МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КОЛЕДЖ ЧЕРНІВЕЦЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

**ЗВІТ**

про навчальну практику

студентки II курсу, спеціальності

«Комп`ютерні науки»

204 групи

*Гончарук Анастасія Ігорівна*

Період практики з “20” 2020 р. по “7” 2020 р.

Керівник практики від коледжу:

викл. Ковдриш В.В. \_\_\_\_\_\_\_\_\_\_

/підпис/

Оцінка:\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

/підпис/

ЧЕРНІВЦІ, 2020

ТАБЛИЦЯ ОЦІНЮВАННЯ

|  |  |  |
| --- | --- | --- |
| Завдання | Оцінка за завдання | Підпис керівника практики |
| Завдання 1. |  |  |
| Завдання 2. |  |  |
| Завдання 3. |  |  |
| Завдання 4. |  |  |
| Завдання 5. |  |  |
| Завдання 6. |  |  |
| Завдання 7. |  |  |
| Завдання 8. |  |  |
| Завдання 9. |  |  |
| Завдання 10. |  |  |
| Завдання 11. |  |  |
| Завдання 12. |  |  |
| Завдання 13. |  |  |
| Завдання 14. |  |  |
| Оформлення звіту | |  |
| Захист звіту | |  |
| Підсумкова оцінка за навчальну практику | |  |

Зміст

[Завдання 1 5](#__RefHeading___Toc2200_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 6](#__RefHeading___Toc2302_2096317517)

[Опис завдання 7](#__RefHeading___Toc3512_2096317517)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 7](#__RefHeading___Toc3404_2096317517)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 7](#__RefHeading___Toc3409_2096317517)

[Висновок 7](#__RefHeading___Toc3716_2096317517)

[Додаток(а) 7](#__RefHeading___Toc3718_2096317517)

[Додаток(б) 8](#__RefHeading___Toc3720_2096317517)

[Завдання 2 9](#__RefHeading___Toc3737_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 9](#__RefHeading___Toc2302_20963175171)

[Опис завдання 11](#__RefHeading___Toc3512_20963175171)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 14](#__RefHeading___Toc3404_20963175171)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 15](#__RefHeading___Toc3409_20963175171)

[Висновок 16](#__RefHeading___Toc3716_20963175171)

[Додаток(а) 16](#__RefHeading___Toc3718_20963175171)

[Додаток(б) 16](#__RefHeading___Toc3720_20963175171)

[Завдання 3 17](#__RefHeading___Toc3760_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 17](#__RefHeading___Toc2302_20963175172)

[Опис завдання 17](#__RefHeading___Toc3512_20963175172)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 18](#__RefHeading___Toc3404_20963175172)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 18](#__RefHeading___Toc3409_20963175172)

[Висновок 22](#__RefHeading___Toc3716_20963175172)

[Додаток(а) 26](#__RefHeading___Toc3718_20963175172)

[Додаток(б) 28](#__RefHeading___Toc3720_20963175172)

[Завдання 4 29](#__RefHeading___Toc3762_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 30](#__RefHeading___Toc2302_20963175173)

[Опис завдання 30](#__RefHeading___Toc3512_20963175173)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 30](#__RefHeading___Toc3404_20963175173)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 31](#__RefHeading___Toc3409_20963175173)

[Висновок 31](#__RefHeading___Toc3716_20963175173)

[Додаток(а) 32](#__RefHeading___Toc3718_20963175173)

[Додаток(б) 34](#__RefHeading___Toc3720_20963175173)

[Завдання 5 35](#__RefHeading___Toc3764_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 36](#__RefHeading___Toc2302_20963175174)

[Опис завдання 36](#__RefHeading___Toc3512_20963175174)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 36](#__RefHeading___Toc3404_20963175174)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 36](#__RefHeading___Toc3409_20963175174)

[Висновок 39](#__RefHeading___Toc3716_20963175174)

[Додаток(а) 38](#__RefHeading___Toc3718_20963175174)

[Додаток(б) 39](#__RefHeading___Toc3720_20963175174)

[Завдання 6 43](#__RefHeading___Toc2200_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 44](#__RefHeading___Toc2302_2096317517)

[Опис завдання 45](#__RefHeading___Toc3512_2096317517)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 45](#__RefHeading___Toc3404_2096317517)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 46](#__RefHeading___Toc3409_2096317517)

[Висновок 47](#__RefHeading___Toc3716_2096317517)

[Додаток(а) 48](#__RefHeading___Toc3718_2096317517)

[Додаток(б) 49](#__RefHeading___Toc3720_2096317517)

[Завдання 7 50](#__RefHeading___Toc3737_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 5](#__RefHeading___Toc2302_20963175171)1

[Опис завдання 51](#__RefHeading___Toc3512_20963175171)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 51](#__RefHeading___Toc3404_20963175171)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 51](#__RefHeading___Toc3409_20963175171)

[Висновок 52](#__RefHeading___Toc3716_20963175171)

[Додаток(а) 53](#__RefHeading___Toc3718_20963175171)

[Додаток(б) 54](#__RefHeading___Toc3720_20963175171)

[Завдання 8 56](#__RefHeading___Toc3760_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 57](#__RefHeading___Toc2302_20963175172)

[Опис завдання 57](#__RefHeading___Toc3512_20963175172)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 57](#__RefHeading___Toc3404_20963175172)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 58](#__RefHeading___Toc3409_20963175172)

[Висновок 58](#__RefHeading___Toc3716_20963175172)

[Додаток(а) 59](#__RefHeading___Toc3718_20963175172)

[Додаток(б) 59](#__RefHeading___Toc3720_20963175172)

[Завдання 9 60](#__RefHeading___Toc3762_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 61](#__RefHeading___Toc2302_20963175173)

[Опис завдання 61](#__RefHeading___Toc3512_20963175173)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 62](#__RefHeading___Toc3404_20963175173)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 62](#__RefHeading___Toc3409_20963175173)

[Висновок 62](#__RefHeading___Toc3716_20963175173)

[Додаток(а) 63](#__RefHeading___Toc3718_20963175173)

[Додаток(б) 63](#__RefHeading___Toc3720_20963175173)

[Завдання 10 64](#__RefHeading___Toc3764_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 65](#__RefHeading___Toc2302_20963175174)

[Опис завдання 65](#__RefHeading___Toc3512_20963175174)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 65](#__RefHeading___Toc3404_20963175174)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 66](#__RefHeading___Toc3409_20963175174)

[Висновок 67](#__RefHeading___Toc3716_20963175174)

[Додаток(а) 67](#__RefHeading___Toc3718_20963175174)

[Додаток(б) 68](#__RefHeading___Toc3720_20963175174)

[Завдання 11 69](#__RefHeading___Toc2200_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 68](#__RefHeading___Toc2302_2096317517)

[Опис завдання 68](#__RefHeading___Toc3512_2096317517)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 68](#__RefHeading___Toc3404_2096317517)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 69](#__RefHeading___Toc3409_2096317517)

[Висновок 70](#__RefHeading___Toc3716_2096317517)

[Додаток(а) 71](#__RefHeading___Toc3718_2096317517)

[Додаток(б) 72](#__RefHeading___Toc3720_2096317517)

[Завдання 12 74](#__RefHeading___Toc3737_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 75](#__RefHeading___Toc2302_20963175171)

[Опис завдання 76](#__RefHeading___Toc3512_20963175171)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 76](#__RefHeading___Toc3404_20963175171)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 76](#__RefHeading___Toc3409_20963175171)

[Висновок 77](#__RefHeading___Toc3716_20963175171)

[Додаток(а) 77](#__RefHeading___Toc3718_20963175171)

[Додаток(б) 78](#__RefHeading___Toc3720_20963175171)

[Завдання 13 79](#__RefHeading___Toc3760_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 80](#__RefHeading___Toc2302_20963175172)

[Опис завдання 80](#__RefHeading___Toc3512_20963175172)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 81](#__RefHeading___Toc3404_20963175172)

[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 82](#__RefHeading___Toc3409_20963175172)

[Висновок 82](#__RefHeading___Toc3716_20963175172)

[Додаток(а) 83](#__RefHeading___Toc3718_20963175172)

[Додаток(б) 84](#__RefHeading___Toc3720_20963175172)

[Завдання 14 85](#__RefHeading___Toc3762_2096317517)

[ПОСТАНОВКА ЗАДАЧІ 86](#__RefHeading___Toc2302_20963175173)

[Опис завдання 86](#__RefHeading___Toc3512_20963175173)

[ТЕОРЕТИЧНІ ВІДОМОСТІ 87](#__RefHeading___Toc3404_20963175173)

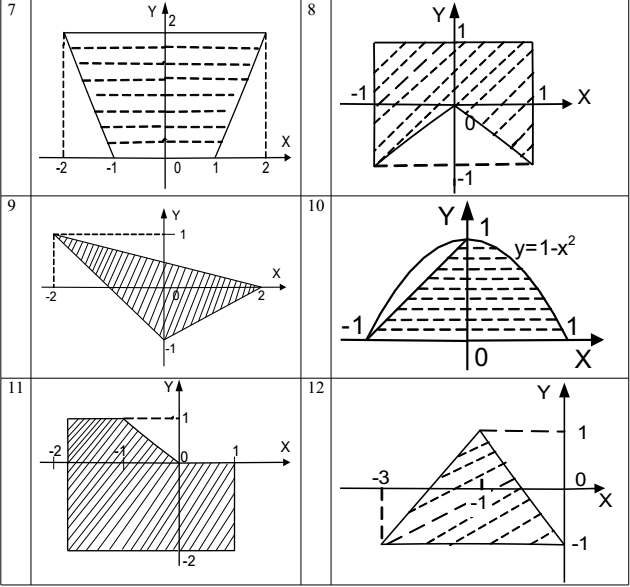
[СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ 88](#__RefHeading___Toc3409_20963175173)

[Висновок 89](#__RefHeading___Toc3716_20963175173)

[Додаток(а) 90](#__RefHeading___Toc3718_20963175173)

[Додаток(б) 92](#__RefHeading___Toc3720_20963175173)

# Завдання 1

1. **Лабораторна робота №1 ВИКОРИСТАННЯ ЛОГІЧНОГО ОПЕРАТОРА IF ТА ОПЕРАТОРА ВИБОРУ SWITCH .перевірка умови**
2. **Завдання:** Створити алгоритм та написати програму для розвязку наступної задачі: перевірку попадання точки ( яка буде вводитись вручну із клавіатури) в заштриховану область.
3. 

