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. 0321 _____ Земсков Д. И.

Преподаватель _____ Чугунов Л. А.

Санкт-Петербург

2022

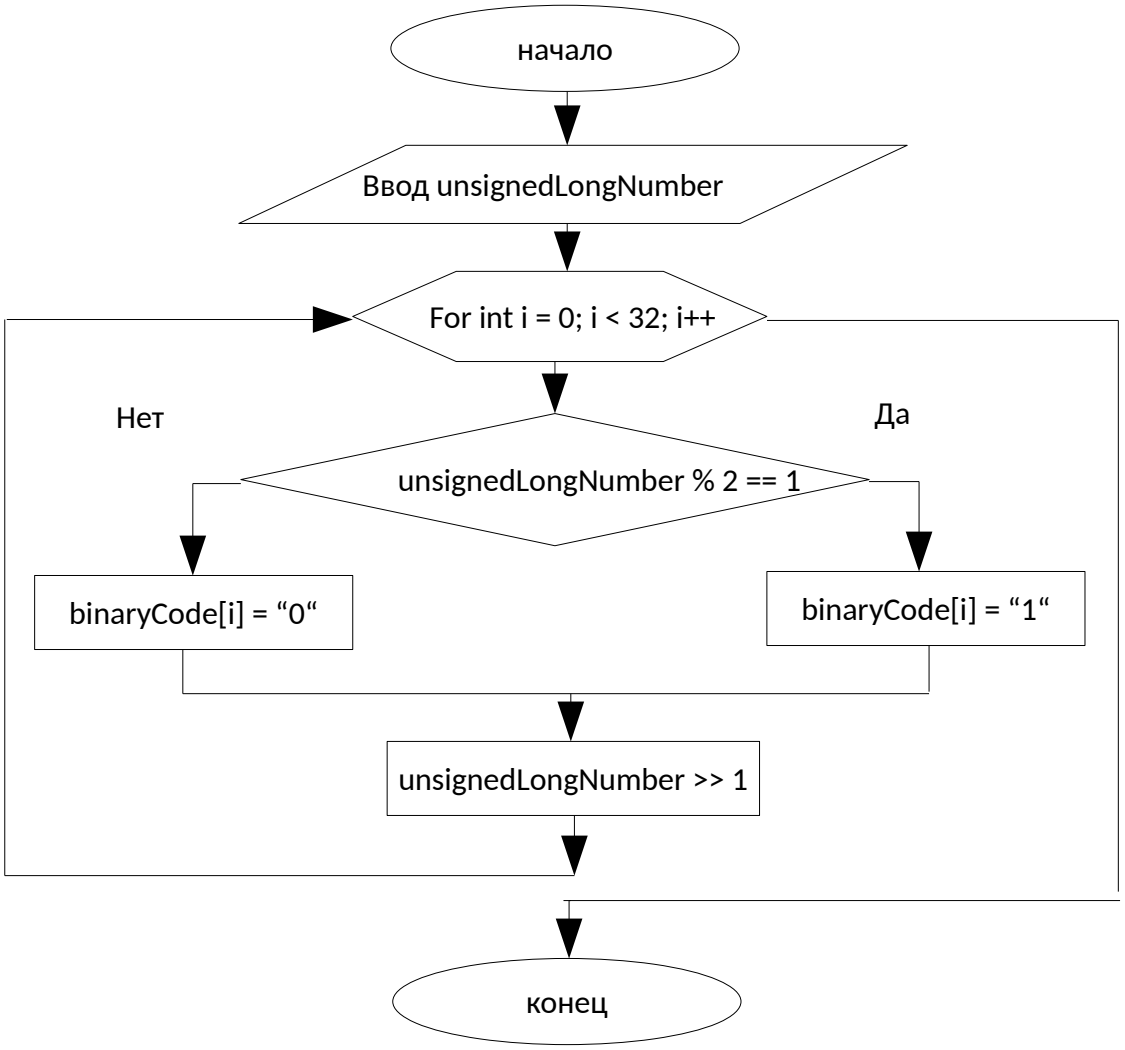
Цель работы.

