Чернівецький національний університет імені Юрія Федьковича

Навчально-науковий інститут фізико-технічних та комп’ютерних наук

Кафедра комп’ютерних наук

Звіт

про виконання лабораторної роботи No1

«Масиви. Введення – виведення в С++. Файлові потоки С++.»

з дисципліни

«Об’єктно-орієнтоване програмування»

Виконав: студент 141 групи

Обершт Валентина Василівна

Перевірив: канд.фіз.-мат. наук,

доцент Лазорик В.В.

Оцінка:

Дата захисту: 27.02.2023

Чернівці 2023

Тема: Масиви. Введення – виведення в С++. Файлові потоки С++.

Мета: Вивчення засобів мови С++ роботи з динамічними масивами, операціями

простого консольного введення-виведення та файлового консольного введення-виведення.

# Завдання до лабораторної роботи

Завдання 1. Задано одновимірний масив А розміру N. Першу половину елементів масиву помножити на 2, а другу - на 3.

Завдання 2. Знайти номер останнього мінімального елемента серед парних додатних

елементів, що лежать правіше першого від’ємного елемента.

Завдання 3. На площині задано множину точок з координатами (x1, y1), (x2, y2),…,(xn, yn), n ≤ 200 і координати вершин трикутника A(xa, ya), B(xb, yb), C(xc, yc). Розробити програму, яка обчислює кількість точок заданої множини, що належать ∆АВС.

Адреса репозиторію, у якому виконувалось завдання – https://github.com/ValentinaOb/Lab1

# Завдання 1

## Опис вхідних та вихідних даних

Вхідні дані:

i - індекс

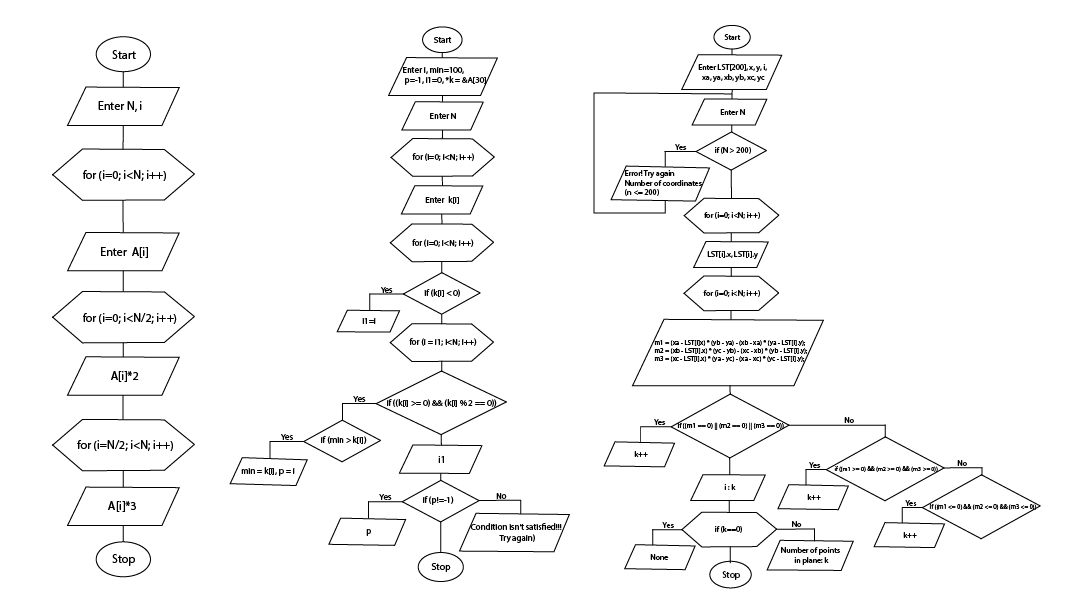
N – к-ть елементів масиву А

A[30] – масив дійсних чисел з максимальною кількістю елементів 30

Вихідні дані:

А[N] - масив дійсних чисел, перша частина якого множиться на 2, а другу – на 3, з введеною нами кількістю елементів на початку

## Схема алгоритму розв’язання завдання



## Лістинг програми

int main()

{

int i, N;

int A[30];

cout << "N: ";

cin >> N;

cout << "A: ";

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

cin >> A[i];

}

printf("You have entered : ");

for (i = 0; i < N / 2; i++) printf("%d ", (A[i] \* 2));

for (i = N / 2; i < N; i++) printf("%d ", (A[i] \* 3));

cout << "\n";

return 0;

}

## Розробка тестів та результати тестування

1. Вхідні дані:

