МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М.Є. Жуковського
«Харківський авіаційний інститут»

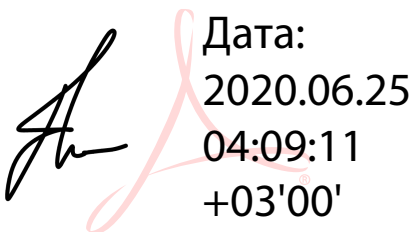
Факультет радіоелектроніки, комп'ютерних систем та інфокомунікацій

Кафедра комп'ютерних систем, мереж і кібербезпеки

Звіт

з *Навчальної практики*
(назва дисципліни)

на тему «Зворотний і додатковий код числа»



Виконав: студент *1* курсу групи № *515а*
Пахарь Б.Є.
(прізвище й ініціали студента)

Керівник: *доцент, к.т.н. Бабешко Є.В.*
(посада, науковий ступінь, прізвище й ініціали)

Харків – 2020

Зміст

1. Мета роботи

1) Опанувати навичками використання розподіленої системи керування версіями для проектування та командної роботи.

3) Виконати поставлену задачу.

2. Реферат

При дослідженні сфери комп’ютерних технологій двійковий код являється одною з найголовніших тем вивчення схеми роботи техніки та комп’ютерів. Його частиною є зворотний код, а також додатковий код.

Зворотний код – метод обчислювальної математики, який дозволяє вирахувати одне число від другого, виконуючи тільки одну операцію сумми над натуральними числами. Раніше метод використовувався в механічних калькуляторах (арифмометрах). Багато ранніх комп’ютерів, включаючи CDC 6600, LINC, PDP-1 и UNIVAC 1107, використовували зворотний код. Більшість сучасних комп’ютерів використовують додатковий код.

Доповняльний код – найпоширеніший спосіб подання від'ємних чисел у комп'ютерах. Дозволяє замість команди віднімання використовувати команду додавання, для знакових і беззнакових чисел, що зменшує вимоги до архітектури комп'ютера.

Далі буде продемонстровано самостійно розроблена програма по розрахунку зворотного та додаткового коду на мові С, з алгоритмами й поясненням. А також виконане тестування програми, з табличними даними та скріншотами.

3. Завдання

3.1 Варіант № 4

Умова: Знайти зворотний і додатковий код числа.

3.2 Метод рішення

Поставлену задачу було вирішено виконати за допомогою одновимірних масивів цілого типу, та групою циклів for (;;), while. Окремі частини програми організовані у вигляді функцій.