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

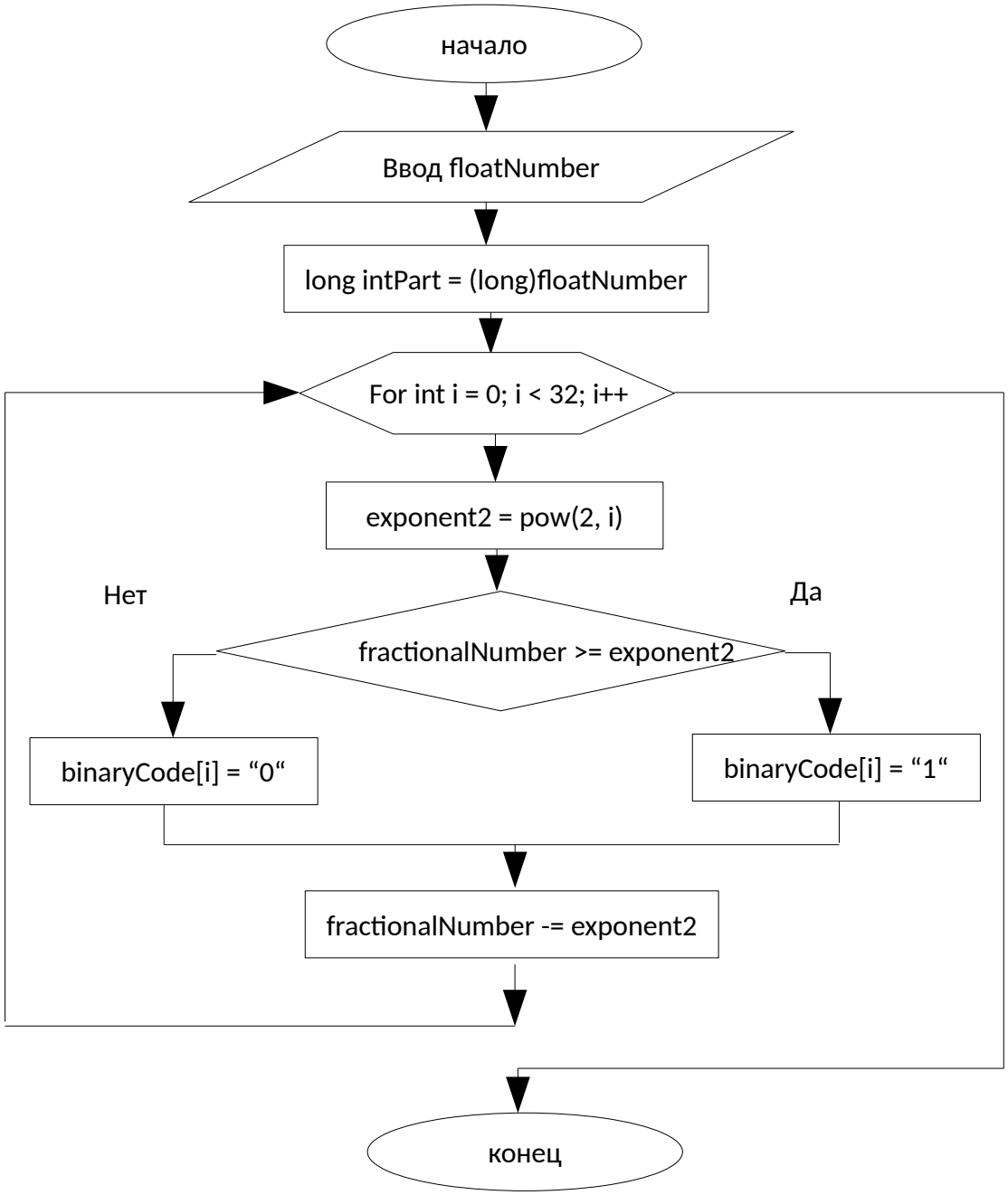
Задание

1. Разработать алгоритм ввода с клавиатуры чисел типов данных unsigned long и float и показать на экране их внутреннее представление в двоичной системе счисления.
2. Написать и отладить программу на языке C++, реализующую разработанный алгоритм.
3. Дополнить алгоритм блоками для выполнения преобразования двоичного полученного кода исходного типа данных и последующего вывода преобразованного кода в двоичной системе счисления и в формате исходного данного. Преобразование заключается в замене местами значений рядом стоящих бит в парах. Количество пар и номер старшего разряда левой пары задаётся с клавиатуры.

