;

for (int i = sizeof(long long) \* 8 - 1; i >= 0; i--)

{

long long test = ul >> i & 1;

if (test)

{

num\_1++;

}

}

return num\_1;

}

/\*\*

\* @brief Смещение в числе типа unsigned long всех битов 1 влево на 1 позицию, а 0 вправо на 1 позицию.

\* @param ul - число типа unsigned long.

\*/

void move\_bits\_int(unsigned long ul)

{

unsigned long ul1 = ul, mask = 0;

int num\_1;

cout << "---------------------" << endl;

num\_1 = find\_1(ul);

for (int i = 0; i < num\_1; i++)

{

mask |= 1 << (sizeof(unsigned long) \* 8 - 1 - i);

}

cout << "Move bits 1 to the left in unsigned long: ";

print\_binar\_long(mask);

cout << endl;

cout << "Binary sistem : " << mask << endl;

}

/\*\*

\* @brief Смещение в числе типа double всех битов 1 влево на 1 позицию, а 0 вправо на 1 позицию.

\* @param d - число типа double.

\*/

void move\_bits\_double(double d)

{

int num\_1;

long long mask = 0;

union

{

double d;

long long ll;

} u;

cout << endl;

u.d = d;

cout << "---------------------" << endl;

num\_1 = find\_1\_l(u.ll);

cout << "Move bits 1 to the left in double: " << endl;

for (int i = 0; i < num\_1; i++)

{

mask |= 1LL << (sizeof(long long) \* 8 - 1 - i);

}

print\_binar\_long\_long(mask);

cout << endl;

cout << "Binary system: " << double(mask) << endl;

cout << "---------------------" << endl;

}

int main()

{

unsigned long ul;

cout << "-----------LONG-----------------" << endl;

cout << "Enter the unsigned long digit: ";

while (!(cin >> ul))

{

cout << "Invalid number? try again";

cin.clear();

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

};

cout << "Binary representstion of unsigned long is ";

print\_binar\_long(ul);

cout << "Decimal represintation of unsigned long is ";

cout << ul << endl;

move\_bits\_int(ul);

cout << "-----------DOUBLE-----------------" << endl;

double d;

cout << "Enter the double digit: ";

while (!(cin >> d))

{

cout << "Invalid number? try again";

cin.clear();

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

};

cout << "Binary representation of double is ";

print\_binar\_double(d);

cout << "Decimal representation of double is ";

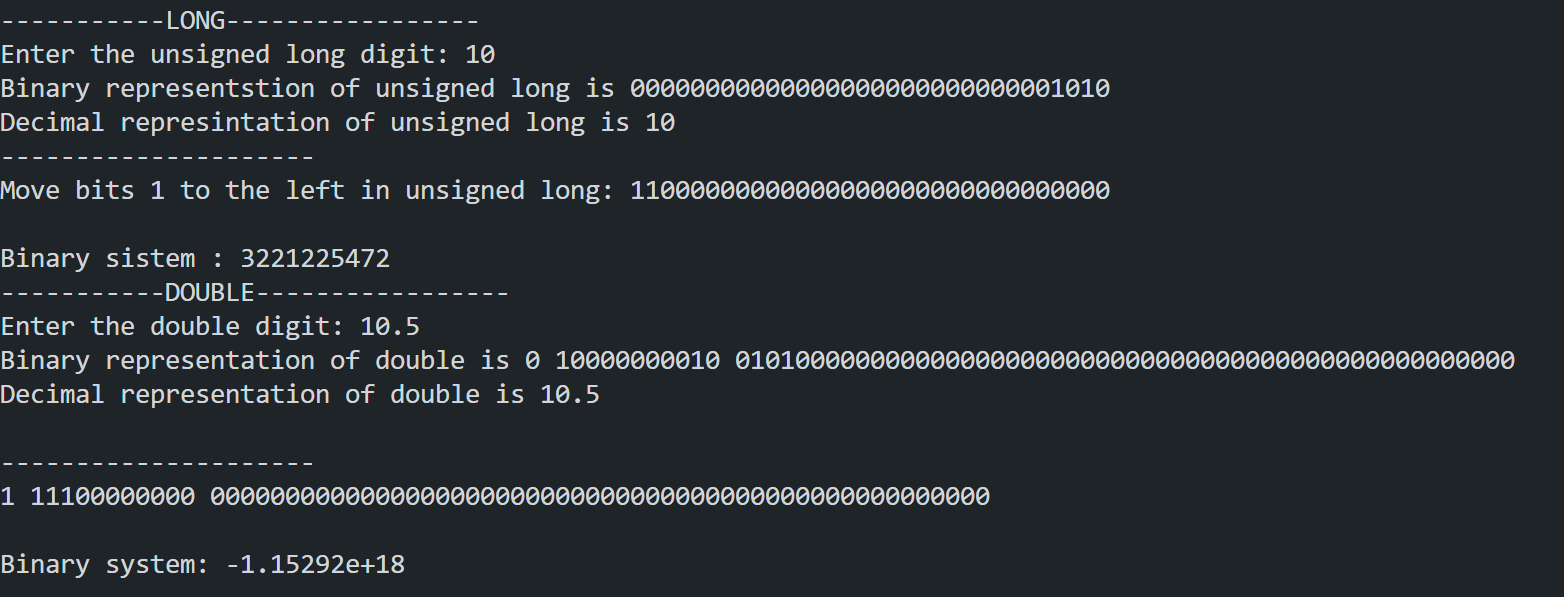
cout << d << endl;