3.3 Алгоритми

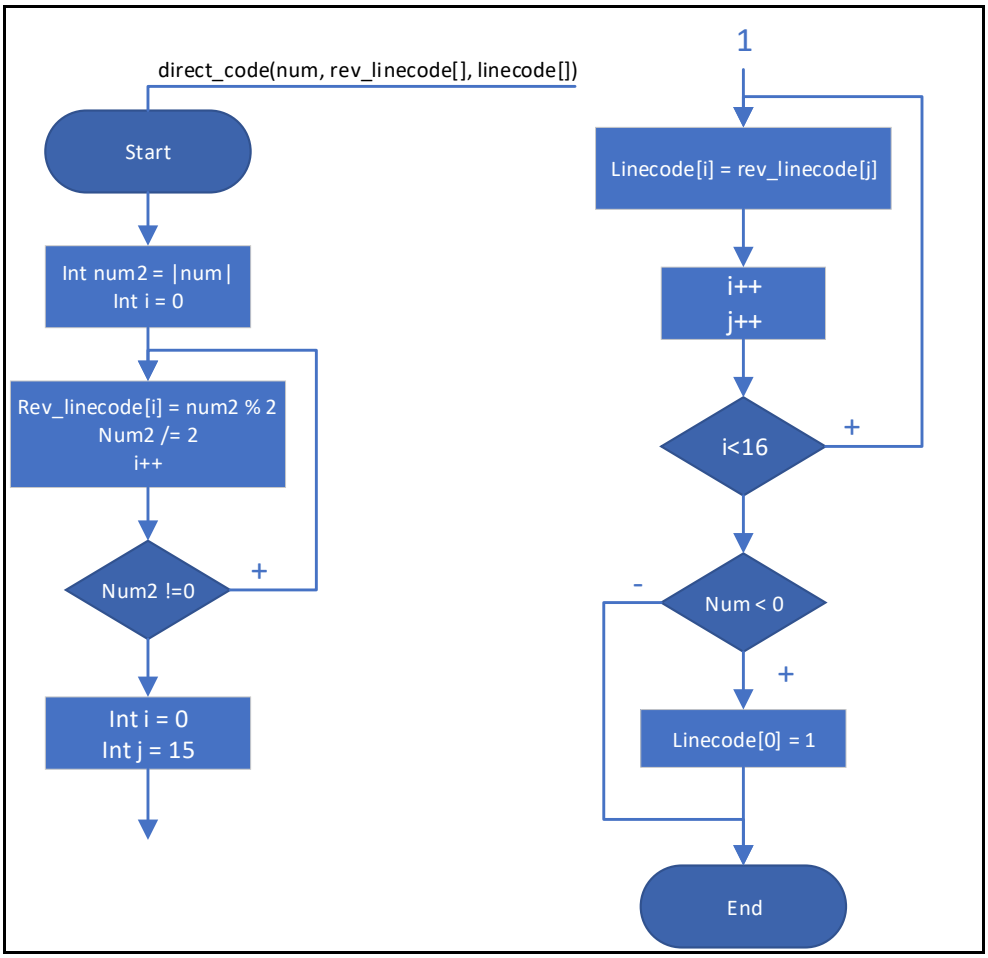


Рисунок 1.1 – Функція по знаходженню прямого коду числа.