N=5, A[5] = {7, 9, 1, 0, 4}

Вихідні:

A[5] = {14, 18, 3, 0, 12}

1. Вхідні дані:

N=12, A[12] = {7, 1, 5, 10, 4, 6, 9, 11, 2, 1, 4, 9}

Вихідні:

A[12] = { 14, 2, 10, 20, 8, 12, 27, 33, 6, 3, 12, 27}

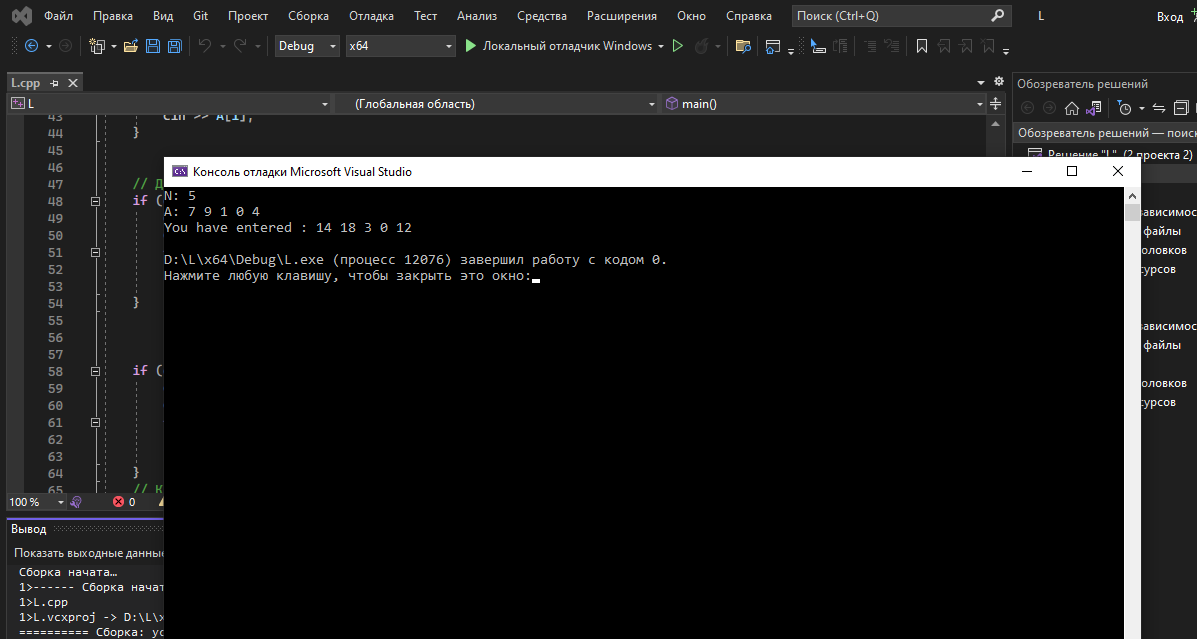
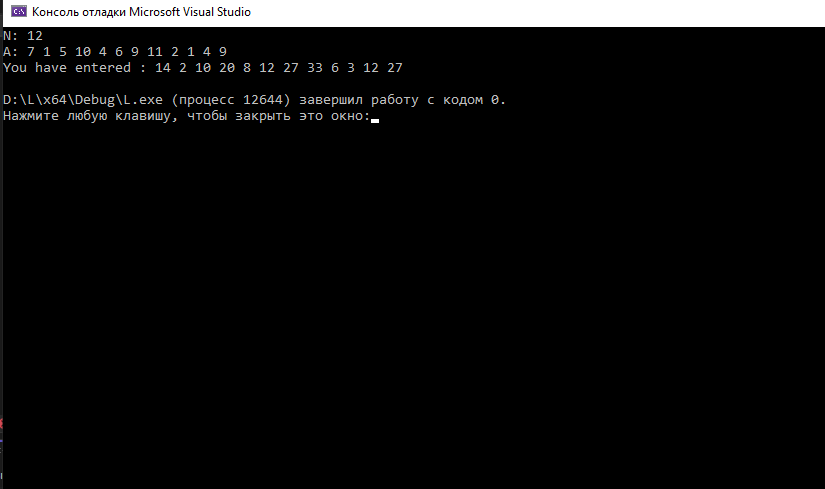
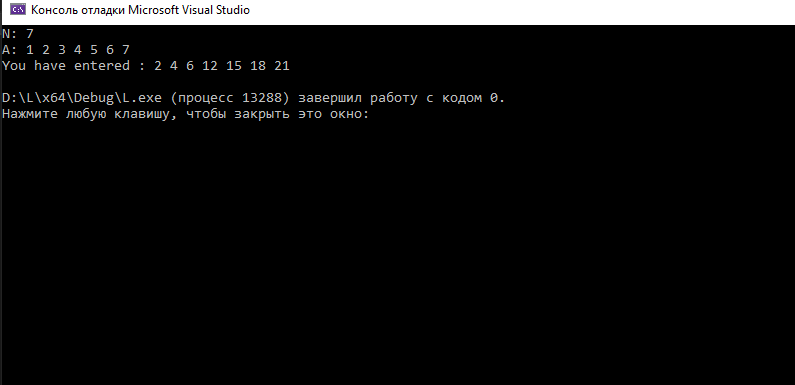
1. Вхідні дані:

N=7, A[7] = {1, 2, 3, 4, 5, 6, 7}

Вихідні:

A[7] = { 2, 4, 6, 12, 15, 18, 21}

## Лістинг (скріншоти) вхідних та вихідних даних і результатів розв’язання завдання

1. 
2. 
3. 

## Аналіз отриманих результатів

1. Вхідні: A[5] = {7, 9, 1, 0, 4}

Процес: A[5] = {7\*2, 9\*2, 1\*3, 0\*3, 4\*3}

Вихідні: A[5] = {14, 18, 3, 0, 12}

1. Вхідні: A[12] = {7, 1, 5, 10, 4, 6, 9, 11, 2, 1, 4, 9}

Процес: A[12] = {7\*2, 1\*2, 5\*2, 10\*2, 4\*2, 6\*2, 9\*3, 11\*3, 2\*3, 1\*3, 4\*3, 9\*3}

Вихідні: A[12] = { 14, 2, 10, 20, 8, 12, 27, 33, 6, 3, 12, 27}

1. Вхідні: A[7] = {1, 2, 3, 4, 5, 6, 7}

Процес: A[7] = {1\*2, 2\*2, 3\*2, 4\*3, 5\*3, 6\*3, 7\*3}

Вихідні: A[7] = { 2, 4, 6, 12, 15, 18, 21}

# Завдання 2

## Опис вхідних та вихідних даних

Вхідні дані:

i - індекс

N – к-ть елементів масиву А

A[30] – масив дійсних чисел з максимальною кількістю елементів 30

min = 100 – мінімальне число

p = -1 – позиція шуканого нами елемента, початкове значення якого = -1, щоб далі використати його у наших цілях

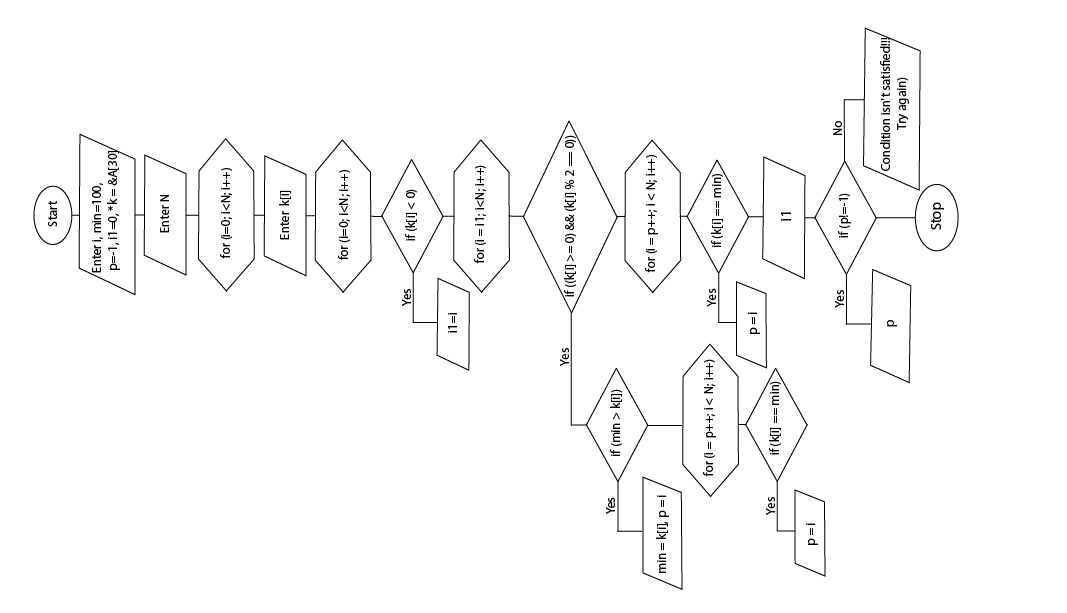
i1 = 0 – позиція, на якій знаходиться перше від’ємне значення масиву