## ТЕОРЕТИЧНІ ВІДОМОСТІ

## У деяких джерелах говориться, що оператор вибору if else - самостійний оператор. Але це не так, if else - це всього лише форма запису оператора вибору if. Оператор if else дозволяє визначити програмісту дію, коли умова істинна і альтернативна дія, коли умова помилкова. Тоді як if дозволяв визначити дію при істинному умови.

## Синтаксис запису оператора вибору if else:

## if (/ \* перевіряється умова \* /)

## {

## / \* Тіло оператора вибору 1 \* /;

## } else

## {

## / \* Тіло оператора вибору 2 \* /;

## }

Читається так: «Якщо перевіряється умова істинно, то виконується тіло оператора вибору 1, інакше (тобто перевіряється умова помилкова) виконується тіло оператора вибору 2«. Зверніть увагу на те, як записаний оператор if else. Слово else спеціально зрушено раво для того щоб програмний код був зрозумілий і його було зручночитати.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ

## 

## Висновок

Під час виконання данного завдання отримала навички з використання логічних операторів.

## 

## Додаток(а)

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. double x, y;
6. cout << "Input x and y: \n";
7. cin >> x >> y;
8. if (y <= 1 && y >= 0) {
9. if (x >= 0 && x <= 1) {
10. cout << "true";
11. }
12. if (y > 0 && y <= x) {
13. cout << "true";
14. }
15. }
16. else {
17. cout << "false";
18. }
19. return 0;
20. }

## Додаток(б)



# Завдання 2

1. **Лабораторна робота №2. Одновимірні масиви**
2. **Завдання.** **Розробити блок-схеми і програми для заданих задач**.
3. **I. Виконання обчислень.**

Обчислити та вивести середн∙ геометричне елементiв масиву P(11), якi не

кратнi 3.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

*Одновимірна масив*

*Стандартний вид оголошення одновимірного масиву наступний:*

*тип ім'я\_змінної [розмір];*

*В С масиви повинні визначатися однозначно, щоб компілятор міг виділити для них місце в пам'яті. Тут тип оголошує базовий тип масиву і є типом кожного елемента масиву. Параметр розмір визначає, скільки елементів містить масив. В одновимірному масиві повний розмір масиву в байтах обчислюється таким чином:*

*загальне число байт = sizeof (базовий тип) \* число елементів*

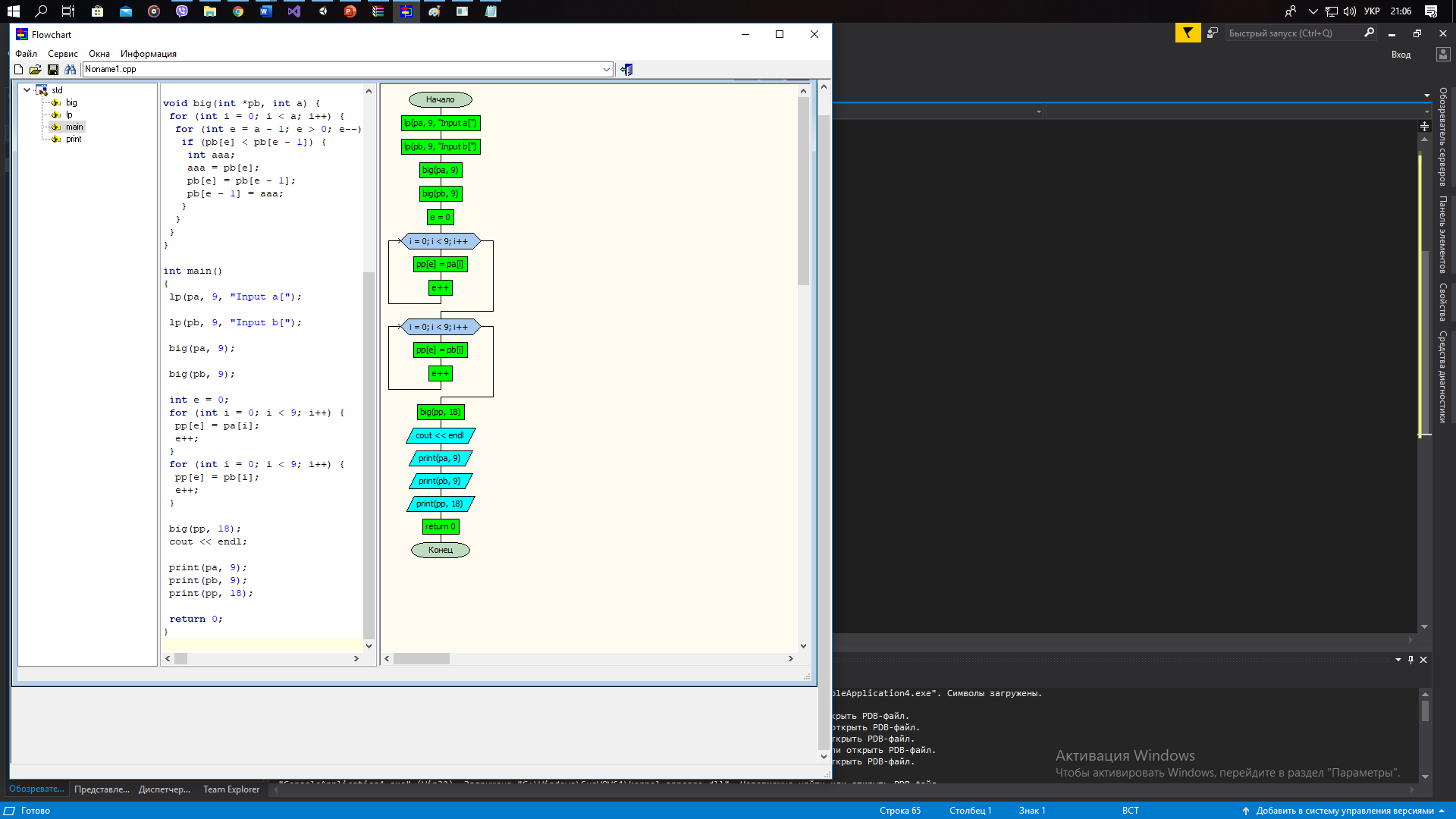
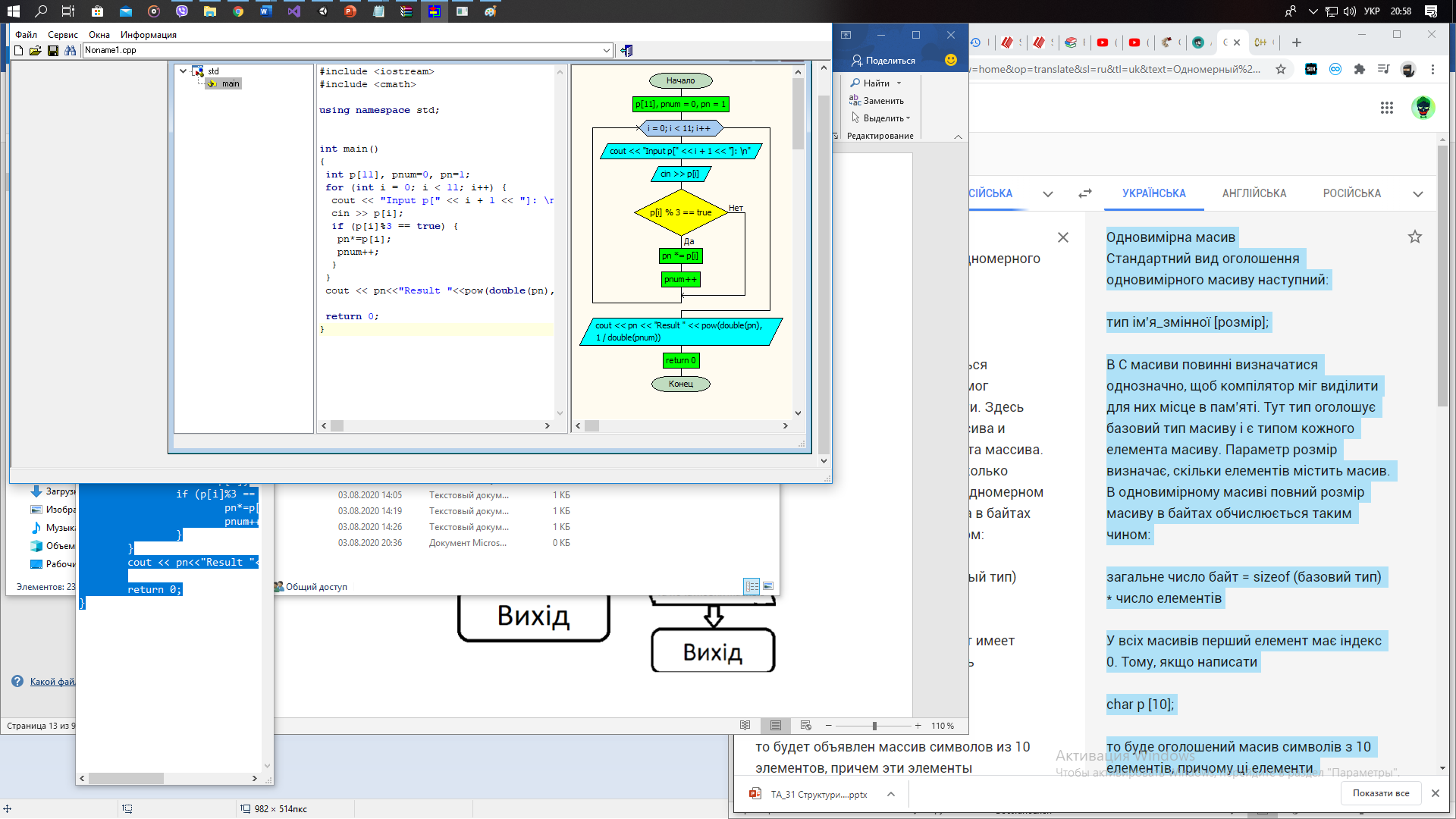
*У всіх масивів перший елемент має індекс 0. Тому, якщо написати*