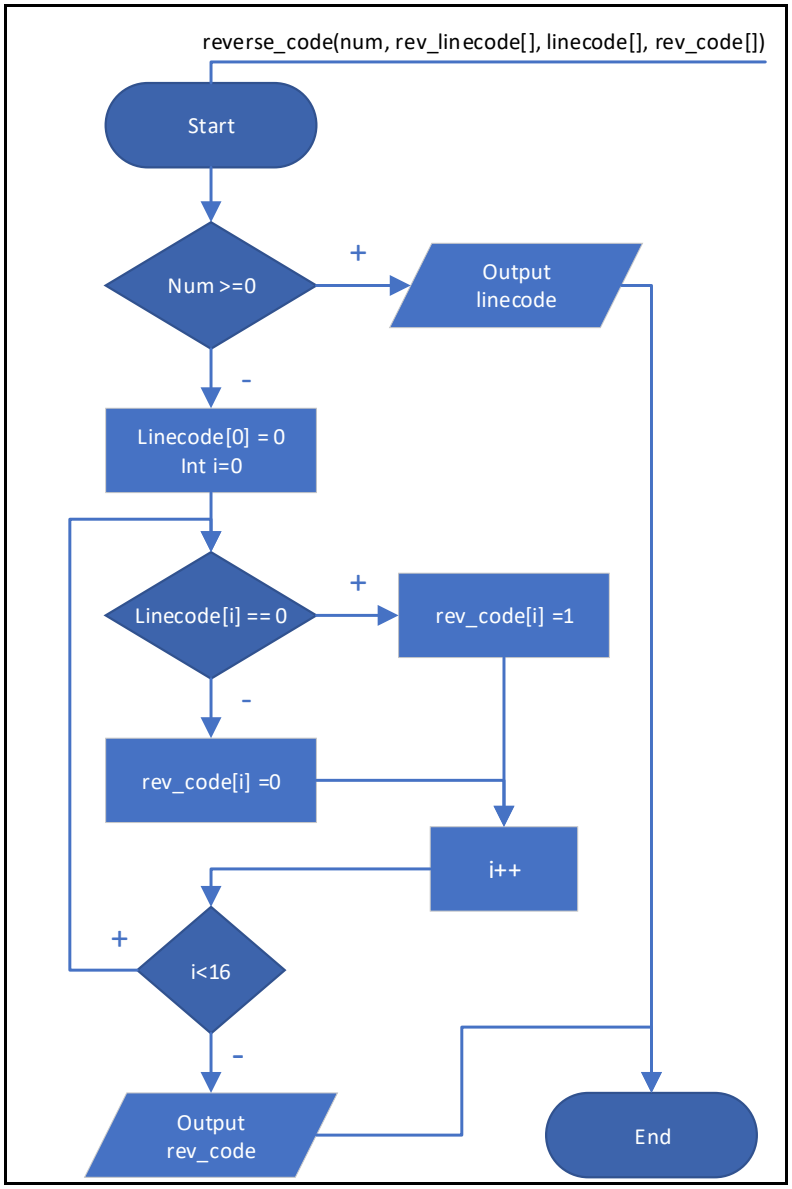


Рисунок 1.2 – Функція по знаходженню зворотного коду.

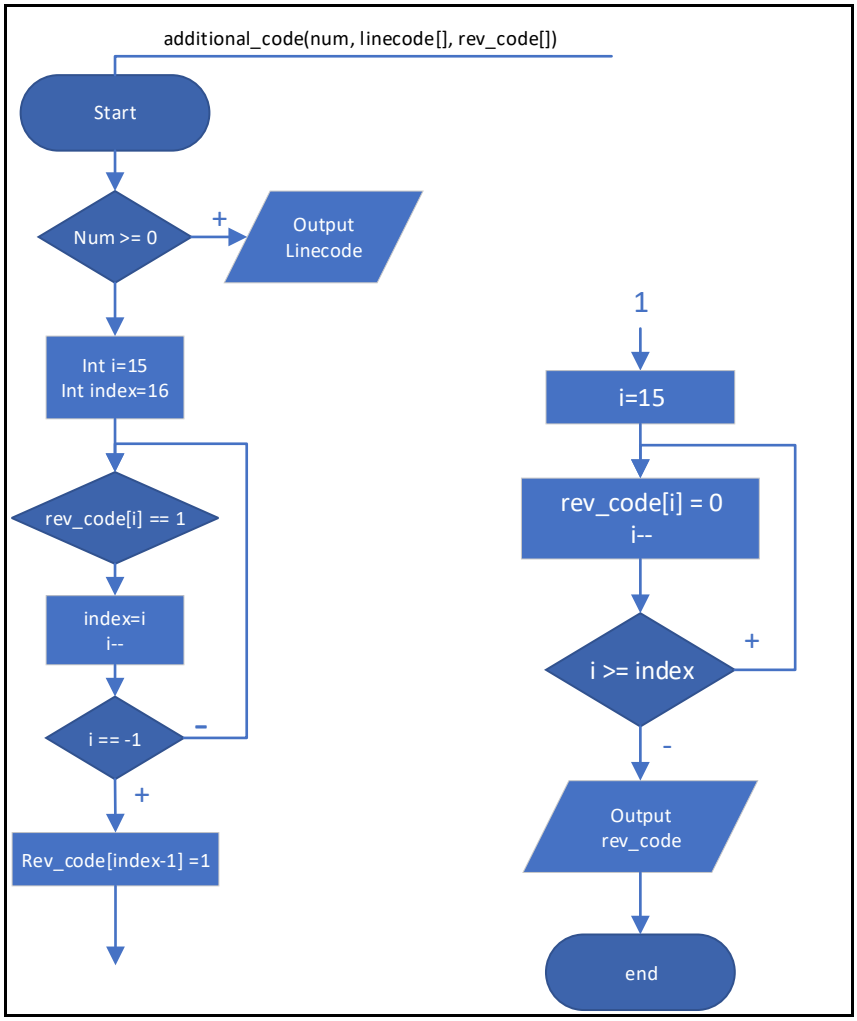


Рисунок 1.3 – Функція по знаходженню додаткового коду.