\*k – вказівник на масив A[30]

Вихідні дані:

p – позиція шуканого нами елемента

## Схема алгоритму розв’язання завдання



## Лістинг програми

int main()

{

int N, i, min = 100, p = -1, i1 = -1;

int\* k;

k = new int[30];

cout << "Input N : ";

cin >> N;

cout << "Input A : ";

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

cin >> k[i];

}

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

cout << k[i] << " ";

}

cout << endl;

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

if (k[i] < 0) { i1 = i; break; }

}

if(i1==-1){

cout << "Condition isn't satisfied!!! Try again) \n";

return 0;

}

for (i = i1; i < N; i++) {

if ((k[i] >= 0) && (k[i] % 2 == 0)) {

if (min > k[i]) {

min = k[i]; p = i;

}

}

}

for (i = i1; i < N; i++) {

if ((k[i] >= 0) && (k[i] % 2 == 0)) {

for (i = p++; i < N; i++) {

if (k[i] == min) p = i;

}

}

}

cout << "The position of the negative element (I1): " << i1 << endl;

if (p != -1) {

cout << "P: " << p << endl;

}

else {

cout << "Condition isn't satisfied!!! Try again) \n";

}

if (out2.is\_open()) {

out2 << "The position: " << p << endl;

out2 << "A: ";

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

out2 << k[i] << " ";

}

out2 << "\nN: " << N << endl;

}

delete[] k;

return 0;

}

## Розробка тестів та результати тестування

1. Вхідні дані: N = 7, A[7] = {2, 0, -1, -2, 2, 7, 2}

Вихідні: i1 = 2, p = 6

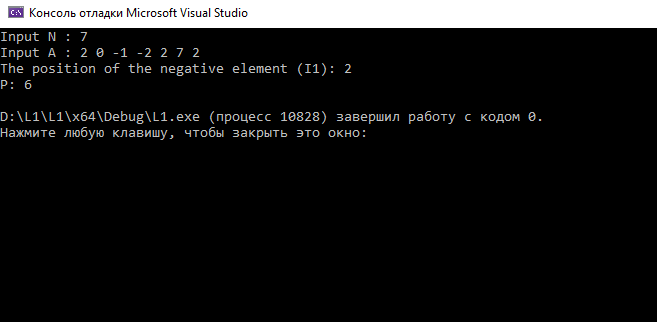
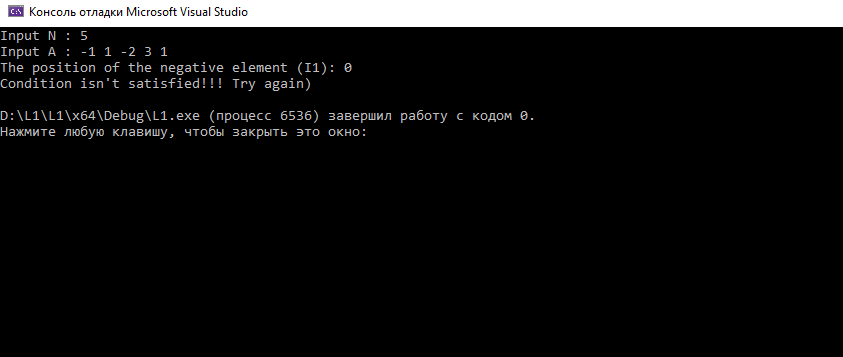
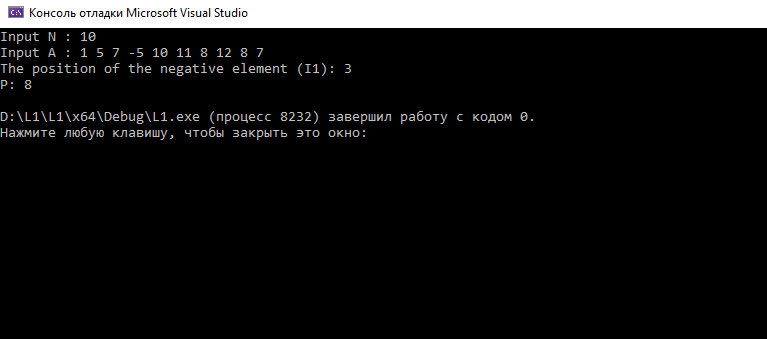
1. Вхідні дані: N = 5, A[5] = {}

Вихідні: i1 = 0,

1. Вхідні дані: N = 10, A[10] = {1, 5, 7, -5, 10, 11, 8, 12, 8, 7}

Вихідні: i1 = 3, p = 8

## Лістинг (скріншоти) вхідних та вихідних даних і результатів розв’язання завдання

1. 
2. 
3. 