*char р [10];*

*то буде оголошений масив символів з 10 елементів, причому ці елементи адресуються індексом від 0 до 9.*

*В С ++ відсутня перевірка кордонів масивів. Можна вийти за один кінець масиву і записати значення в будь-яку змінну, що не відноситься до масиву, або навіть в код програми. Робота з надання перевірки кордонів покладається на програміста. Наприклад, слід переконатися, що масив символів, куди здійснюється введення, має достатню довжину для прийняття найдовшою послідовності символів.*

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ



## Висновок

1. Під час виконання данного завданнь отримала навички з використання одновимірних масисів

## Додаток(а)

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

int p[11], pnum = 0, pn = 1;

for (int i = 0; i < 11; i++) {

cout << "Input p[" << i + 1 << "]: \n";

cin >> p[i];

if (p[i] % 3 == true) {

pn \*= p[i];

pnum++;

}

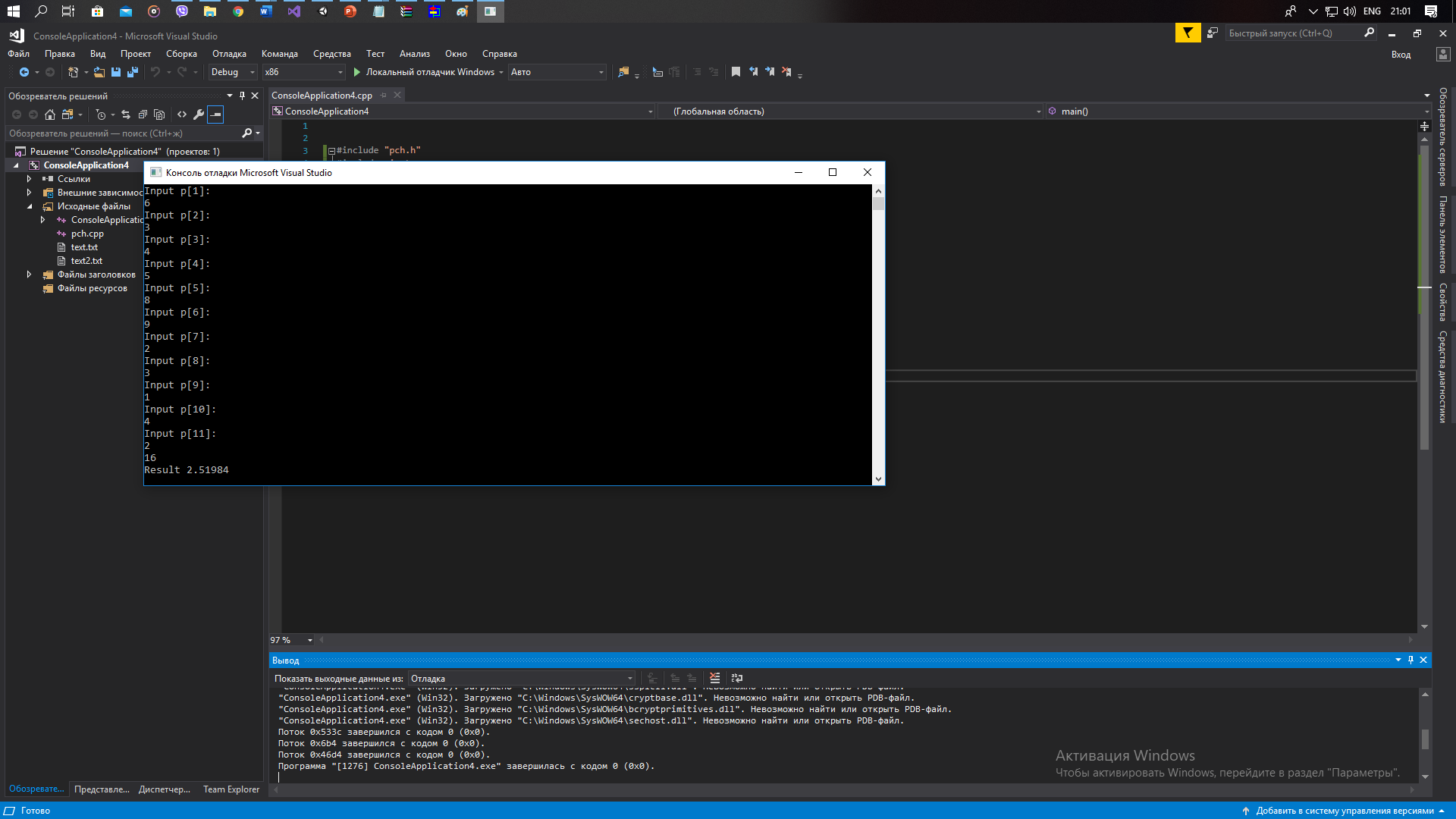
}

cout << pn <<endl<< "Result " << pow(double(pn), 1 / double(pnum));

return 0;

## }

## Додаток(б)



## Додаток(в)

**II. Упорядкування елементів масиву.**

1. Об’єднати два упорядкованих за зростанням масиви F(9) та К(9) в один масив С(18), також упорядкований за зростанням. Вивести на екран всі масиви.

#include <iostream>

#include <string>

using namespace std;

int pa[9], pb[9], pp[18];

void lp(int \*pb, int a, string p) {

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

cout << p << i + 1 << "]: \n";

cin >> pb[i];

}

}

void print(int \*pb, int a) {

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

cout << pb[i] << " ";

}

cout << endl;

}

void big(int \*pb, int a) {

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

for (int e = a - 1; e > 0; e--) {

if (pb[e] < pb[e - 1]) {

int aaa;

aaa = pb[e];

pb[e] = pb[e - 1];

pb[e - 1] = aaa;

}

}

}

}

int main()

{

lp(pa, 9, "Input a[");

lp(pb, 9, "Input b[");

big(pa, 9);

big(pb, 9);

int e = 0;

for (int i = 0; i < 9; i++) {

pp[e] = pa[i];

e++;

}

for (int i = 0; i < 9; i++) {

pp[e] = pb[i];

e++;

}

big(pp, 18);

cout << endl;

print(pa, 9);

print(pb, 9);

print(pp, 18);

return 0;

}

Додаток(г)



**Завдання 3**

1. **Лабораторна робота №3. Двовимірні масиви. робота із матрицями**
2. **Завдання.**  Розробити алгоритми та програми для заданих задач.
3. **I. Обробка елементів матриці.**

В заданiй матрицi Z(7,3) визначити iндекси ненульових елементiв. Вивести вихiдну матрицю та iндекси ненульових елементiв.