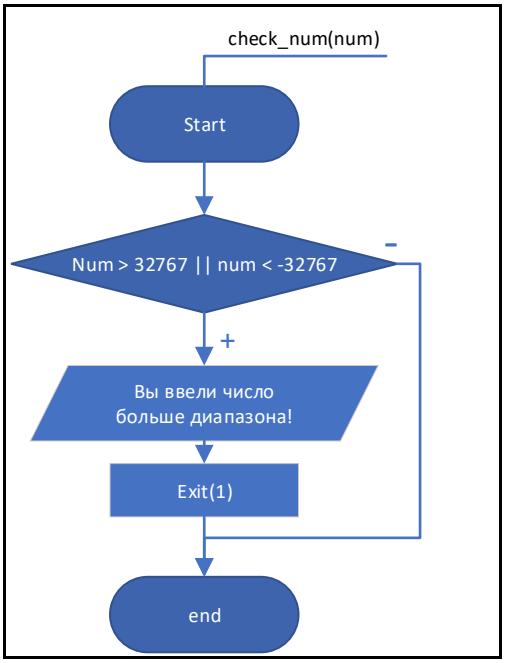


Рисунок 1.4 – Функція по перевірці коректності вхідних даних.

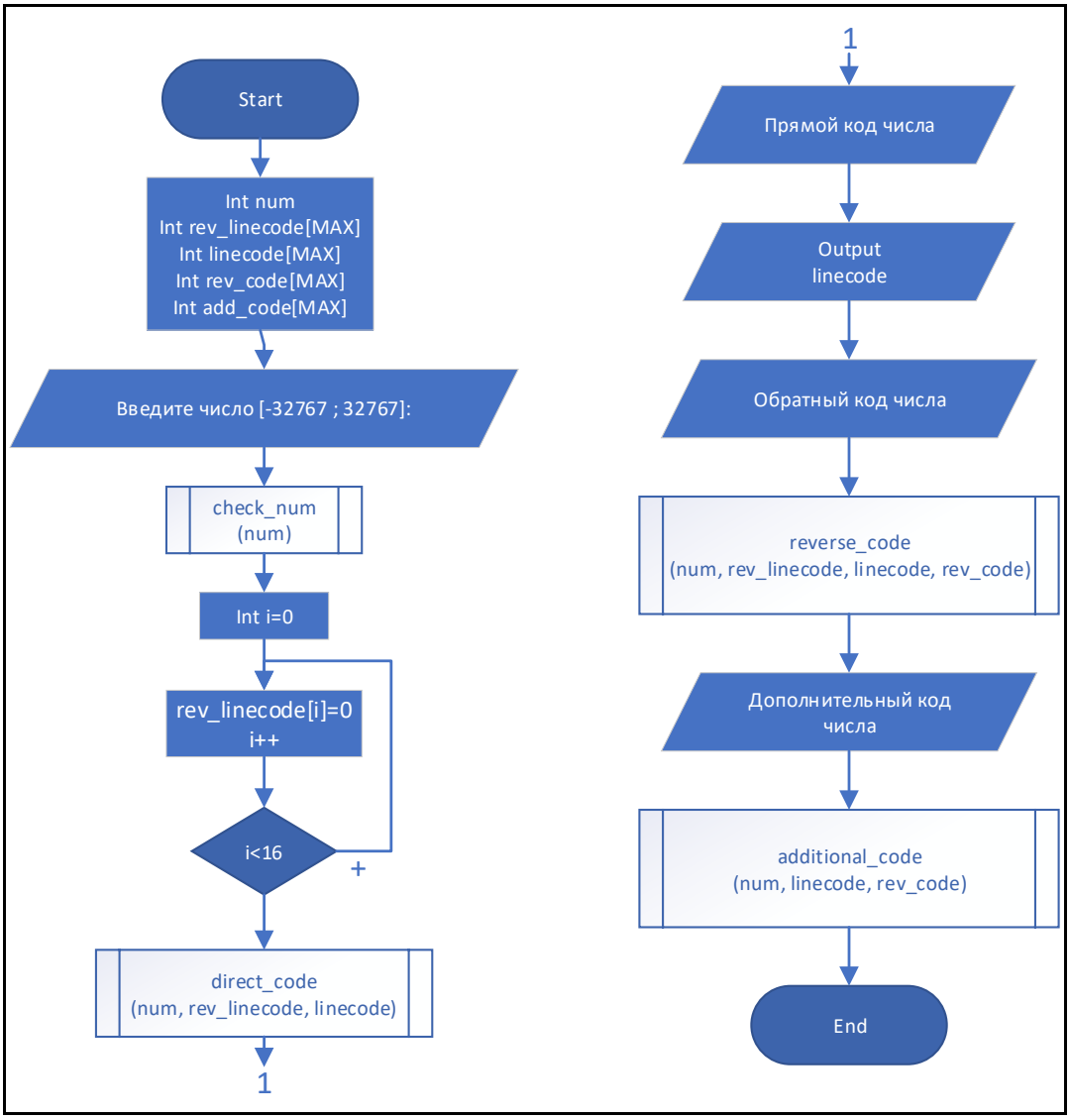


Рисунок 1.5 – Головна функція Main ().