## Аналіз отриманих результатів

Зелений – від’ємний елемент

Червоний – розміщений до 1-го від’ємного значення

Голубий – непарний елемент

Оранжевий – не мінімальний елемент, який відповідає усім іншим вимогам

Жовтий – не останній елемент, якій відповідає усім вимогам

1. Вхідні: N = 7, A[7] = {2, 0, -1, -2, 2, 7, 2}

Процес: 2, 0, -1, -2, 2, 7, 2

Вихідні: i1 = 2, p = 6

1. Вхідні: N = 5, A[5] = {-1, 1, -2, 3, 1}

Процес: -1, 1, -2, 3, 1

Вихідні: i1 = 0. Такого елемента не існує (Condition isn’t satisfied!!!)

1. Вхідні: N = 10, A[10] = {1, 5, 7, -5, 10, 11, 8, 12, 8, 7}

Процес: 1, 5, 7, -5, 10, 11, 8, 12, 8, 7

Вихідні: i1 = 3, p = 8

# Завдання 3

## Опис вхідних та вихідних даних

Вхідні дані:

i - індекс

x, y – координати точки, яку перевіряєм

LST [200] – список з максимальною к-тю елементів 200, сюди вводимо к-ть точок, які будем розглядати у проекті

N – к-ть точок, які будуть перевірятись у даному проекті

xa, ya – координати точки А, однієї вершини трикутника

xb, yb – координати точки В, другої вершини трикутника

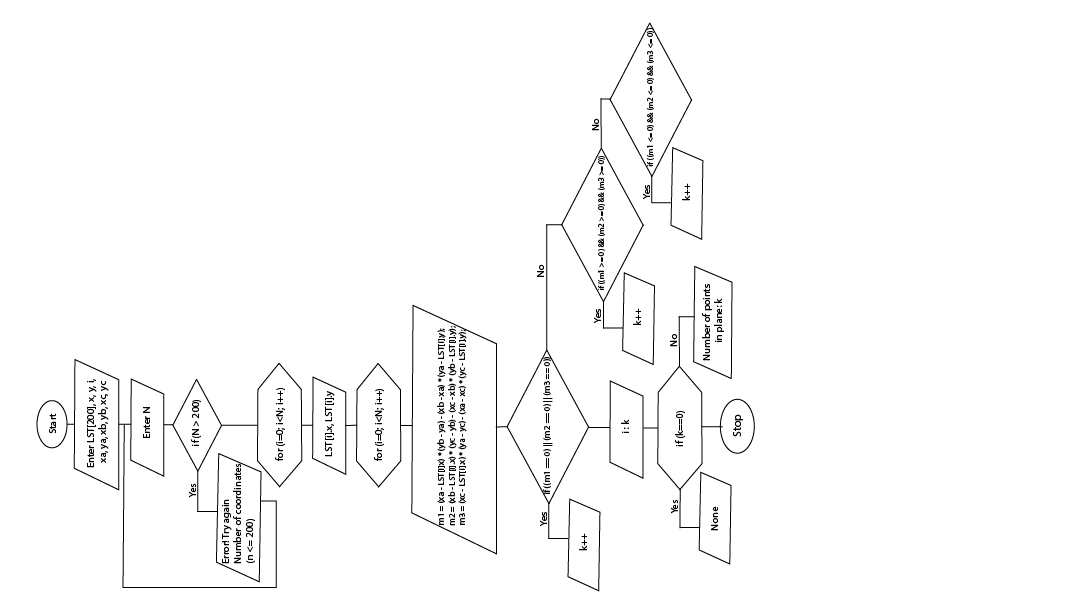
xc, yc – координати точки С, третьої вершини трикутника

m1, m2, m3 – данні, які вираховуються за формулою, для виявлення чи знаходиться точка у площині трикутника

Вихідні дані:

k – к-ть точок, які належать трикутнику

## Схема алгоритму розв’язання завдання



## Лістинг програми

typedef struct {

float x, y;

} Coord;

Coord LST[200];

int main()

{

int n = 200, i, k = 0;

float xa, ya, xb, yb, xc, yc, m1, m2, m3;

cout << "Input A(xa, ya): ";

cin >> xa >> ya;

cout << "Input B(xb, yb): ";

cin >> xb >> yb;

cout << "Input C(xc, yc): ";

cin >> xc >> yc;

int N;

