**Федеральное агентство по образованию РФ**

**Государственное образовательное учреждение высшего профессионального образования**

**Белгородский Государственный Технологический Университет им В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем

**Лабораторная работа №9**

**Побитовые операции в Си**

Выполнил:

Студент группы КБ-211

Бассараб Дмитрий

Принял:

Лукьянов А.М.

Белгород 2022

**Цель работы:** получить навыки работы с отдельными разрядами целочисленных объектов.

**Задания:**

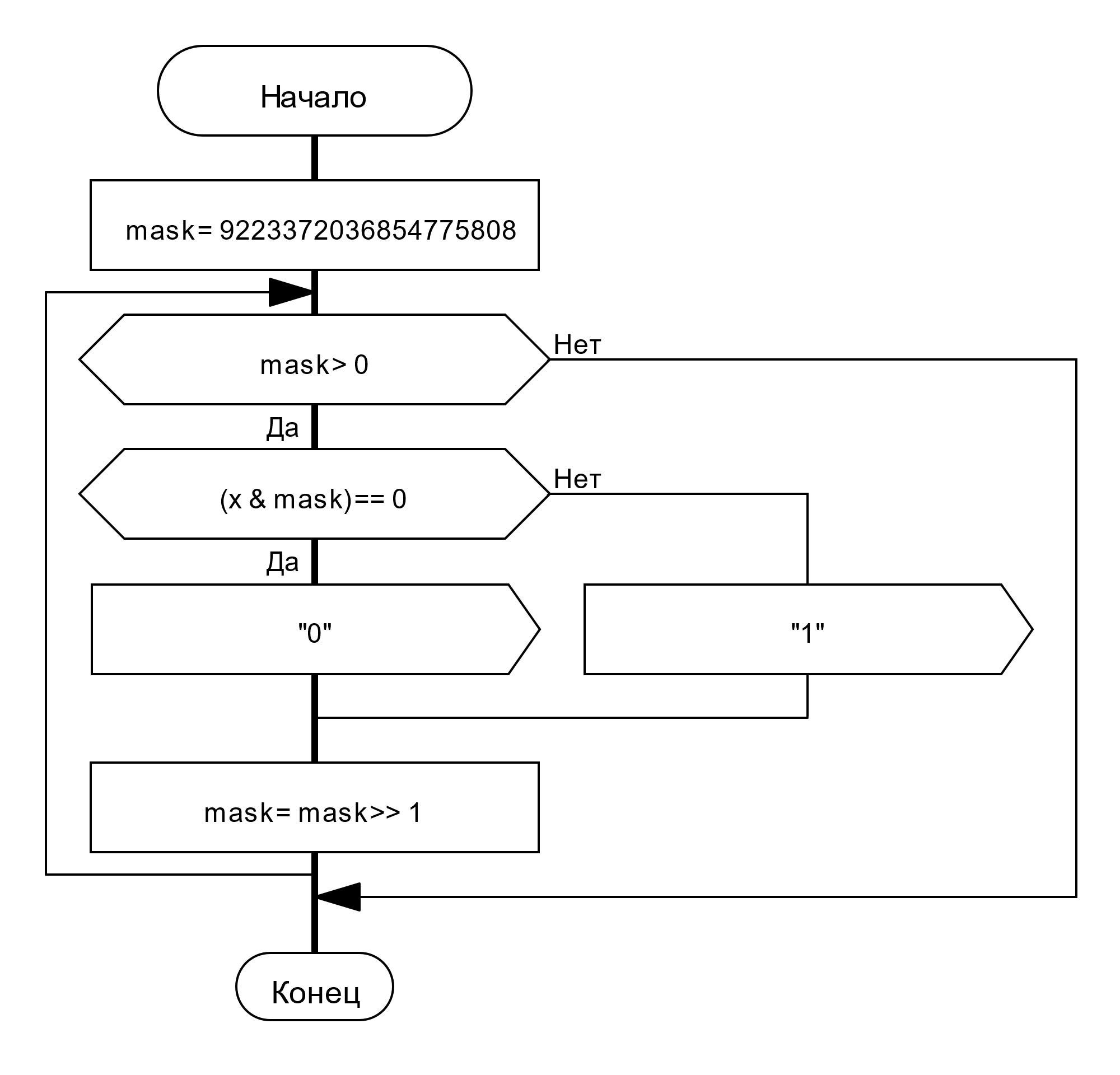
1. Изучить побитовые операции в языках Си.
2. Разработать алгоритм и составить программу для решения задачи соответствующего варианта. Исходные данные и результаты вывести в двоичной системе и в системе счисления, используемой в задаче. Во всех случаях, где возможно, использовать побитовые операции.
3. Подобрать тестовые данные.

