**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.**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 3376 |  | Михайлов Н.Ф. и Прошунина С.А. |
| Преподаватель |  | Ильин С.Е. |

Санкт-Петербург

2024

**Задание на лабораторную работу.**

1. Разработать алгоритм ввода с клавиатуры требуемых типов данных (unsigned char и double), показать на экране их внутреннее представление в двоичной системе счисления. Также реализовать следующие действия в двоичном представлении: в пределах заданной группы все биты, значение которых равно единице сдвинуть вправо, а все биты, значение которых равно нулю влево.
2. Написать и отладить программу на языке С++, реализующую разработанный алгоритм.

**Блок-схема алгоритма**

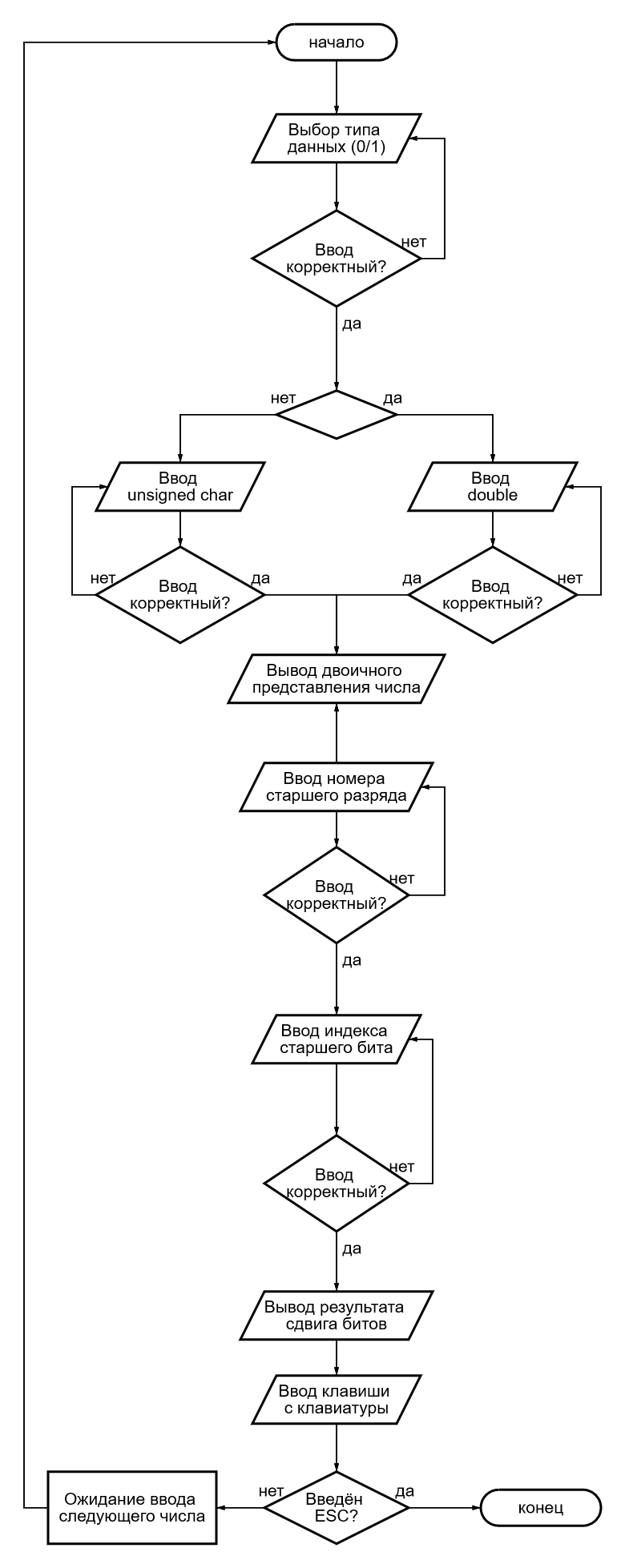


Рисунок. 1 ­. Основной алгоритм программы

**Текст программы**

#include <conio.h>

#include <iostream>

#include <cstring>

enum {

ESC = 27,

};

union DOUBLE {

double m\_double = 0;

unsigned char byte[8];

} number;

void PrintBinaryDouble(const DOUBLE& number) {

for (int i = 7; i >= 0; i--) {

for (int j = 7; j >= 0; j--) {

bool bite = number.byte[i] & (1 << j);

std::cout << bite;

}

std::cout << " ";

}

std::cout << std::endl;

}

void PrintBinaryChar(unsigned char t\_char) {

for (int i = 7; i >= 0; --i) {

std::cout << ((t\_char >> i) & 1);

}

std::cout << std::endl;

}

unsigned char BitSwaper(unsigned char& value, unsigned bitAmmount, unsigned index) {

unsigned ones = 0, zeros = 0;

for (unsigned i = index; i < index + bitAmmount; i++) {

if (value & (1 << i)) ones++;

else zeros++;

}

std::cout << "Количество единиц: " << ones << " Количество нулей: " << zeros << std::endl;

while (ones + zeros > 0) {

if (ones > 0) {

value |= (1 << index); // Устанавливаем единицу

ones--;

}

else {

value &= ~(1 << index); // Устанавливаем ноль

zeros--;

}

index++;

}

return value;