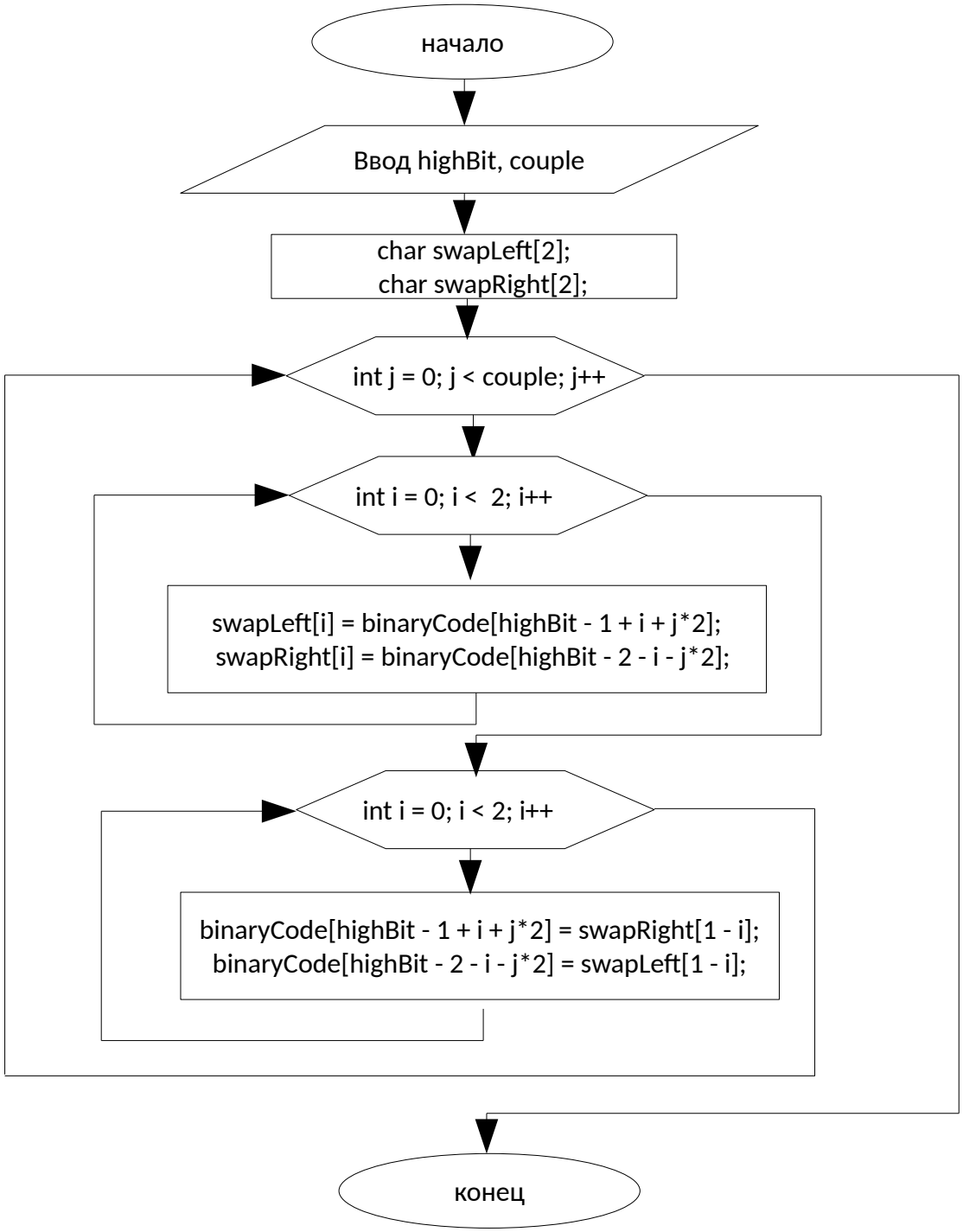
Блок-схема алгоритма преобразования числа типа unsigned long в двоичное представление.



Блок-схема алгоритма преобразования числа типа float в двоичное представление.



Блок-схема алгоритма преобразования полученных чисел согласно вариату.