3.4 Код програми

```
/**
* @author Pahar B.E., gr.515a
* @date 24.06.2020
* @brief Practice
*/
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
#include <math.h>
#define MAX 16 //количество бит

int direct_code(int, int[], int[]);
int check_num(int);
int reverse_code(int, int[], int[], int[]);
int additional_code(int, int[], int[]);

int main()
{
    int num;
    int rev_linecode[MAX]; //пр. код наоборот
    int linecode[MAX]; //прямой код
    int rev_code[MAX]; //обратный код
    int add_code[MAX]; //дополнительный код
    setlocale(LC_ALL, "Rus");
    printf("Введите число [-32767 ; 32767]: "); //вводим число
    scanf_s("%d", &num);
    check_num(num); //проверка на корректность входных данных

    for (int i = 0; i < 16; i++)
        rev_linecode[i] = 0; //заполняем массив нулями 0000 0000

    //вычисление прямого кода числа
    direct_code(num, rev_linecode, linecode);
    printf("Прямой код числа:          ");
    for (int i = 0; i < 16; i++) {
        printf("%d", linecode[i]);
        if (i == 3 || i == 7 || i == 11) printf(" ");
    }
    //вычисление обратного кода числа
    printf("\nОбратный код числа:        ");
    reverse_code(num, rev_linecode, linecode, rev_code);

    //вычисление дополнительного кода
    printf("\nДополнительный код числа: ");
    additional_code(num, linecode, rev_code);
    return 0;
}
/* Вычисление прямого кода числа
*  @ param num
*  @ param rev_linecode[]
*  @ param linecode[]
*  @ return 0 - функция ничего не возвращает
*/
int direct_code (int num, int rev_linecode[], int linecode[]) {
    int num2 = fabs(num);
    int i = 0;
    while (num2 != 0) {
        rev_linecode[i] = num2 % 2;
        num2 /= 2;
        i++;
    }
    //reverse
```

```
    for (int i = 0, j = 15; i < 16; i++, j--)
        linecode[i] = rev_linecode[j];
    if (num < 0) linecode[0] = 1;
    return 0;
}

/* Вычисление обратного кода числа
*  @ param num
*  @ param rev_linecode[]
*  @ param linecode[]
*  @ param rev_code[]
*  @ return 0 - функция ничего не возвращает
*/
int reverse_code (int num, int rev_linecode[], int linecode[], int rev_code[]) {
    if (num >= 0) {
        for (int i = 0; i < 16; i++) {
            printf("%d", linecode[i]);
            if (i == 3 || i == 7 || i == 11) printf(" ");
        }
    }
    else {
        linecode[0] = 0; //new
        for (int i = 0; i < 16; i++) {
            if (linecode[i] == 0)
                rev_code[i] = 1;
            else
                rev_code[i] = 0;
        }
        for (int i = 0; i < 16; i++) {
            printf("%d", rev_code[i]);
            if (i == 3 || i == 7 || i == 11) printf(" ");
        }
    }
    return 0;
}

/* Вычисление дополнительного кода числа
*  @ param num
*  @ param rev_linecode[]
*  @ param linecode[]
*  @ return 0 - функция ничего не возвращает
*/
int additional_code (int num, int linecode[], int rev_code[]) {
    if (num >= 0) {
        for (int i = 0; i < 16; i++) {
            printf("%d", linecode[i]);
            if (i == 3 || i == 7 || i == 11) printf(" ");
        }
    }
    else {//дополнительный код отрицательного числа
        int i = 15;
        int index = 16; //13
        while (rev_code[i] == 1) {
            index = i;
            i--;
            if (i == -1) break;
        }
        rev_code[index - 1] = 1;

        for (i = 15; i >= index; i--) {
            rev_code[i] = 0;
        }
        for (int i = 0; i < 16; i++) {
            printf("%d", rev_code[i]);
            if (i == 3 || i == 7 || i == 11) printf(" ");
```

```
        }
    }
    return 0;
}

/* Проверка корректности ввода числа
*  @ param num
*  @ return 0 - функция ничего не возвращает
*/
int check_num (int num) {
    if (num > 32767 || num < -32767) {
        printf("Вы ввели число больше диапазона!");
        exit(1);
    }
    return 0;
}
```