Задание индивидуального варианта №23:

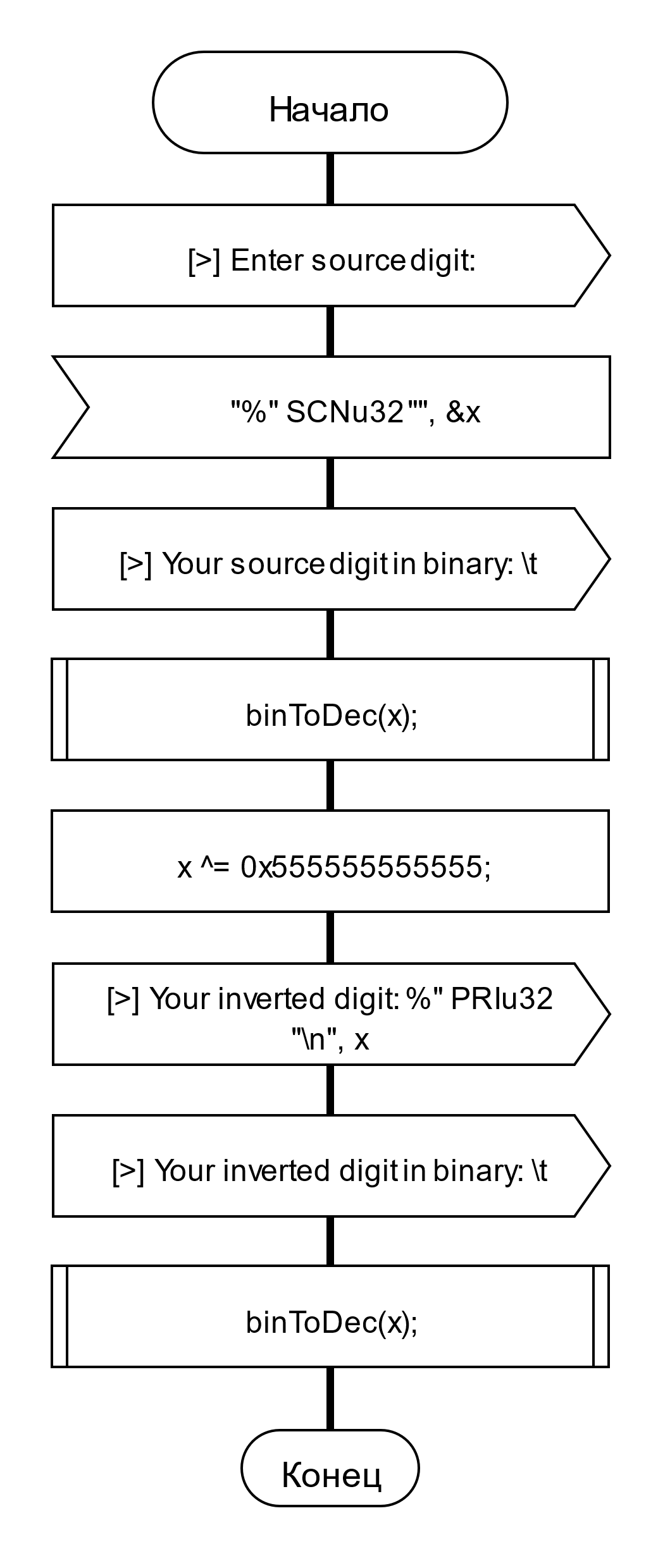
*Дано длинное целое неотрицательное число. Получить число, инвертируя его четные биты.*

**Алгоритм решения в блок-схеме:**

Блок-схема функции binToDec:



Блок-схема функции main:



**Программа на C (https://pastebin.com/VrH0H0N0):**

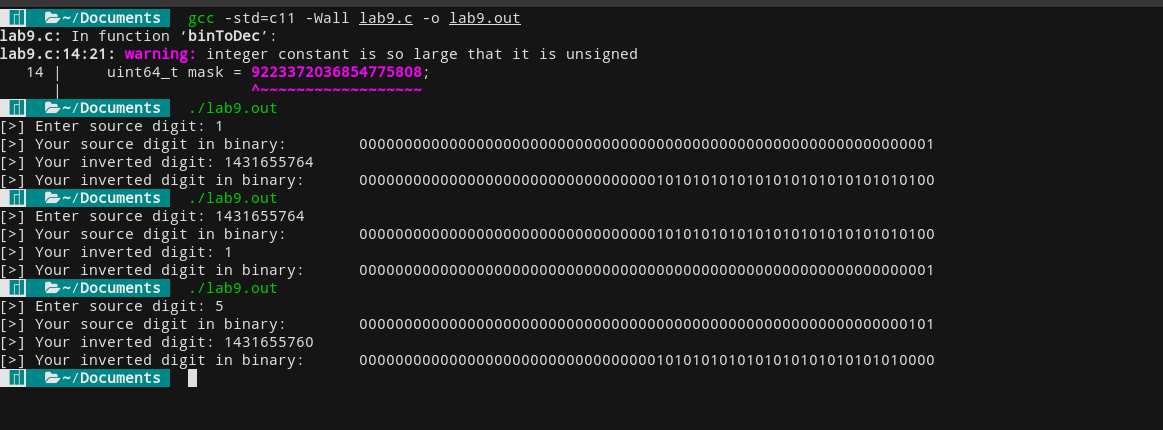
/\* Дано длинное целое неотрицательное число.  
Получить число, инвертируя его четные биты. \*/  
// >C99  
#include <stdio.h>  
#include <stdbool.h>  
#include <inttypes.h>  
  
uint8\_t SIZE\_BYTES = 32;  
uint32\_t x = 0;  
  
#define max\_uint64\_t 18446744073709551615 + 1  
  
void binToDec(uint64\_t x) {  
 uint64\_t mask = 9223372036854775808;  
 //1000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000  
 while (mask > 0) {  
 if ((x & mask) == 0)  
 printf("0");  
 else  
 printf("1");  
 mask = mask >> 1;  
 }  
}  
int main() {  
 printf("[>] Enter source digit: ");  
 scanf("%" SCNu32 "", &x);  
 printf("[>] Your source digit in binary: \t");  
 binToDec(x);  
 printf("\n");  
 x ^= 0x555555555555; // xor, 0101 in binary  
 printf("[>] Your inverted digit: %" PRIu32 "\n", x);  
 printf("[>] Your inverted digit in binary: \t");  
 binToDec(x);  
 printf("\n");  
}

**Исправленные ошибки при работе с лабораторной:**

/\* Дано длинное целое неотрицательное число.  
Получить число, инвертируя его четные биты. \*/  
// >C99  
#include <stdio.h>  
#include <stdbool.h>  
#include <inttypes.h>  
  
uint8\_t SIZE\_BYTES = 32;  
uint32\_t x = 0;  
  
#define max\_uint64\_t 18446744073709551615  
  
void binToDec(x, SIZE\_BYTES) {  
 for(SIZE\_BYTES - 1; SIZE\_BYTES >= 0; SIZE\_BYTES--) {  
 uint32\_t result = 0;  
 result = x >> SIZE\_BYTES;  
 if (result & 1)  
 printf("1");  
 else  
 printf("0");  
 }  
}   
int main() {  
 printf("[>] Enter source digit: ");  
 scanf("%" SCNu32 "", &x);  
 printf("[>] Your source digit in binary: \t");  
 binToDec(x, SIZE\_BYTES);  
 printf("\n");  
 x ^= 0x55555555;  
 printf("[>] Your inverted digit: %" PRIu32 "\n", x);  
 printf("[>] Your inverted digit in binary: \t");  
 binToDec(x, SIZE\_BYTES);  
 printf("\n");  
}

Ошибка заключалась в том, что бинарный код единички начинается с ‘1’, хотя должна с ‘0’, то есть, вместо “1000000000000000000000000000000000000000000000000000000000000001” справедливо будет “0000000000000000000000000000000000000000000000000000000000000101". Поэтому, я просто переписал новую функцию и использовал 32-байтный тип, вместо 64-байтного из-за того, что нет большего типа данных, не используя длинной арифметики.

**Тестовые данные:**

****

**Вывод:** в ходе выполнения лабораторной работы я получил навыки работы с отдельными разрядами целочисленных объектов, научился различать порядок байтов (big-endian и little-endian), подробно изучил библиотеку <inttypes.h>.