}

void BitSwaper(DOUBLE& number, unsigned bitAmmount, unsigned index) {

unsigned ones = 0, zeros = 0;

for (unsigned i = index / 8; i <= (index + bitAmmount - 1) / 8; ++i) {

unsigned startBit = (i == index / 8) ? (index % 8) : 0;

unsigned endBit = (i == (index + bitAmmount - 1) / 8) ? ((index + bitAmmount) % 8) : 8;

for (unsigned j = startBit; j < endBit; ++j) {

if (number.byte[i] & (1 << j)) ones++;

else zeros++;

}

}

std::cout << "Количество единиц: " << ones << " Количество нулей: " << zeros << std::endl;

unsigned currentIndex = index;

for (unsigned i = index / 8; i <= (index + bitAmmount - 1) / 8; ++i) {

unsigned startBit = (i == index / 8) ? (index % 8) : 0;

unsigned endBit = (i == (index + bitAmmount - 1) / 8) ? ((index + bitAmmount) % 8) : 8;

for (unsigned j = startBit; j < endBit; ++j) {

if (ones > 0) {

number.byte[i] |= (1 << j); // Устанавливаем единицу

ones--;

}

else if (zeros > 0) {

number.byte[i] &= ~(1 << j); // Устанавливаем ноль

zeros--;

}

}

}

}

int main() {

//DOUBLE number;

int keyInput;

unsigned bitAmmount = 1, index = 1;

bool dataType = 0; // unsigned char = 0, double = 1

long long int t\_char = -1;

unsigned char unsignedChar = 0;

do {

keyInput = \_getch();

std::cout << "Выберите тип данных (unsigned char - 0, double - 1) : ";

std::cin >> dataType;

while (dataType != 1 && dataType != 0 || std::cin.fail()) {

std::cout << "Введите корректное число. Либо 1, либо 0: ";

std::cin.clear();

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

std::cin >> dataType;

}

if (!dataType) {

std::cout << "Введите число, соответсвующее типу (unsigned char): ";

std::cin >> t\_char;

while (t\_char > 255 || t\_char < 0 || std::cin.fail()) {

std::cout << "Введите корректное число в диапазоне [0;255]: ";

std::cin.clear();

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

std::cin >> t\_char;

}

unsignedChar = static\_cast<unsigned char>(t\_char);

PrintBinaryChar(unsignedChar);

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

}

else {

std::cout << "Введите число, соответсвующее типу (double): ";

std::cin >> number.m\_double;

PrintBinaryDouble(number);

}

std::cout << "Введите номер старшего разряда: ";

std::cin >> index;

if (!dataType) {

while (index >= 8|| index < 0 || std::cin.fail()) {

std::cout << "Введите корректный индекс в диапазоне [0;7]: ";

std::cin.clear();

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

std::cin >> index;

}

}

else {

while (index >= 64 || index < 0 || std::cin.fail()) {

std::cout << "Введите корректный индекс в диапазоне [0;63]: ";

std::cin.clear();

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

std::cin >> index;

}

}

std::cout << "Введите количество бит: ";

std::cin >> bitAmmount;

if (!dataType) {

if (bitAmmount + index > 8) bitAmmount = 8 - index;

PrintBinaryChar(BitSwaper(unsignedChar, bitAmmount, index));

std::cout<< "Число после преобразований: " << static\_cast<int>(unsignedChar) << std::endl;

}

else {

if (bitAmmount + index > 64) bitAmmount = 64 - index;

BitSwaper(number, bitAmmount, index);

PrintBinaryDouble(number);

}

} while (keyInput != ESC);

}

**Примеры запуска и работы программы**

После запуска программы на экране появляется консоль, в которой пользователю предоставляется выбор типа данных (рис 2)



Рисунок. 2 ­Выбор типа данных

После выбора данных, производится ввод числа и вывод его представления в двоичном виде (рис 3)

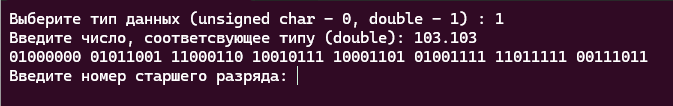


Рисунок. 3 ­Ввод числа в нужном типе и вывод в двоичном виде

Программа запрашивает на ввод номер старшего разряда и количество бит в группе. В консоли выводится количество единиц и нулей на заданном участке, а также вывод измененного представления числа в двоичном виде для сравнения. (рис 4)

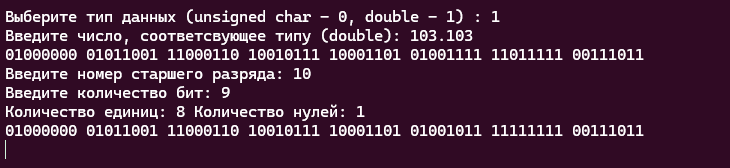


Рисунок. 4 ­Ввод старшего разряда, количества бит, а также вывод результата сдвигов

Далее программа продолжает свою работу и предлагает заново ввести тип данных. Для окончания работы требуется нажать ESC.

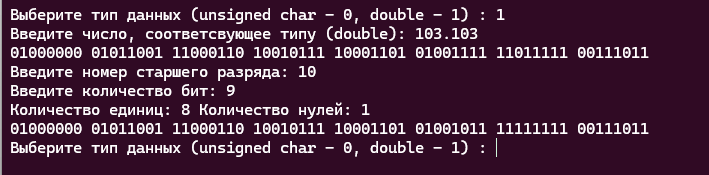
**

Рисунок. 5 ­ Ожидание ввода следующего числа