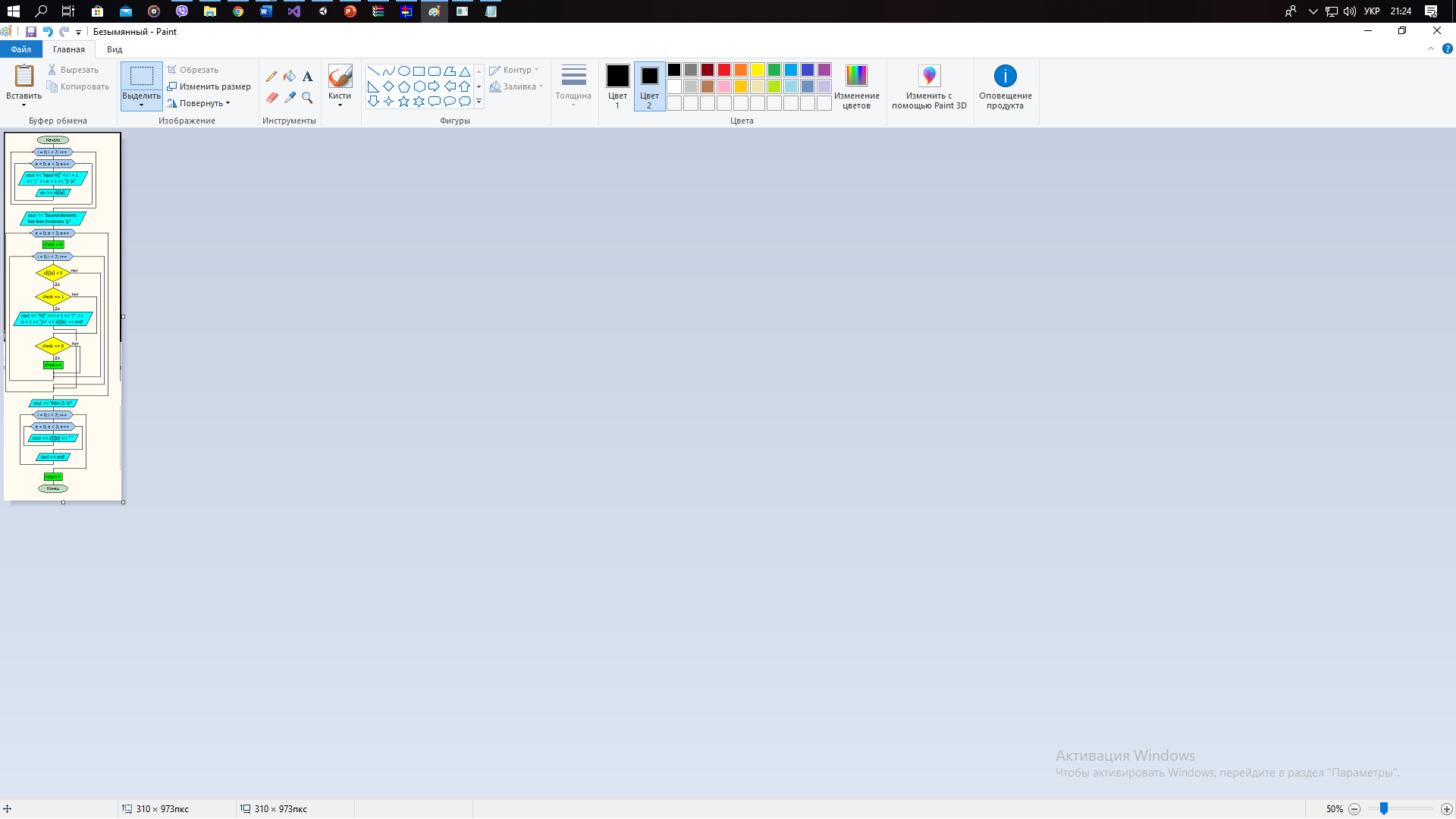
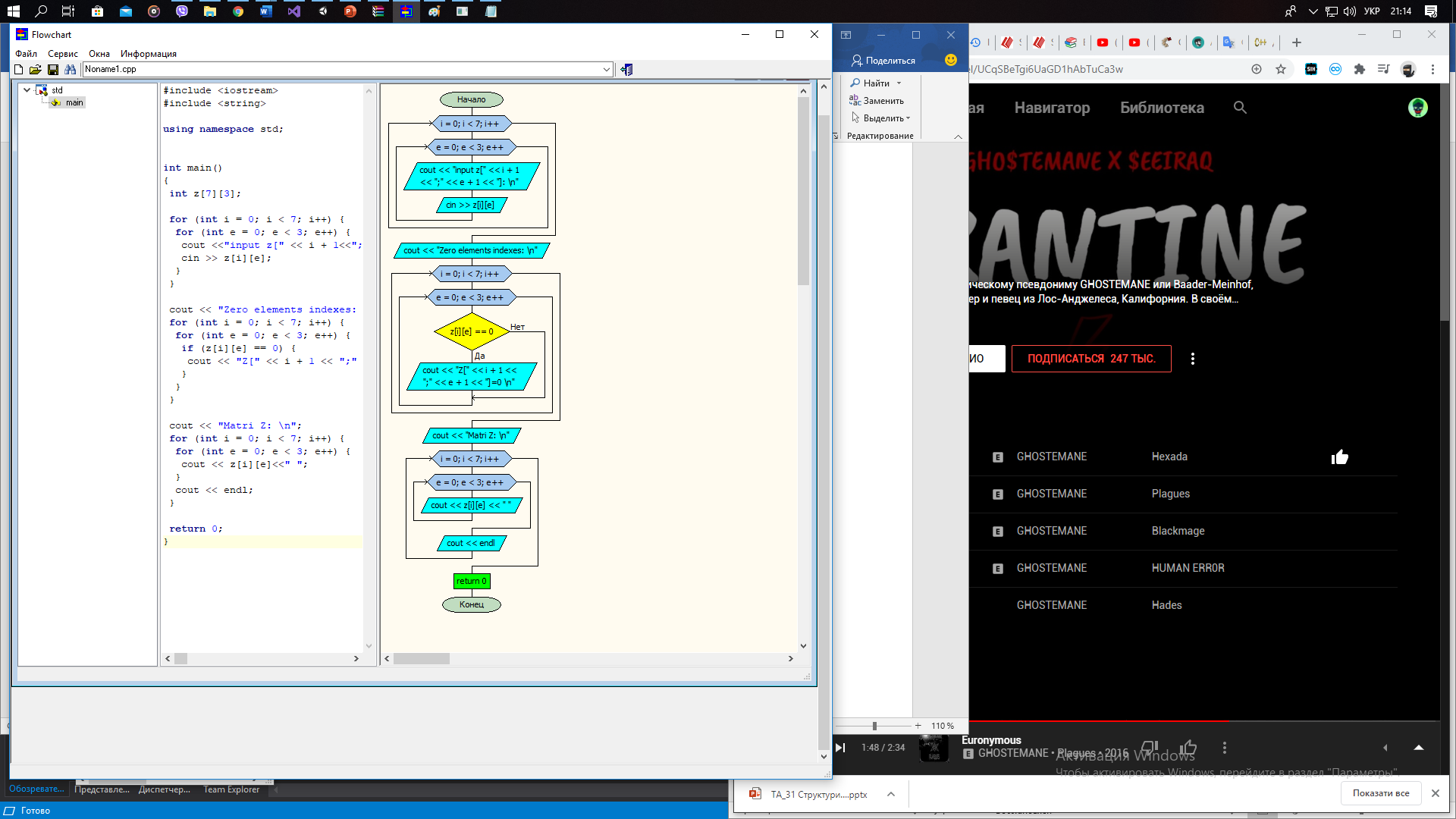
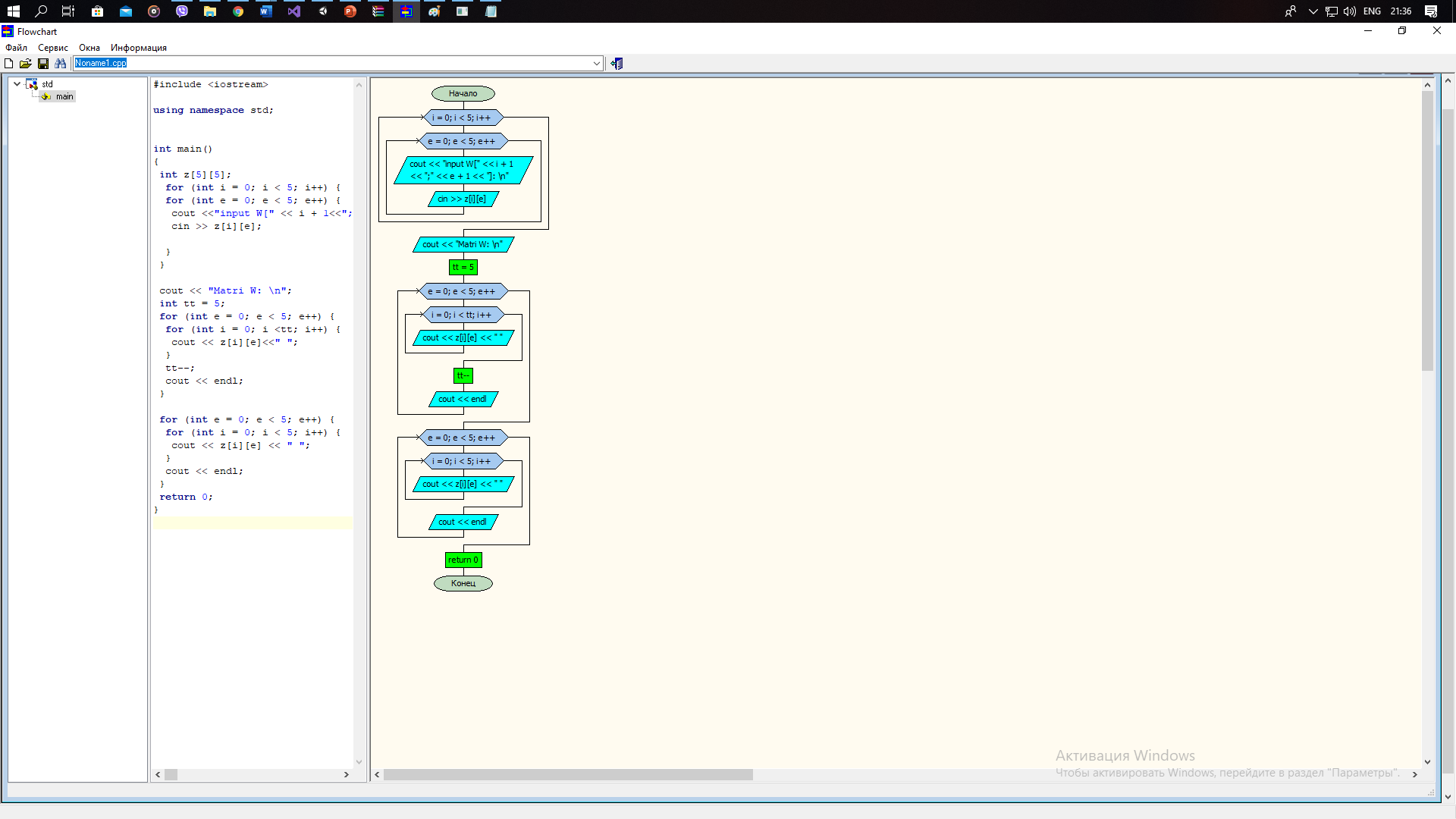
## ТЕОРЕТИЧНІ ВІДОМОСТІ