Num: {

cout << "Number of coordinates (n <= 200) ";

cin >> N; }

if (N > 200) {

cout << "Error" << " " << "Try again \n";

cout << "Number of coordinates(n <= 200) ";

goto Num;

}

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

cout << "Input(x" << i << "; y" << i << ") : ";

cin >> LST[i].x >> LST[i].y;

}

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

m1 = (xa - LST[i].x) \* (yb - ya) - (xb - xa) \* (ya - LST[i].y);

m2 = (xb - LST[i].x) \* (yc - yb) - (xc - xb) \* (yb - LST[i].y);

m3 = (xc - LST[i].x) \* (ya - yc) - (xa - xc) \* (yc - LST[i].y);

if ((m1 == 0) || (m2 == 0) || (m3 == 0)) {

k++;

}

else if ((m1 >= 0) && (m2 >= 0) && (m3 >= 0)) {

k++;

}

else if ((m1 <= 0) && (m2 <= 0) && (m3 <= 0)) {

k++;

}

cout << i << ": " << k << endl;

}

if (k == 0) cout << "None \n";

else {

cout << "Number of points in plane: " << k;

}

return 0;

}

## Розробка тестів та результати тестування

1. Вхідні дані: A (15.41; 7.3), B (-12.99; 4.71), C (5.28; 3.49), N = 3, D (6.31; 4.87), E (-7.47; 9.84), F (5.71; 20.3)

Вихідні: k = 1

1. Вхідні дані: A (4.5; -8.36), B (15.97; 6.78), C (19.27; -7.74), N = 3,

D (-8.47; 10.84), E (1.57; 3.48), F (7.07; 45.07)

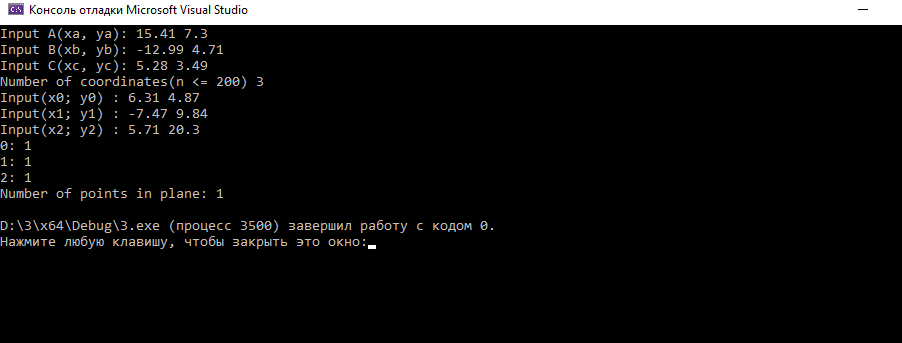
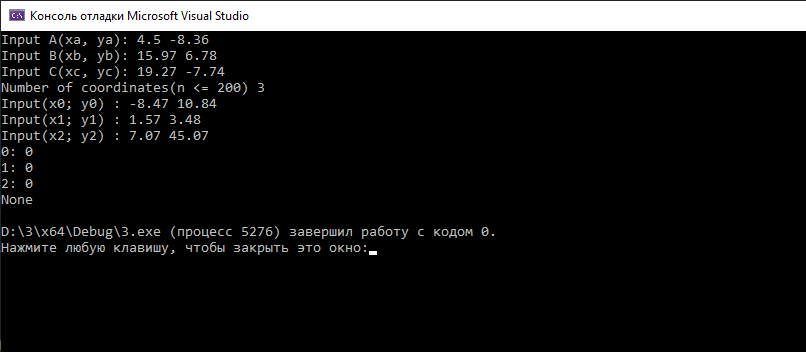
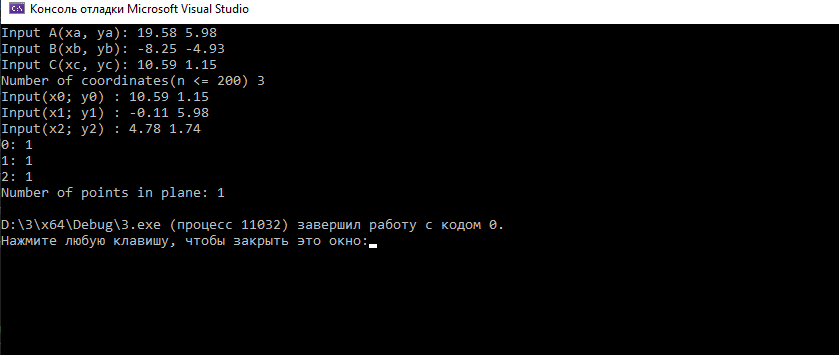
Вихідні: k = 0