Текст программы:

```
#include <iostream>
#include <math.h> // для функции pow
const int SIZE_OF_LONG = 32;  // число типа unsigned long состоит из 32 разрядов в двоичной системе
const int SIZE_EXP_OF_FLOAT = 8; // количество битов, выделенных для порядка числа типа float
const int SIZE_MANTISSA_OF_FLOAT = 23; // количество битов, выделенных для мантиссы числа типа float
const int SIZE_OF_FLOAT = SIZE_EXP_OF_FLOAT + SIZE_MANTISSA_OF_FLOAT + 1;

using namespace std;

char *unsignedLongNumberSave;
char *unsignedFloatNumberSave;

void printBinary(char* binaryCode, int sizeToPrint) {
    for (int i = sizeToPrint; i != 0 ; i--) {
        cout << binaryCode[i - 1];
    }
    cout << endl;
}

void convertToBinary(char *binaryCode, unsigned long unsignedLongNumber, int numberOfBit, int startPoint) {
    for (int i = startPoint; i < (numberOfBit + startPoint); i++) {
        binaryCode[i] = unsignedLongNumber % 2 == 1 ? '1' : '0';
         unsignedLongNumber = unsignedLongNumber >> 1; // побитовый сдвиг вправо
    }
}

void zeroing(char *binaryCode, int numberOfBit) { // обнуляем элементы массива
    for (int i = 0; i < numberOfBit; i++) {
        binaryCode[i] = '0';
    }
}
```

```
    void convertFractionalToBinary(char *binaryCode, double fractionalNumber,
int numberOfBit) {
        for (int i = 1; i < numberOfBit; i++) {

            double exponent2 = pow(2, -i);
            if ((fractionalNumber >= exponent2) && (fractionalNumber != 0)) {
                binaryCode[numberOfBit - i] = '1';
                fractionalNumber -= exponent2;
            }
        }
    }

    void    convertIntToBinary(char    *binaryCode,    unsigned    long
unsignedLongNumber, int numberOfBit) {
        for (int i = 1; i <= numberOfBit; i++) {

            unsigned long exponent2 = pow(2, numberOfBit - i);
              if ((unsignedLongNumber >= exponent2) && (unsignedLongNumber !=
0)) {
                binaryCode[numberOfBit - i] = '1';
                unsignedLongNumber -= exponent2;
            }
        }
    }

    int findExponent(char *binaryCode, int numberOfBit) {
        for (int i = numberOfBit; i > 0; i--) {
            if ((binaryCode[i - 1]) == '1') {
                return (numberOfBit - i + 1) * (-1);
            }
        }
        return 0;
    }

    void mergeBinaryArray(char *mainBinaryCode, char *addBinaryCode, int wide,
int offset) {
        for (int i = 1; i <= wide; i++) {
                            mainBinaryCode[SIZE_MANTISSA_OF_FLOAT   -   i]   =
addBinaryCode[(SIZE_MANTISSA_OF_FLOAT * 2) - i + offset];
        }
    }
```

```
      void mergeIntAndFloat(char *binaryCode, char *intPart, int sizeIntPart,
char *floatPart, int sizeFloatPart, int offset) {
          for (int i = 0; i <= offset; i++){
              binaryCode[SIZE_MANTISSA_OF_FLOAT - i + 1] = intPart[offset - i];
          }
          for (int i = 0; i < (SIZE_MANTISSA_OF_FLOAT - offset); i++){
                      binaryCode[SIZE_MANTISSA_OF_FLOAT  -  offset  -  i]  =
floatPart[sizeFloatPart - i - 1];
          }
      }

      char *convertLongToBinary() {
          cout << "Введите число типа Unsigned long:";
          unsigned long unsignedLongNumber;
          cin >> unsignedLongNumber;
           char *binaryCodeLong = new char[SIZE_OF_LONG]; // выделяем память под
массив символов для числа Long
          zeroing(binaryCodeLong, SIZE_OF_LONG);
          convertToBinary(binaryCodeLong, unsignedLongNumber, SIZE_OF_LONG, 0);
            cout  <<  "Число  в  формате  unsigned  long  в  двоичной  системе  будет
выглядеть так: ";
          unsignedLongNumberSave = binaryCodeLong;
          return binaryCodeLong;
      }

      char *convertFloatToBinary() {
          cout << "Введите число типа float:";
          float floatNumber;
          cin >> floatNumber;
          char *binaryCodeFloat = new char[SIZE_OF_FLOAT];
          zeroing(binaryCodeFloat, SIZE_OF_FLOAT);
