НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ ТА ПРОГРАМУВАННЯ»

«Програмування ч.1»

*Звіт з лабораторної роботи №8*

*Тема: «Вступ до блок-схем алгоритмів»*

Виконав:

ст. гр. КІТ-120А

Старовойтов Н.А.

Перевірив:

Челак В.В.

Харків – 2020

***Мета:*** Отримати навички роботи з блок-схемами алгоритмів та їх застосування.

***Індивідуальне завдання***

Робота на оцінку “відмінно”.

Завдання №1: «Визначити найбільший спільний дільник для двох заданих чисел».

Завдання №2: «Дано двовимірний масив з N \* N цілих чисел. Виконати циклічне зрушення елементів рядків масиву в напрямку справа наліво (перший елемент рядка повинен переміститися в ії кінець)».

Завдання №3: «Реалізувати функцію, що визначає, скільки серед заданої послідовності чисел таких пар, у котрих перше число менше наступного, використовуючи функцію з варіативною кількістю аргументів».

**Опис програми**

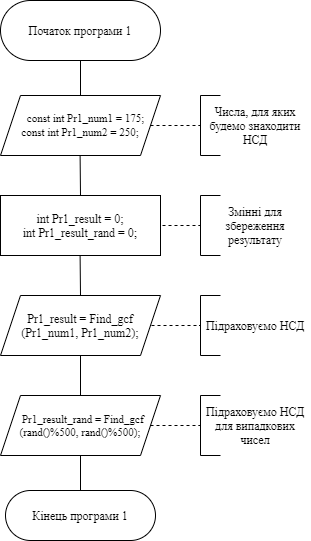
**Алгоритм вирішення поставленої задачі**

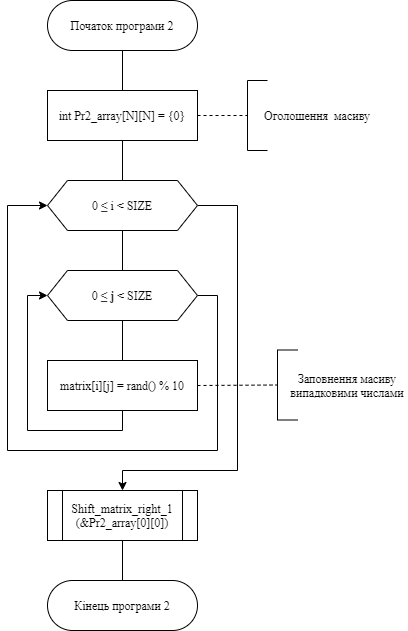
Для вирішення Завдання №1 використовується програмна реалізація алгоритма Євкліда. Функція Find\_gcf приймає два числа та повертає їх НСД. Викликається функція двічі – з заданими та згенерованими випадково числами.

Для вирішення Завдання №2 використовується покроковий зсув кожного рядка разом з одночасним записом першого елемента рядка у масив temp. Після зсуву останньому елементу рядка надається значення, що було збережено у масиві temp. Операція повторюється N разів. Всі ці дії виконуються у функції Shift\_matrix\_right\_1. Для заповнення масиву було використано генератор псевдовипдакових чисел. Результат зберігається у тому ж масиві, що й первісні дані.

Для вирішення Завдання №3 використовується функція Num\_of\_ordered\_pairs з варіативною кількістю аргументів. Вона приймає число – кількість наступних аргументів, та, власне, ці аргументи. Повертає число – кількість пар чисел, що відповідають умові задачі. Викликається функція двічі – з заданими та згенерованими випадково числами.