Найпростішим видом багатовимірного масиву є двовимірний масив. Двовимірний масив - це масив одновимірних масивів. Двовимірний масив оголошується наступним чином:

Двовимірні масиви зберігаються у вигляді матриці, де перший індекс відповідає за рядок, а другий -за стовпець. Це означає, що правий індекс змінюється швидше лівого, якщо рухатися по масиву в порядку розташування елементів в пам'яті. На рис. показано графічне представлення двовимірного масиву в пам'яті. Лівий індекс можна розглядати як покажчик на рядок.

Коли двовимірний масив використовується як аргумент функції, передається покажчик на перший елемент. Функція, яка отримує двовимірний масив, повинна, як мінімум, визначати розмір першого виміру, оскільки компілятору необхідно знати довжину кожного рядка для коректної індексації масиву.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ

1.  

## Висновок

1. Під час виконання данного завданнь отримала навички з використання двовимірних масисів

## Додаток(а)

1. #include <iostream>
2. #include <string>
3. using namespace std;
4. int main()
5. {
6. int z[7][3];
7. for (int i = 0; i < 7; i++) {
8. for (int e = 0; e < 3; e++) {
9. cout << "input z[" << i + 1 << ";" << e + 1 << "]: \n";
10. cin >> z[i][e];
11. }
12. }
13. cout << "Zero elements indexes: \n";
14. for (int i = 0; i < 7; i++) {
15. for (int e = 0; e < 3; e++) {
16. if (z[i][e] == 0) {
17. cout << "Z[" << i + 1 << ";" << e + 1 << "]=0 \n";
18. }
19. }
20. }
21. cout << "Matri Z: \n";
22. for (int i = 0; i < 7; i++) {
23. for (int e = 0; e < 3; e++) {
24. cout << z[i][e] << " ";
25. }
26. cout << endl;
27. }
28. return 0;
29. }

## Додаток(б)

## 

**II. Обробка рядків і стовпчиків матриці.**

В заданiй матрицi W(7,3) визначити в кожному стовпчику другий вiд'ємний елемент. Вивести вихiдну матрицю, знайденi елементи та їх iндекси.

## Додаток(в)

#include <iostream>

using namespace std;

int main()

{

int z[7][3];

for (int i = 0; i < 7; i++) {

for (int e = 0; e < 3; e++) {

cout << "input W[" << i + 1 << ";" << e + 1 << "]: \n";

cin >> z[i][e];

}

}

cout << "Second elements less than 0 indexes: \n";

for (int e = 0; e < 3; e++) {

int check = 0;

for (int i = 0; i < 7; i++) {

if (z[i][e] < 0) {

if (check == 1) {

cout << "W[" << i + 1 << ";" << e + 1 << "]=" << z[i][e] << endl;

break;

}

if (check == 0) {

check++;

}

}

}

}

cout << "Matri Z: \n";

for (int i = 0; i < 7; i++) {

for (int e = 0; e < 3; e++) {

cout << z[i][e] << " ";

}

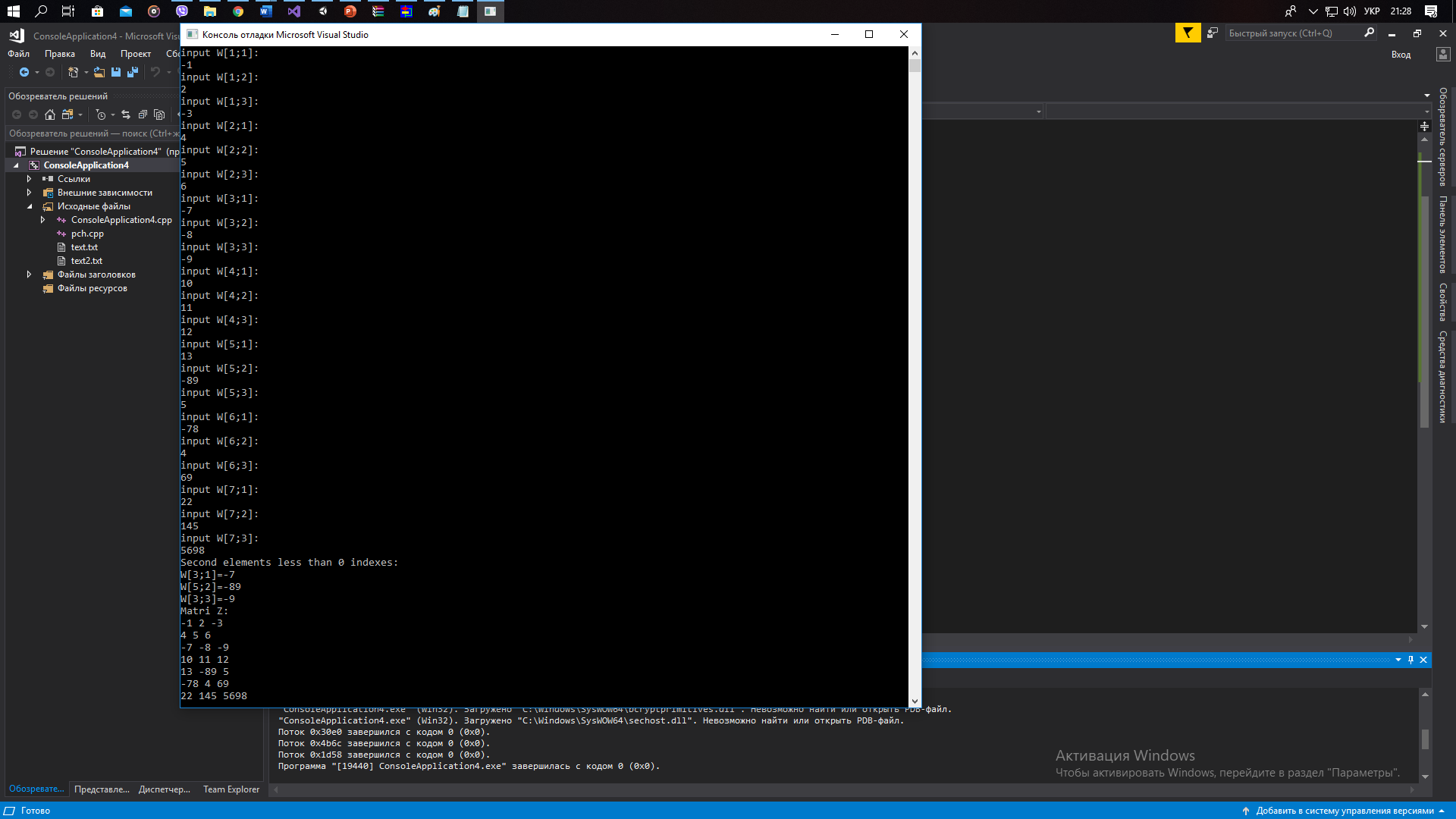
cout << endl;

}

return 0;

