|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное автономное образовательное учреждение высшего образования  «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ» |

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ассистент |  |  |  | Синёв Н. И. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7 |
| ПОРАЗРЯДНАЯ ОБРАБОТКА ЦЕЛЫХ ЧИСЕЛКУРС 1 |
| по курсу: ОСНОВЫ ПРОГРАММИРОВАНИЯ |
|  |
|  |

****РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 1941 |  |  |  | Князюк Р.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2020

1. **Постановка задачи.**

В длинном целом числе N поменять байты в зеркальном порядке.

1. **Формализация.**

* **Список переменных в данной программе:**
  + Переменные типа **long long (8 байт)**:

***test\_1* –** переменная, хранящая обрабатываемое число;

***temp\_1,temp\_2 –*** переменные для перестановки байтов;

***mask\_1 –*** для считывания левого байта (в процессе выполнения кода сдвигается вправо на 8 бит(1 байт));

***mask\_2*** *–* для считывания самого правого байта (в процессе выполнения кода сдвигается влево на 8 бит(1 байт));

***support\_mask*** *–* для удаления 1 при сдвиге вправо;

* + Переменные типа **int**:

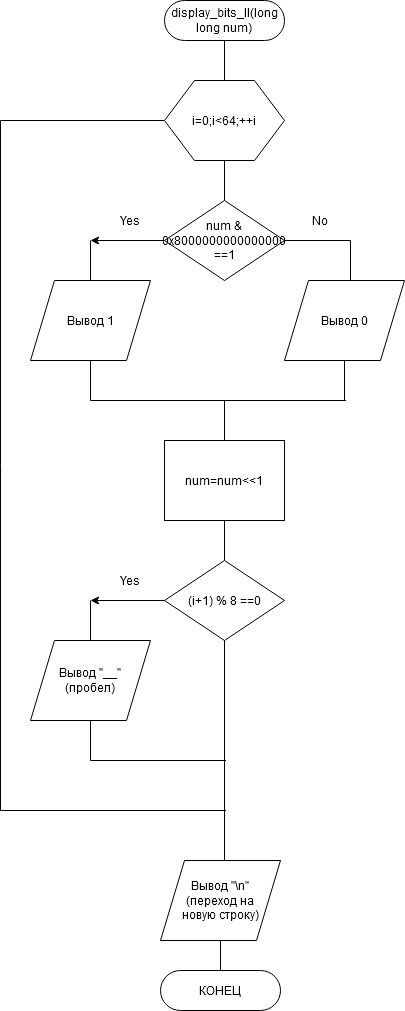
***i*** – счетчик ,используемый в циклах для работы;

* Все данные вводятся с клавиатуры.
* Для удобства просмотра данных в консоли изменим ее размер перед выполнением программы.
* Алгоритм для вывода числа в двоичном виде будет отдельной подпрограммой.
* В качестве длинного целого числа будем использовать переменную **long long** (знакового типа);
* Так как необходимо поменять байты в зеркальном порядке, воспользуемся двумя масками. ***mask\_1=0xff00000000000000*** (первый байт заполнен единицами, остальные нулями), ***mask\_2=0x00000000000000ff*** (последний байт заполнен 1, а остальные 0).***,support\_mask=0x00ffffffffffffff .***Маски нужны для извлечения байтов с правой и левой стороны
* Обработку числа организуем циклом for (так как известно кол-во итераций), количеств итераций равно 4(количеству байтов типа **long** **long** деленое на 2).
* В цикле, сначала через побитовое умножение получаем 8 и 1 байт и сохраняем их в temp\_1,temp\_2. После меняем местами полученные байты сдвигом. Обнуляем байты в числе, после чего складываем с числом наши выделенные байты. Таким образом мы переставили 1 и 8 байт. Аналогично для 2 и 7,3 и 6, 4 и 5.
* Для выделения необходимых байтов будем сдвигать маски. **mask\_1** вправо, **mask\_2** влево.
* После ввода числа, программа выведет его в двоичном формате и начнет обработку.
* В конце программа должны вывести обработанное число в двоичном формате;
* Операционная система: Windows 7 Максимальная
* Написание кода: Visual Studio Code
* Компилятор: MinGW GCC-8.2.0-5
* Запуск программы через \*.exe

**Таблица 1 – Тестовые примеры**

|  |  |  |
| --- | --- | --- |
| **Пример** | **№1** | **№2** |
| **Данные** | **1280**  **(00000000 00000000 00000000 00000000 00000000 00000000 00000101 00000000)** | **675539944105574400**  **(00001001 01100000 00000000 00000000 00000000 00000000 00000000 00000000)** |
| **Результат** | **00000000 00000101**  **00000000 00000000**  **00000000 00000000**  **00000000 00000000** | **00000000 00000000 00000000 00000000 00000000 00000000 01100000 00001001** |