**Схеми алгоритмів функцій**

****

****

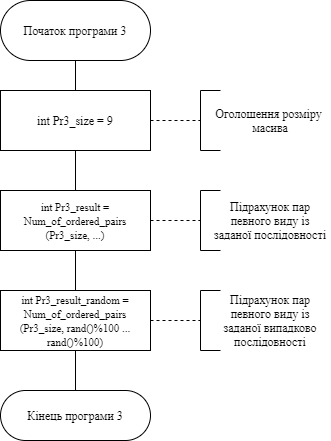
****

Рисунок 1 — Функція main, яка складається з трох програм

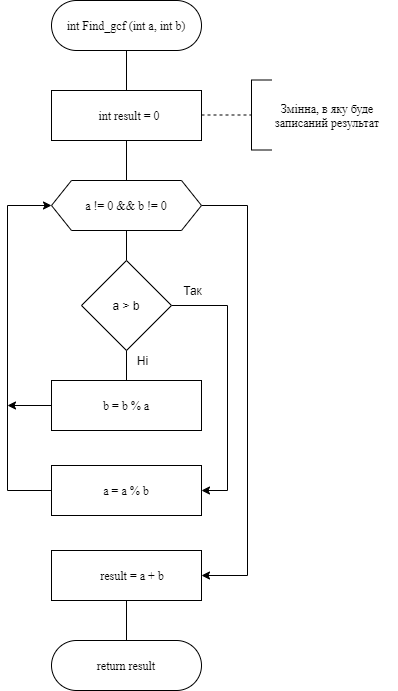
****

Рисунок 2 — Функція Find\_gcf

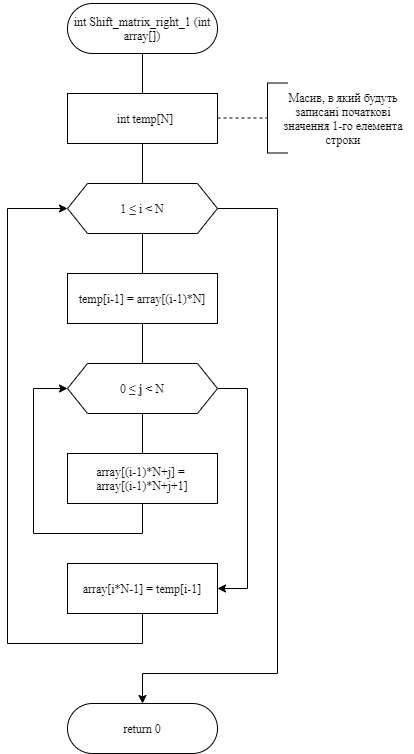
****

Рисунок 3 — Функція Num\_of\_ordered\_pairs

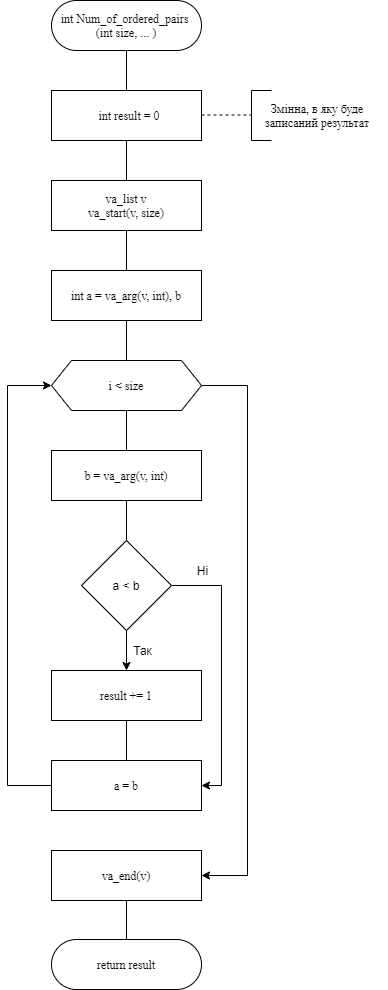
****

Рисунок 4 — Функція Shift\_matrix\_right\_1

**Текст програми**

#include <stdlib.h>

#include <stdarg.h>

#include <time.h>

#define N 3

int Calculation1 (int a, int b){

int result = 0;

while (a != 0 && b != 0){

if (a > b) a = a % b;

else b = b % a;

}

result = a + b;

return result;

}

int Calculation2 (int array[]) {

int temp[N];

for (int i = 1; i <= N; i++) {

temp[i-1] = array[(i-1)\*N];

for (int j = 0; j < N; j++) {

array[(i-1)\*N+j] = array[(i-1)\*N+j+1];

}

array[i\*N-1] = temp[i-1];

}

return 0;

}

int Calculation3 (int size, ... )

{

int result = 0;

int i = 1;

va\_list v;

va\_start(v, size);

int a = va\_arg(v, int), b;

while (i < size){

b = va\_arg(v, int);

if (a < b) result += 1;

a = b;

i++;

}

va\_end(v);

return result;

}

int main(){

///// Program 1 /////

srand(time(NULL));

const int Pr1\_num1 = 175;

const int Pr1\_num2 = 250;

int Pr1\_result = 0, Pr1\_result\_rand = 0;

Pr1\_result = Calculation1 (Pr1\_num1, Pr1\_num2);

Pr1\_result\_rand = Calculation1 (rand()%500, rand()%500);

///// Program 2 /////

int Pr2\_array[N][N] = {0};

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

Pr2\_array[i][j] = rand() % 10;

}

}

Calculation2 (&Pr2\_array[0][0]);

///// Program 3 /////

int Pr3\_size = 9;

int Pr3\_result = Calculation3 (Pr3\_size, 2, 2, 8, 7, 9, 3, 12, 5, 99);

int Pr3\_result\_random = Calculation3 (Pr3\_size, rand()%100, rand()%100, rand()%100, rand()%100, rand()%100, rand()%100, rand()%100, rand()%100, rand()%100);

return 0;

}

**Результати роботи програми**

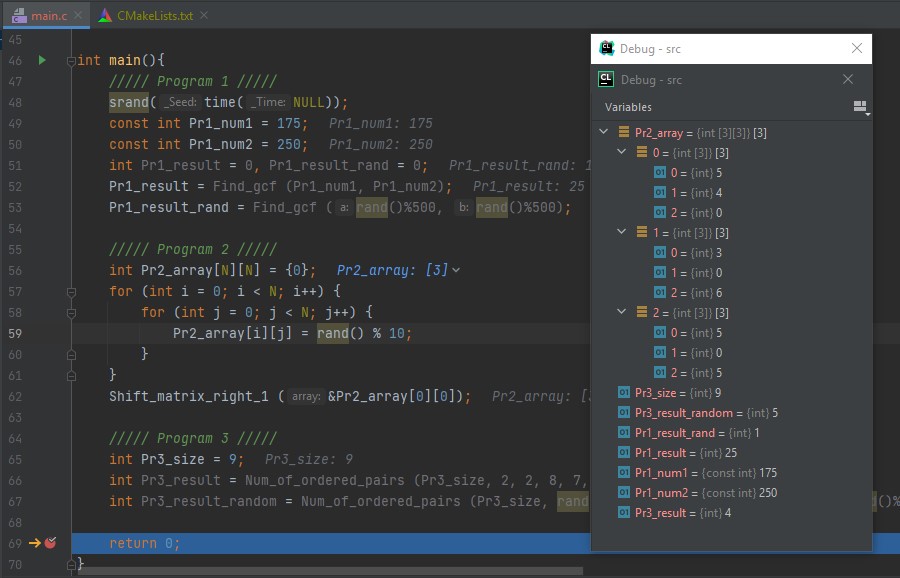


Рисунок 5 — Результат успішного виконання програми

**Висновки**

Під час виконання даної лабораторної роботи було отримано навички роботи з блок-схемами алгоритмів та їх застосуванням.