}

## Додаток(г)



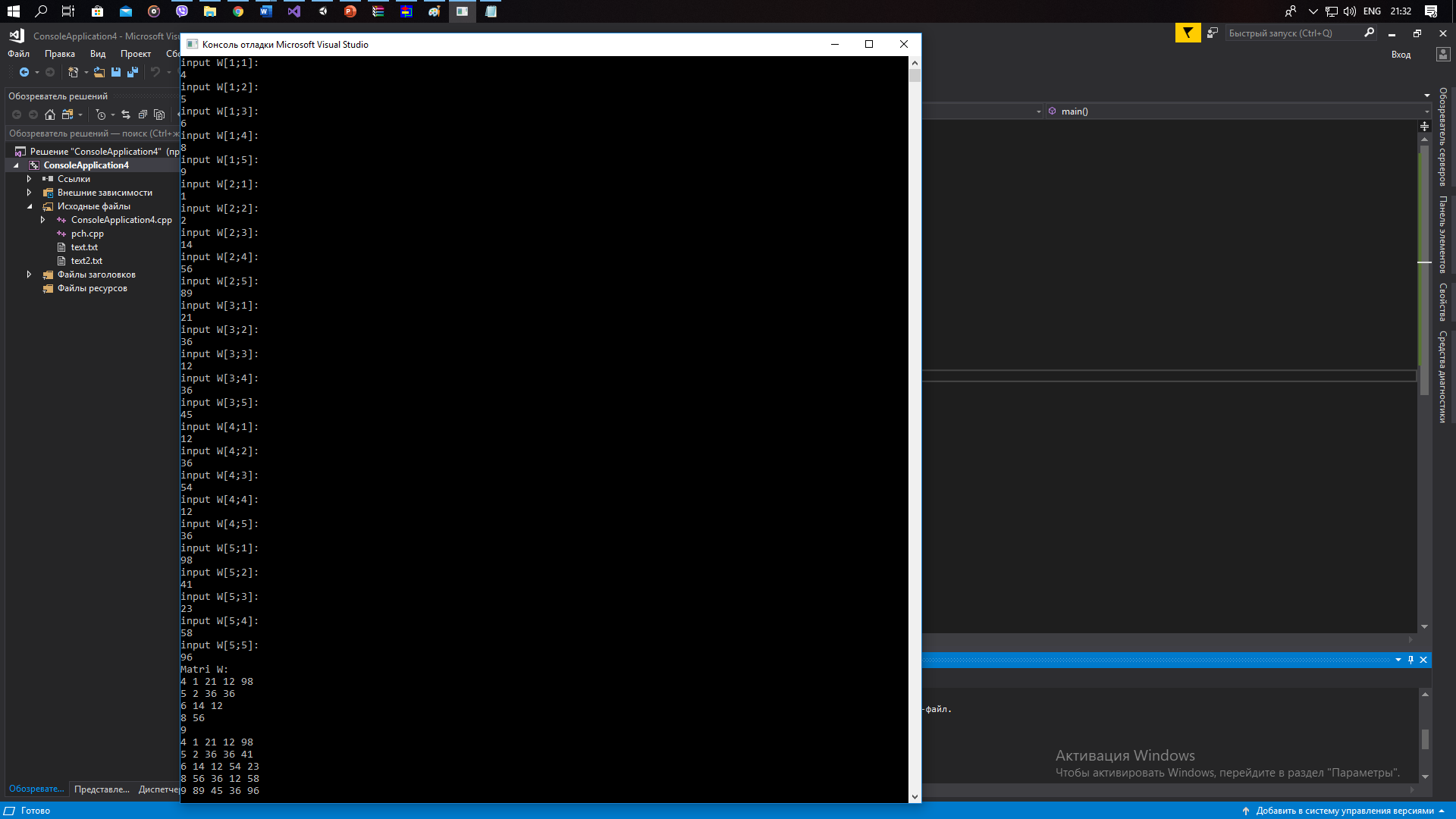
**Обробка частини матриці**

Обчислити суму та кількість додатних елементів, що знаходяться над головною діагоналлю в заданій матриці W(5,5). Вивести вихiдну матрицю, суму та кількість додатних елементів.

## Додаток(д)

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int z[5][5];
6. for (int i = 0; i < 5; i++) {
7. for (int e = 0; e < 5; e++) {
8. cout <<"input W[" << i + 1<<";"<<e+1 << "]: \n";
9. cin >> z[i][e];
11. }
12. }
13. cout << "Matri W: \n";
14. int tt = 5;
15. for (int e = 0; e < 5; e++) {
16. for (int i = 0; i <tt; i++) {
17. cout << z[i][e]<<" ";
18. }
19. tt--;
20. cout << endl;
21. }
22. for (int e = 0; e < 5; e++) {
23. for (int i = 0; i < 5; i++) {
24. cout << z[i][e] << " ";
25. }
26. cout << endl;
27. }
28. return 0;
29. }