1. Вхідні дані: A (19.58; 5.98), B (-8.25; -4.93), C (10.59; 1.15), N = 3,

D (10.59; 1.15), E (-0.11; 5.98), F (4.78; 1.74)

Вихідні: k = 1

## Лістинг (скріншоти) вхідних та вихідних даних і результатів розв’язання завдання

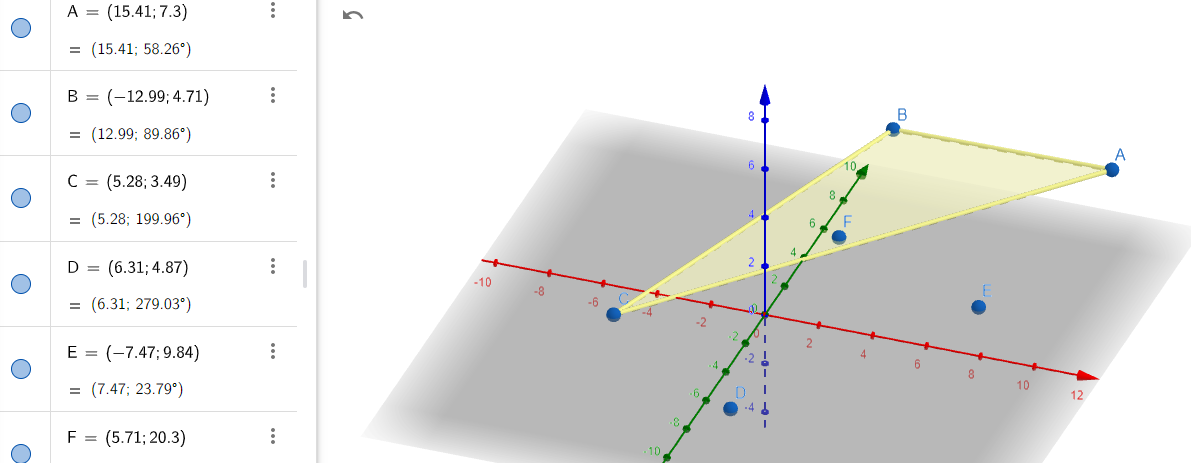
1. 
2. 
3. 

## Аналіз отриманих результатів

1. Вхідні: A (15.41; 7.3), B (-12.99; 4.71), C (5.28; 3.49), N = 3, D (6.31; 4.87), E (-7.47; 9.84), F (5.71; 20.3)

Процес: F (5.71; 20.3) – лежить у площині трикутника

Вихідні: k = 1

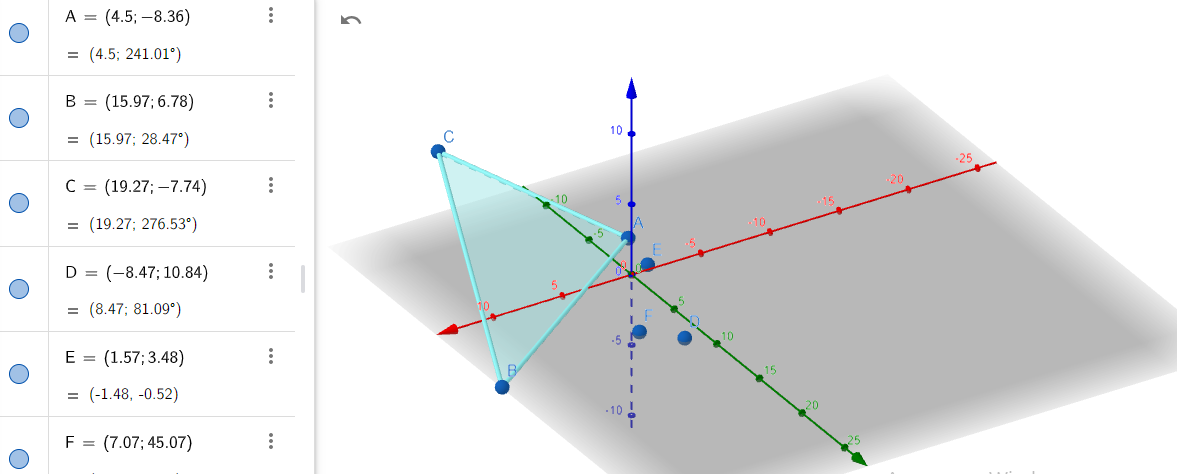


1. Вхідні: A (4.5; -8.36), B (15.97; 6.78), C (19.27; -7.74), N = 3,

D (-8.47; 10.84), E (1.57; 3.48), F (7.07; 45.07)