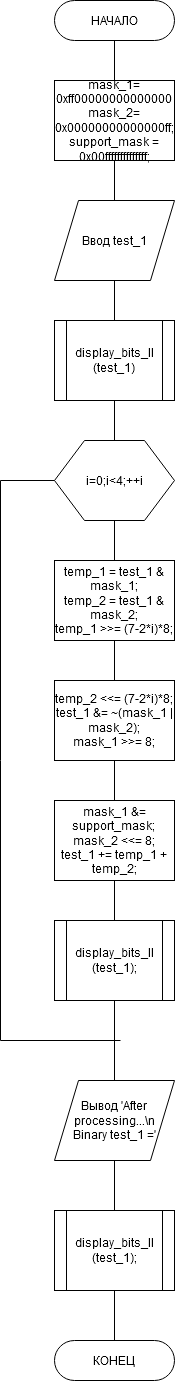
1. **Блок схемы программы.**
   1. **Подпрограмма вывода числа в двоичном виде.**

****

**Рисунок 1 – подпрограммы вывода**

**display\_bits\_ll(long long num)**

* 1. **Основная программа.**

****

**Рисунок 2 – Блок-схема основной программы**

1. **Код программы на Си.**

#include <stdio.h>

#include <stdlib.h>

//------------------------------------------------------------------------

//------------------------------------------------------------------------

void display\_bits\_ll(long long num){

    int i;

    for(i=0;i<64;++i){

        //0x8000000000000000 в двоичной версии 100...000 (63 нуля)

        printf("%d",(num & 0x8000000000000000) ? 1: 0);

        num<<=1;

        if((i+1) % 8 == 0)

            printf("  ");

    }

    printf("\n");

}

//------------------------------------------------------------------------

//------------------------------------------------------------------------

int main(void){

    //объявляем переменные

    long long test\_1,temp\_1,temp\_2,

        mask\_1=0xff00000000000000,

        mask\_2=0x00000000000000ff,

        //support\_mask необходимо для сдвига вправо

        support\_mask=0x00ffffffffffffff;

    int i;

    //меняем размер консоли

    system("mode con cols=110 lines=25");

    //ввод данных

    printf("Enter test number.\ntest\_1 (long long) = ");

    scanf("%lld",&test\_1);

    //вывод в двоичном виде

    printf("Binary test\_1 = ");

    display\_bits\_ll(test\_1);

    //обработка числа

    for(i=0;i<4;++i){

        temp\_1 = test\_1 & mask\_1;

        temp\_2 = test\_1 & mask\_2;

        temp\_1 >>= (7-2\*i)\*8;

        temp\_2 <<= (7-2\*i)\*8;

        test\_1 &= ~(mask\_1 | mask\_2);

        mask\_1 >>= 8;

        mask\_1 &= support\_mask;

        mask\_2 <<= 8;

        test\_1 += temp\_1 + temp\_2;

        display\_bits\_ll(test\_1);

    }

    //выводим в двоичном виде

    printf("After processing...\nBinary test\_1 = ");

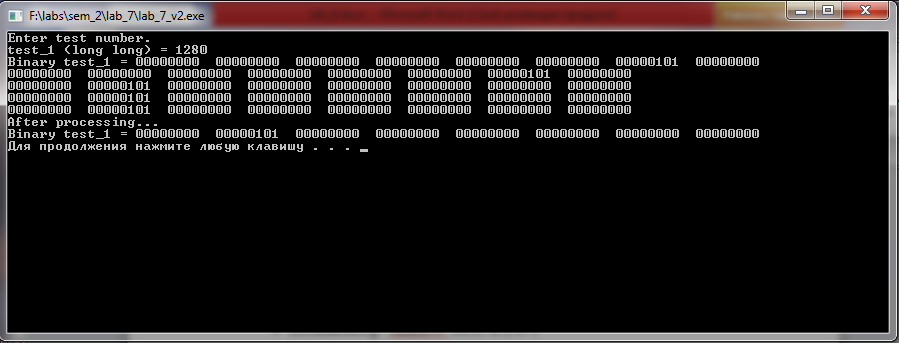
    display\_bits\_ll(test\_1);

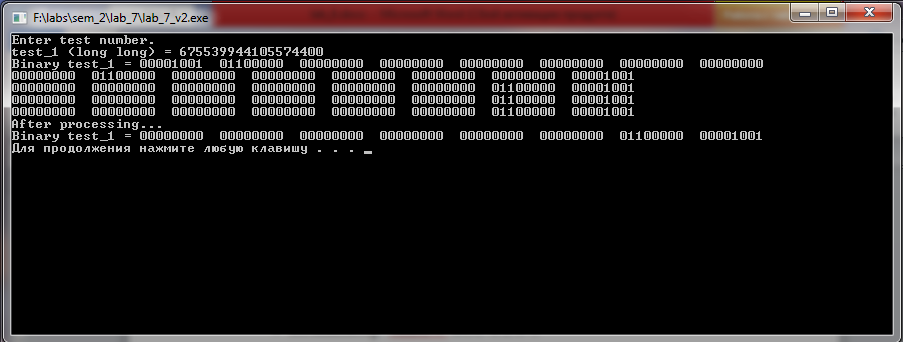
    system("pause");

    return 0;

}

1. **Проверка работы программы на тестовых примерах.**

****

****

**Рисунок 4 – Тестовый пример 2**

**Рисунок 3 – Тестовый пример 1**

1. **Вывод.**

Тестовые примеры (5 пункт отчета) доказывают правильность работы алгоритма и программы. Таким образом, программа способна в длинном числе зеркально поменять порядок байтов, используя поразрядную обработку чисел с использованием масок.