           binaryCodeFloat[SIZE_OF_FLOAT - 1] = floatNumber > 0 ? '0' : '1'; //
определяем знак
          if (floatNumber == 0) { // особый случай когда число = 0
                      convertToBinary(binaryCodeFloat,  -128,  SIZE_EXP_OF_FLOAT,
SIZE_MANTISSA_OF_FLOAT);
              return binaryCodeFloat;
          }
          floatNumber = abs(floatNumber);
          long intPart = (long)floatNumber; // целая часть
           char intPartBinary[SIZE_MANTISSA_OF_FLOAT]; // целая часть в двоичном
представлении
```

```
        zeroing(intPartBinary, SIZE_MANTISSA_OF_FLOAT);
        double floatPart = floatNumber - intPart; // дробная часть. Необходимо использовать тип double для увеличения количества разрядов мантиссы
          char floatPartBinary[SIZE_MANTISSA_OF_FLOAT * 2]; // целая часть в двоичном представлении
        zeroing(floatPartBinary, SIZE_MANTISSA_OF_FLOAT * 2);
        int offset;
        if ((floatPart != 0) && (intPart == 0)) {
                    convertFractionalToBinary(floatPartBinary, floatPart, SIZE_MANTISSA_OF_FLOAT * 2);
             offset = findExponent(floatPartBinary, SIZE_MANTISSA_OF_FLOAT * 2);
                        mergeBinaryArray(binaryCodeFloat, floatPartBinary, SIZE_MANTISSA_OF_FLOAT, offset); // записываем в матрицу мантиссу
             convertToBinary(binaryCodeFloat, 127 + offset, SIZE_EXP_OF_FLOAT, SIZE_MANTISSA_OF_FLOAT); // записываем в матрицу показатель
            unsignedFloatNumberSave = binaryCodeFloat;
            return binaryCodeFloat;
        }
        // далее целая часть не равна 0
        convertIntToBinary(intPartBinary, intPart, SIZE_MANTISSA_OF_FLOAT);
                    convertFractionalToBinary(floatPartBinary, floatPart, SIZE_MANTISSA_OF_FLOAT * 2);
             offset = SIZE_MANTISSA_OF_FLOAT + findExponent(intPartBinary, SIZE_MANTISSA_OF_FLOAT) + 1;
                        mergeIntAndFloat(binaryCodeFloat, intPartBinary, SIZE_MANTISSA_OF_FLOAT, floatPartBinary, SIZE_MANTISSA_OF_FLOAT * 2, offset);
         convertToBinary(binaryCodeFloat, 127 + offset - 1, SIZE_EXP_OF_FLOAT, SIZE_MANTISSA_OF_FLOAT); // записываем в матрицу показатель

        cout << "Число в формате float в двоичной системе будет выглядеть так: ";
        unsignedFloatNumberSave = binaryCodeFloat;
        return binaryCodeFloat;
    }

    void swapBinary(char * binaryCode, int highBit, int couple) {
        char swapLeft[2];
        char swapRight[2];
        for (int j = 0; j < couple; j++) {
            for (int i = 0; i <  2; i++) {
                swapLeft[i] = binaryCode[highBit - 1 + i + j*2];
```

```
                swapRight[i] = binaryCode[highBit - 2 - i - j*2];
            }

            for (int i = 0; i < 2; i++) {
                binaryCode[highBit - 1 + i + j*2] = swapRight[1 - i];
                binaryCode[highBit - 2 - i - j*2] = swapLeft[1 - i];
            }
        }
    }

    int main()
    {
        printBinary(convertLongToBinary(), SIZE_OF_LONG);
        printBinary(convertFloatToBinary(), SIZE_OF_FLOAT);

         cout << endl << "Задание: Поменять местами значения рядом стоящих бит в парах" << endl;
        cout << "Введите номер старшего разряда левой пары: ";
        int highBit;
        cin >> highBit;
        cout << "Введите количество пар: ";
        int couple;
        cin >> couple;
          if ((highBit <= (couple * 2)) || (SIZE_OF_LONG < (couple * 2 + highBit))) {
              cout << "Введено недопустимое количество пар либо введён номер несуществующего старшего разряда левой пары";
            return 0;
        }
        swapBinary(unsignedLongNumberSave, highBit, couple);
        swapBinary(unsignedFloatNumberSave, highBit, couple);
        cout << "Результат для числа типа unsigned long: ";
        printBinary(unsignedLongNumberSave, SIZE_OF_LONG);
        cout << "Результат для числа типа float: ";
        printBinary(unsignedFloatNumberSave, SIZE_OF_FLOAT);

        return 0;
    }
```