move\_bits\_double(d);

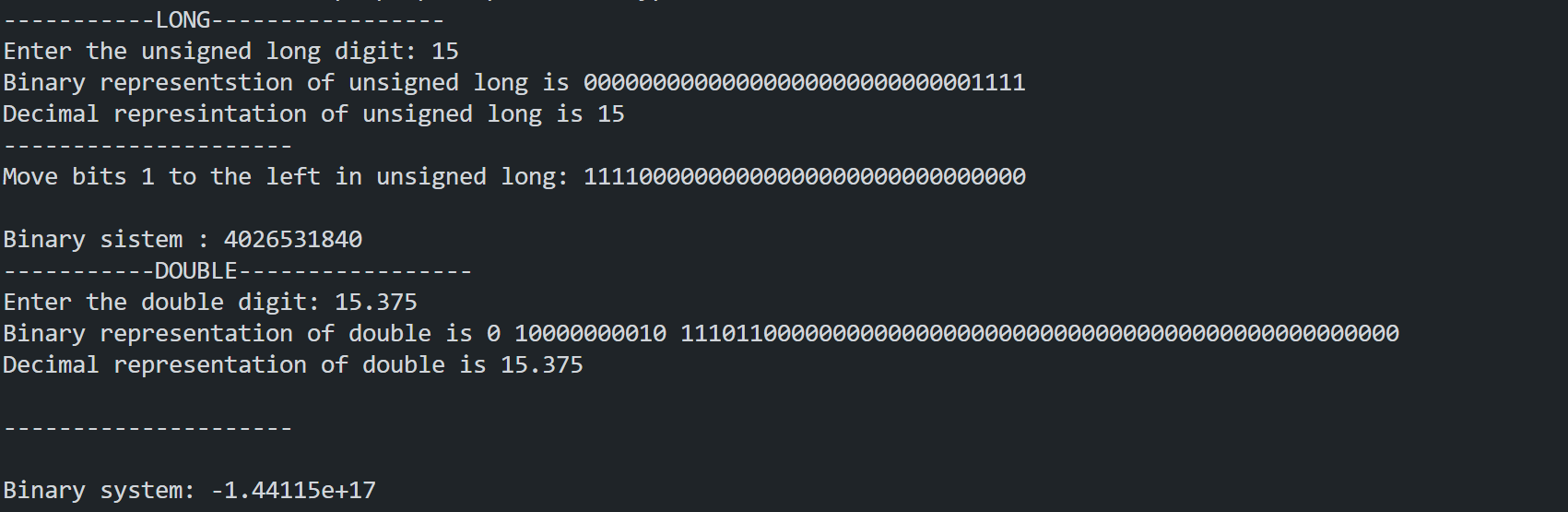
return 0;