4. Тестування програми

Тест №1. Перевірка на правдивість результату

Тест №	Вхідні данні	Вихідні данні	Результат
1	0	Введите число [-32767 ; 32767]: 0 Прямой код числа: 0000 0000 0000 0000 Обратный код числа: 0000 0000 0000 0000 Дополнительный код числа: 0000 0000 0000 0000	Успіх
2	5	Введите число [-32767 ; 32767]: 5 Прямой код числа: 0000 0000 0000 0101 Обратный код числа: 0000 0000 0000 0101 Дополнительный код числа: 0000 0000 0000 0101	Успіх
3	36	Введите число [-32767 ; 32767]: 36 Прямой код числа: 0000 0000 0010 0100 Обратный код числа: 0000 0000 0010 0100 Дополнительный код числа: 0000 0000 0010 0100	Успіх
4	145	Введите число [-32767 ; 32767]: 145 Прямой код числа: 0000 0000 1001 0001 Обратный код числа: 0000 0000 1001 0001 Дополнительный код числа: 0000 0000 1001 0001	Успіх
5	256	Введите число [-32767 ; 32767]: 256 Прямой код числа: 0000 0001 0000 0000 Обратный код числа: 0000 0001 0000 0000 Дополнительный код числа: 0000 0001 0000 0000	Успіх
6	4678	Введите число [-32767 ; 32767]: 4678 Прямой код числа: 0001 0010 0100 0110 Обратный код числа: 0001 0010 0100 0110 Дополнительный код числа: 0001 0010 0100 0110	Успіх

7	32767	Введите число [-32767 ; 32767]: 32767 Прямой код числа: 0111 1111 1111 1111 Обратный код числа: 0111 1111 1111 1111 Дополнительный код числа: 0111 1111 1111 1111	Успіх
8	-36	Введите число [-32767 ; 32767]: -36 Прямой код числа: 1000 0000 0010 0100 Обратный код числа: 1111 1111 1101 1011 Дополнительный код числа: 1111 1111 1101 1100	Успіх
9	-875	Введите число [-32767 ; 32767]: -875 Прямой код числа: 1000 0011 0110 1011 Обратный код числа: 1111 1100 1001 0100 Дополнительный код числа: 1111 1100 1001 0101	Успіх
10	-32767	Введите число [-32767 ; 32767]: -32767 Прямой код числа: 1111 1111 1111 1111 Обратный код числа: 1000 0000 0000 0000 Дополнительный код числа: 1000 0000 0000 0001	Успіх

Тест №2. Перевірка на коректність вхідних даних.

Тест №	Вхідні данні	Вихідні данні	Результат
1	32768	Введите число [-32767 ; 32767]: 32768 Вы ввели число больше диапазона!	Успіх
2	-32768	Введите число [-32767 ; 32767]: -32768 Вы ввели число больше диапазона!	Успіх
3	-99999	Введите число [-32767 ; 32767]: -99999 Вы ввели число больше диапазона!	Успіх
4	40000	Введите число [-32767 ; 32767]: 40000 Вы ввели число больше диапазона!	Успіх

5. Висновок

Під час виконання завдання я досить в значної мірі опанував навичками використання розподіленої системи керування версіями для проектування та командної роботи.

Git – розподілена система зберігання контролю версій програмного забезпечення. Вона відрізняється від численних аналогів можливістю зберігати інформацію в репозиторії на жорсткому диску, ефективно відстежувати будь-які зроблені зміни, відкочуватися на один або кілька кроків назад, якщо в цьому виникне необхідність.

А також я вдосконалив свої знання на практиці в мові програмування С.