Примеры запуска программы.

Пример запуска программы №1:

```
Введите число типа Unsigned long:153647
Число в формате unsigned long в двоичной системе будет выглядеть так: 00000000000000100101100000101111
Введите число типа float:-16.8132
Число в формате float в двоичной системе будет выглядеть так: 11000001100001101000000101101110

Задание: Поменять местами значения рядом стоящих бит в парах
Введите номер старшего разряда левой пары: 5
Введите количество пар: 2
Результат для числа типа unsigned long: 00000000000000100101100011111000
Результат для числа типа float: 11000001100001101000000110111001
```

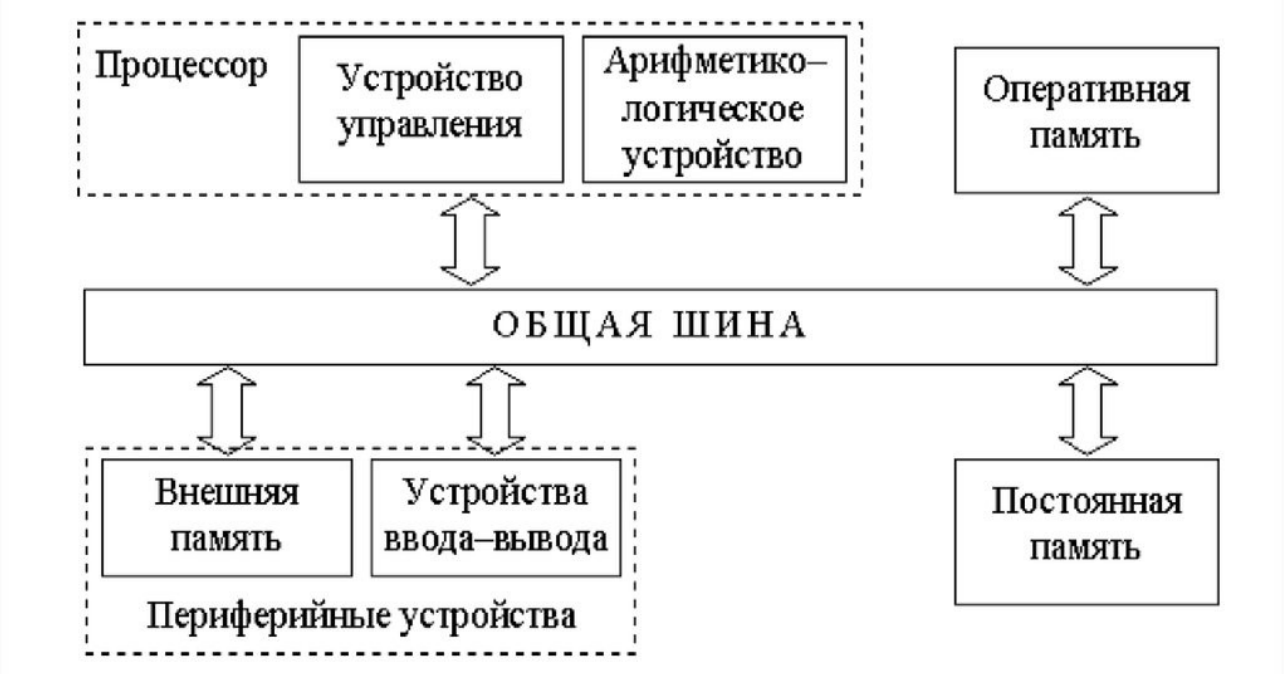
Пример запуска программы №2:

```
Введите число типа Unsigned long:1900057
Число в формате unsigned long в двоичной системе будет выглядеть так: 00000000000111001111111000011001
Введите число типа float:0.00824
00111100000001110000000100010001

Задание: Поменять местами значения рядом стоящих бит в парах
Введите номер старшего разряда левой пары: 8
Введите количество пар: 1
Результат для числа типа unsigned long: 00000000000111001111111000011001
Результат для числа типа float: 00111100000001110000000001010001
```

Пример запуска программы №3 (с вводом неправильных параметров):

```
Введите число типа Unsigned long:0
Число в формате unsigned long в двоичной системе будет выглядеть так: 00000000000000000000000000000000
Введите число типа float:0
11000000000000000000000000000000

Задание: Поменять местами значения рядом стоящих бит в парах
Введите номер старшего разряда левой пары: 3
Введите количество пар: 5
Введено недопустимое количество пар либо введён номер несуществующего старшего разряда левой парыdmitry@c
```

Структурная схема аппаратных средств, используемых при выполнении программы:



Выводы.

В результате данной было осуществлено знакомство с внутренним представлением различных типов, используемых компьютером при их обработке. При отладке была выявлена правильная работа, правильность данных была проконтролирована с помощью сайта: https://www.h-schmidt.net/FloatConverter/IEEE754.html .