Процес: жодна з точок не належить площині трикутника

Вихідні: k = 0

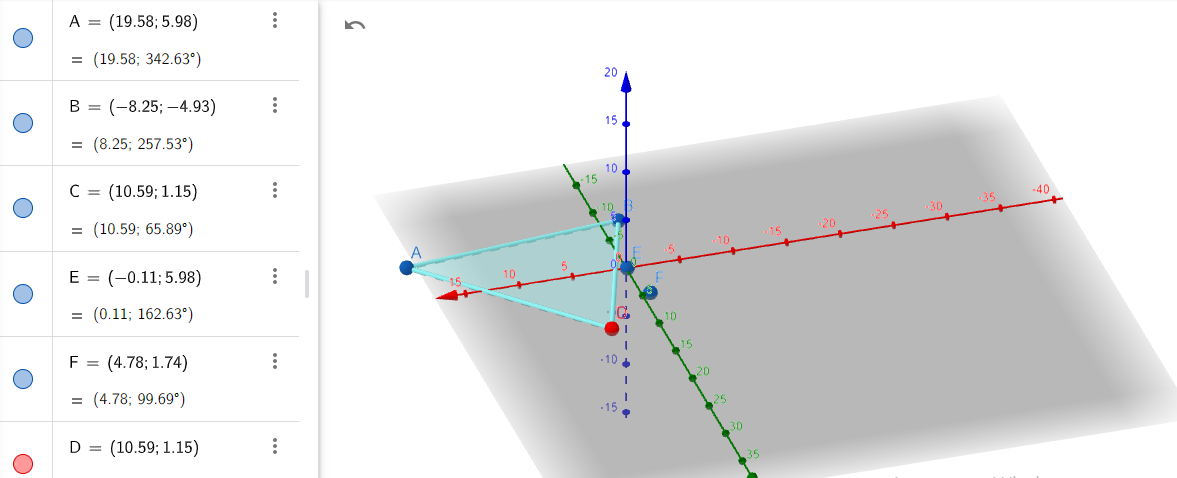


1. Вхідні: A (19.58; 5.98), B (-8.25; -4.93), C (10.59; 1.15), N = 3, D (10.59; 1.15),

E (-0.11; 5.98), F (4.78; 1.74)

Процес: D (10.59; 1.15) – лежить на точці C

Вихідні: k = 1



# Висновок

Вивчила засоби мови С++ роботи з динамічними масивами, операціями

простого консольного введення-виведення та файлового консольного введення-

виведення.

Зміст

[Завдання до лабораторної роботи 2](#_Toc128311772)

[Завдання 1 3](#_Toc128311773)

[Опис вхідних та вихідних даних 3](#_Toc128311774)

[Схема алгоритму розв’язання завдання 3](#_Toc128311775)

[Лістинг програми 4](#_Toc128311776)

[Розробка тестів та результати тестування 4](#_Toc128311777)

[Лістинг (скріншоти) вхідних та вихідних даних і результатів розв’язання завдання 5](#_Toc128311778)

[Аналіз отриманих результатів 6](#_Toc128311779)

[Завдання 2 6](#_Toc128311780)

[Опис вхідних та вихідних даних 6](#_Toc128311781)

[Схема алгоритму розв’язання завдання 7](#_Toc128311782)

[Лістинг програми 7](#_Toc128311783)

[Розробка тестів та результати тестування 9](#_Toc128311784)

[Лістинг (скріншоти) вхідних та вихідних даних і результатів розв’язання завдання 9](#_Toc128311785)

[Аналіз отриманих результатів 10](#_Toc128311786)

[Завдання 3 10](#_Toc128311787)

[Опис вхідних та вихідних даних 10](#_Toc128311788)

[Схема алгоритму розв’язання завдання 12](#_Toc128311789)

[Лістинг програми 12](#_Toc128311790)

[Розробка тестів та результати тестування 14](#_Toc128311791)

[Лістинг (скріншоти) вхідних та вихідних даних і результатів розв’язання завдання 14](#_Toc128311792)

[Аналіз отриманих результатів 15](#_Toc128311793)

[Висновок 16](#_Toc128311794)