}

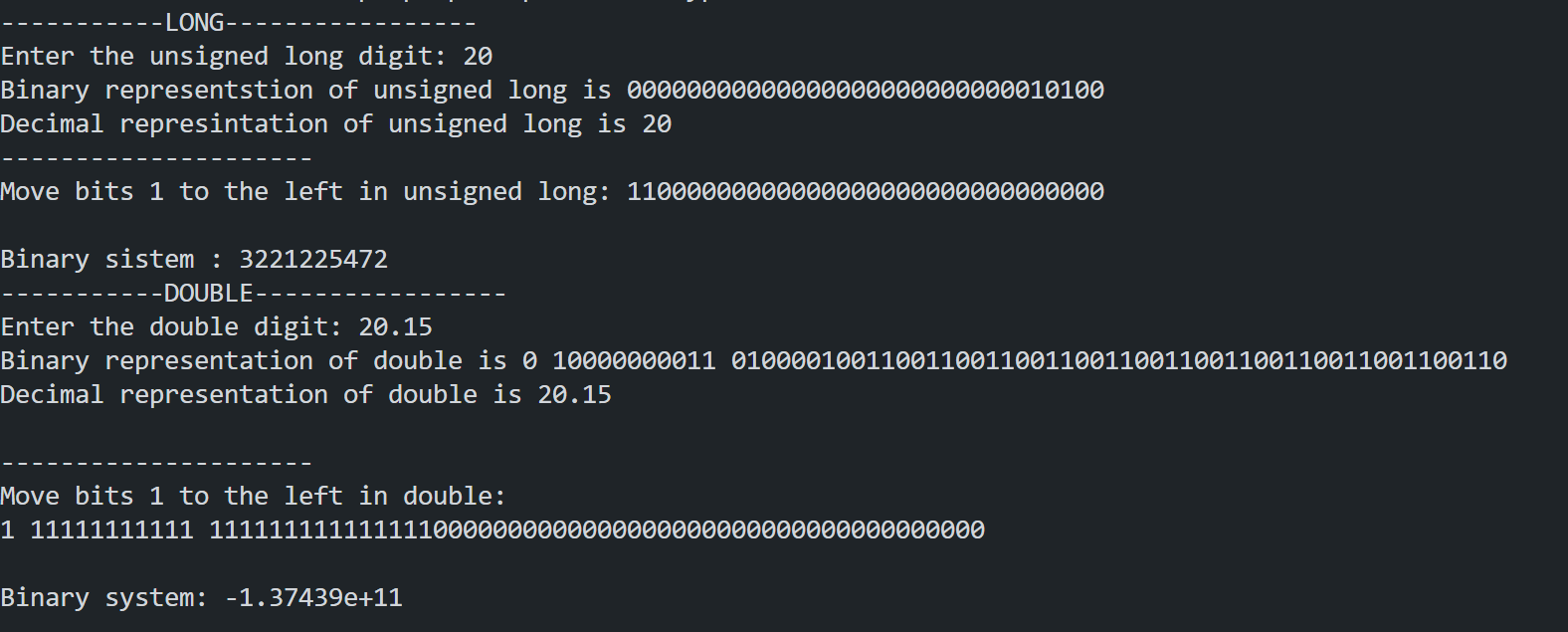
**Примеры выполнения программы**



Пример 1



Пример 2



Пример 3

**Выводы.**

В ходе выполнения работы мы изучили, как компьютер хранит и обрабатывает различные типы данных, а также подробно рассмотрели двоичное представление целых чисел и операции сдвига битов. Мы узнали и применили следующие ключевые моменты:

Мы изучили, как целые числа хранятся в памяти компьютера в виде двоичных кодов. Для этого было реализовано отображение чисел в двоичном виде, что позволило визуализировать, как каждый бит числа используется для представления данных. Также мы научились работать с побитовыми сдвигами чисел.

В процессе работы мы коснулись темы представления чисел с плавающей запятой, где изучили структуру числа: знак, экспонента и мантисса. Это дало понимание, как числа с плавающей запятой хранятся в двоичной системе.

Был разработан и реализован алгоритм, который позволяет менять биты местами.

Таким образом, мы научились эффективно работать с двоичными представлениями данных, узнали, как компьютер хранит целые числа и числа с плавающей запятой, и реализовали операции сдвига и перестановки битовых групп с учётом внутренней структуры данных.