## Додаток(е)



**Завдання 4**

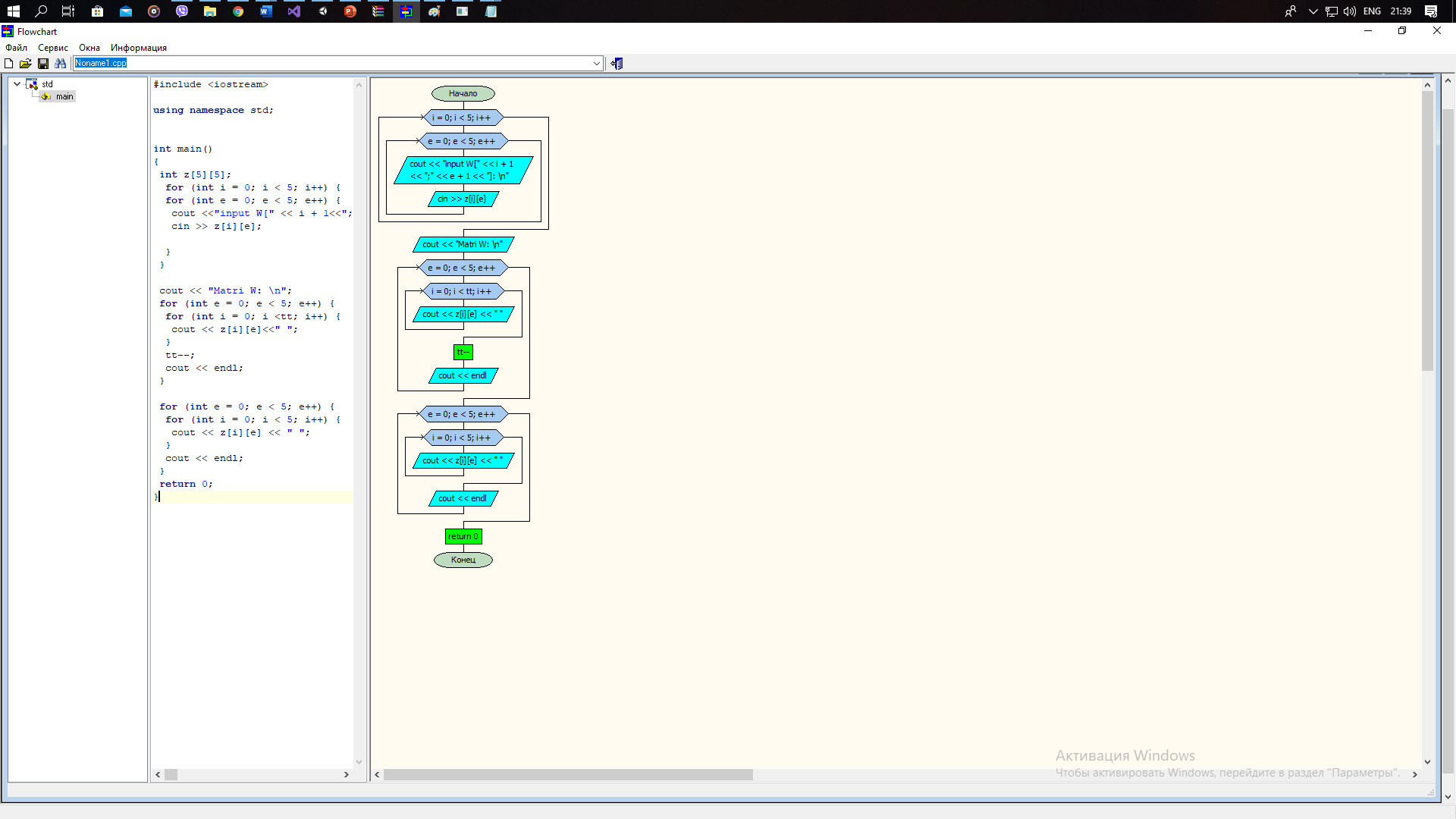
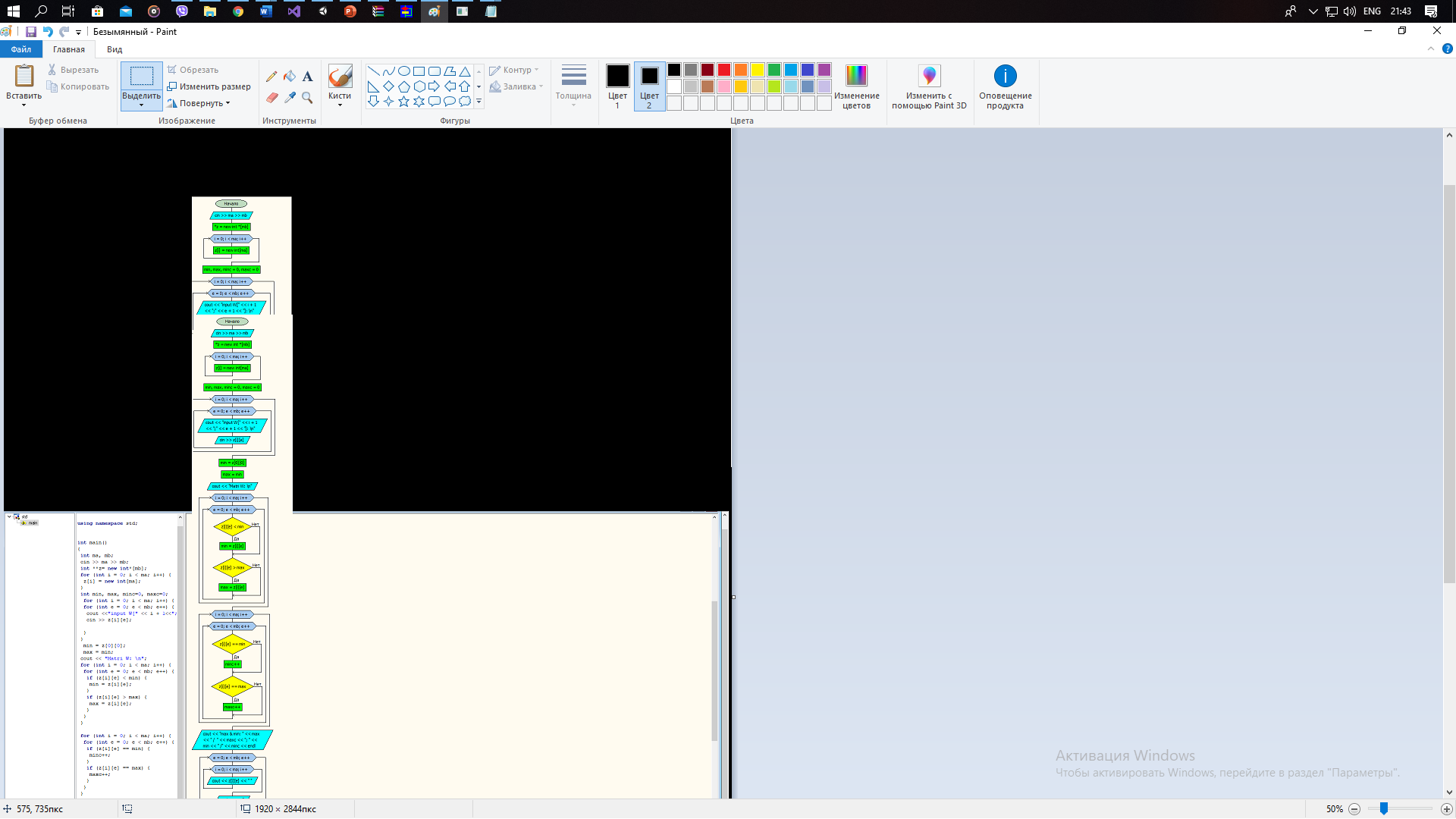
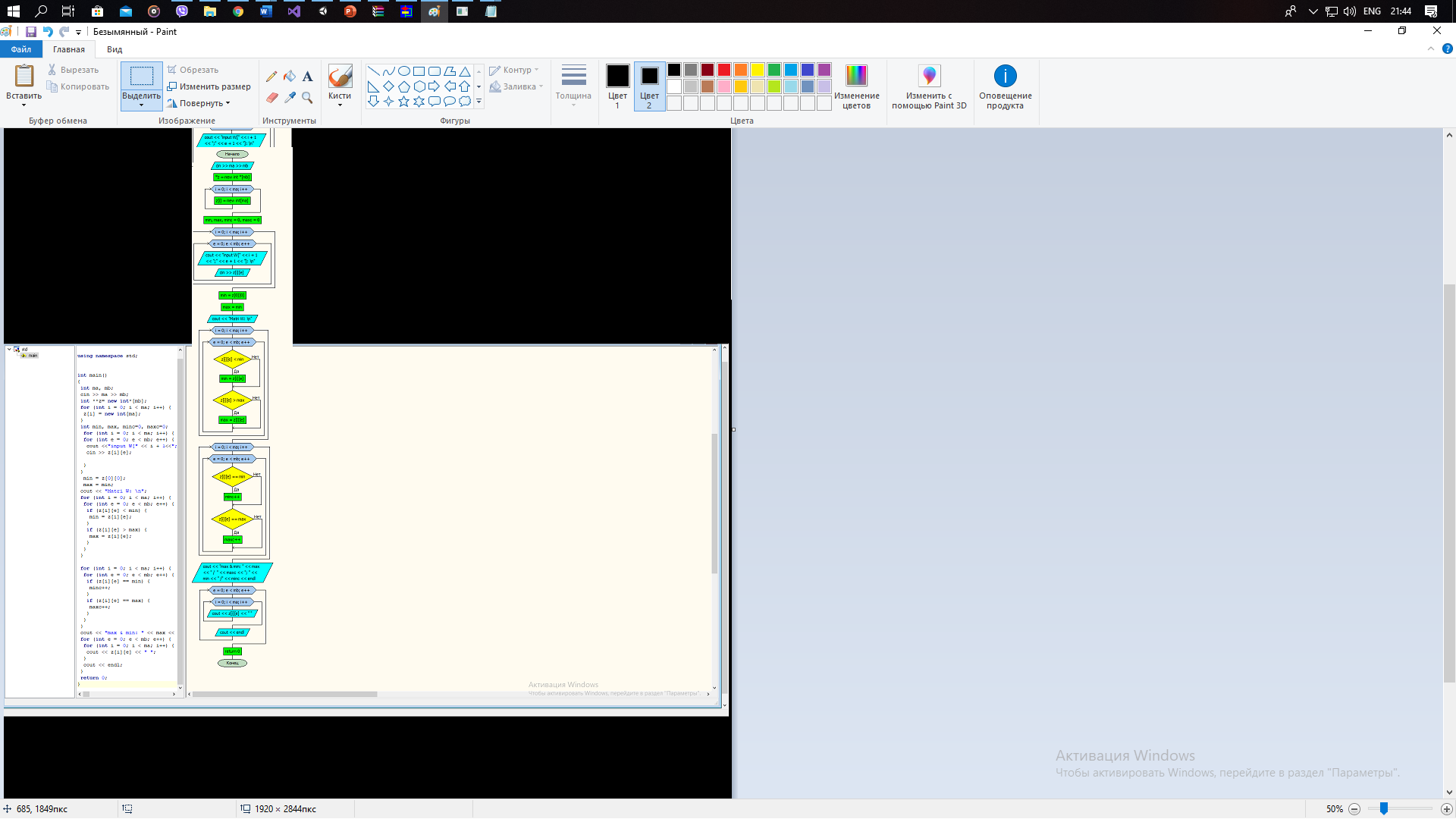
1. **Лабораторна робота №4. Двовимірні масиви. РОбота із матрицями**
2. **Завдання.** Розробити алгоритми та програми для заданих задач.

## Обчислити суму та кількість додатних елементів, що знаходяться над головною діагоналлю в заданій матриці W(5,5). Вивести вихiдну матрицю, суму та кількість додатних елементів.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

1. Найпростішим видом багатовимірного масиву є двовимірний масив. Двовимірний масив - це масив одновимірних масивів. Двовимірний масив оголошується наступним чином:
2. Двовимірні масиви зберігаються у вигляді матриці, де перший індекс відповідає за рядок, а другий -за стовпець. Це означає, що правий індекс змінюється швидше лівого, якщо рухатися по масиву в порядку розташування елементів в пам'яті. На рис. показано графічне представлення двовимірного масиву в пам'яті. Лівий індекс можна розглядати як покажчик на рядок.
3. Коли двовимірний масив використовується як аргумент функції, передається покажчик на перший елемент. Функція, яка отримує двовимірний масив, повинна, як мінімум, визначати розмір першого виміру, оскільки компілятору необхідно знати довжину кожного рядка для коректної індексації масиву.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ

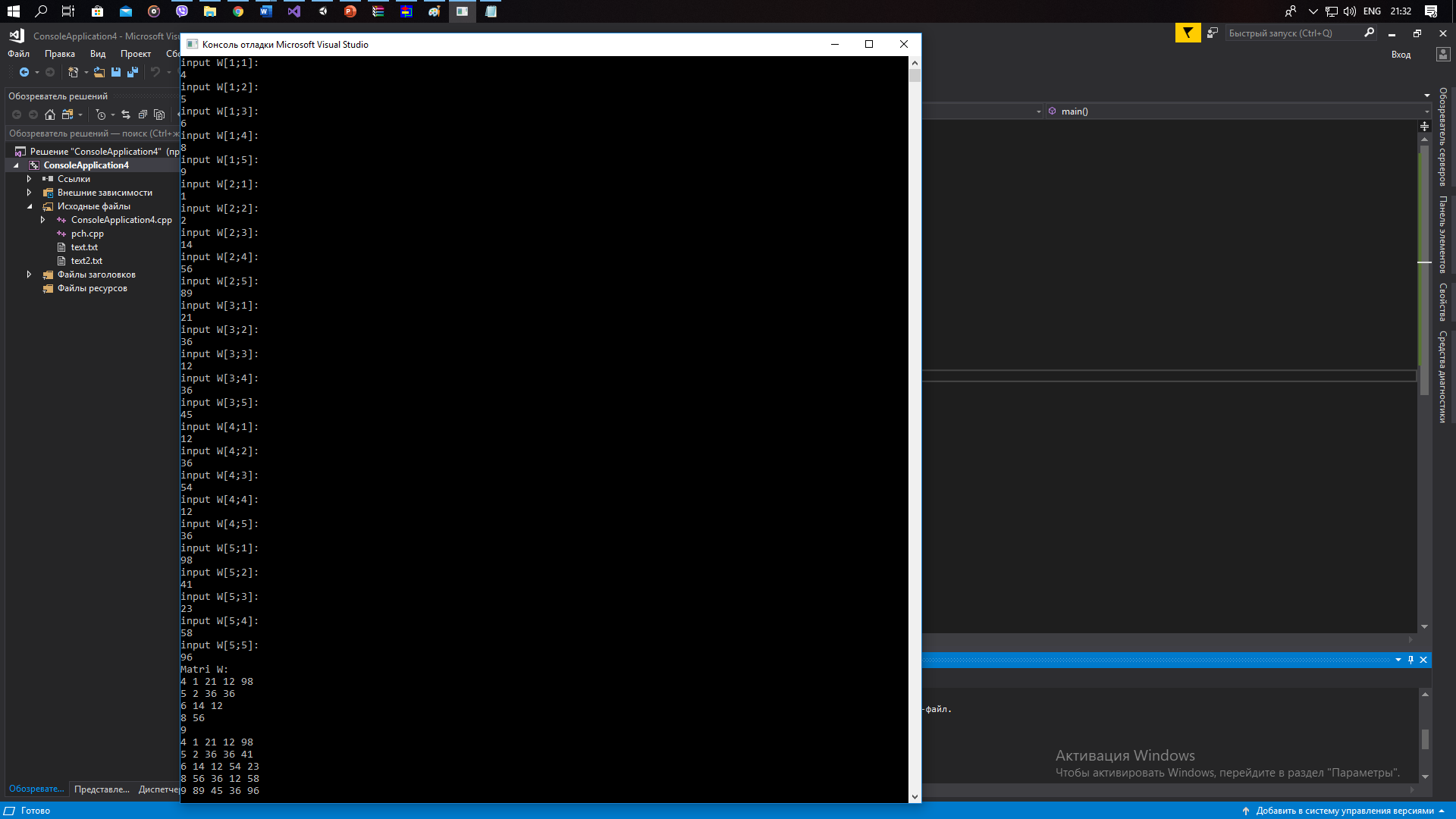
## Висновок

1. Під час виконання данного завданнь отримала навички з використання двовимірних масисів

## Додаток(a)

1. #include <iostream>
2. using namespace std;
3. int main()
4. {
5. int z[5][5];
6. for (int i = 0; i < 5; i++) {
7. for (int e = 0; e < 5; e++) {
8. cout <<"input W[" << i + 1<<";"<<e+1 << "]: \n";
9. cin >> z[i][e];
11. }
12. }
13. cout << "Matri W: \n";
14. int tt = 5;
15. for (int e = 0; e < 5; e++) {
16. for (int i = 0; i <tt; i++) {
17. cout << z[i][e]<<" ";
18. }
19. tt--;
20. cout << endl;
21. }
22. for (int e = 0; e < 5; e++) {
23. for (int i = 0; i < 5; i++) {
24. cout << z[i][e] << " ";
25. }
26. cout << endl;
27. }
28. return 0;
29. }

## Додаток(б)



**Частина 2**

1. 8. Дано дійсну матрицю *А* розмірності *n*x*m*. Знайти кількість входжень максимального та мінімального елементів.
2. *Вхідні дані:* *n =* 3; *m =* 4; .
3. *Вихідні дані:* *count\_min* = 2, *count\_max =* 3.

## Додаток(в)

#include <iostream>

using namespace std;

int main()

{

int ma, mb;

cin >> ma >> mb;

int \*\*z = new int\*[mb];

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

z[i] = new int[ma];

}

int min, max, minc = 0, maxc = 0;

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

for (int e = 0; e < mb; e++) {

cout << "input W[" << i + 1 << ";" << e + 1 << "]: \n";

cin >> z[i][e];

}

}

min = z[0][0];

max = min;

cout << "Matri W: \n";

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

for (int e = 0; e < mb; e++) {

if (z[i][e] < min) {

min = z[i][e];

}

if (z[i][e] > max) {

max = z[i][e];

}

}

}

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

for (int e = 0; e < mb; e++) {

if (z[i][e] == min) {

minc++;

}

if (z[i][e] == max) {

maxc++;

}

}

}

cout << "max & min: " << max << " / " << maxc << "; " << min << " /" << minc << endl;

for (int e = 0; e < mb; e++) {

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

cout << z[i][e] << " ";

}

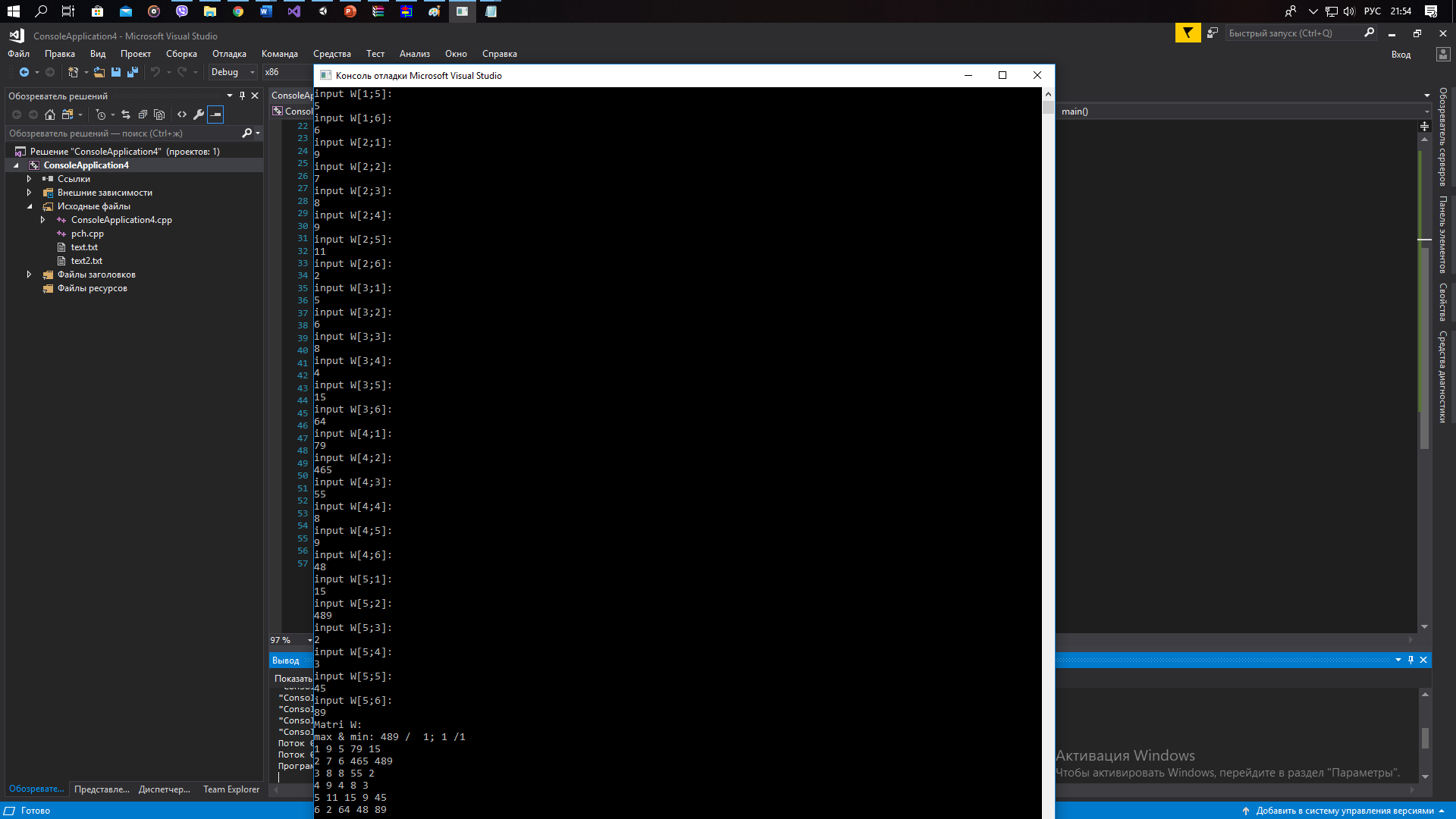
cout << endl;

}

return 0;

## }

## Додаток(г)



**Завдання 5**

**Лабораторна робота №5. Вказівники. Динамічна пам'ять. Двовимірні масиви.**

**Завдання:** *Реалізувати алгоритм та програму для розв’язання задачі відповідного варіанту*

**Завдання для самостійної роботи.**

Дано масив цілих чисел *a*[*i*], *і* = 1, … , *n*. Використовуючи вказівники, знайти кількість різних елементів в ньому.

*Вхідні дані:* *n* = 6; *a*[*i*] = {31, 31, 2, 0, 31, 2}.

*Вихідні дані:* *Кількість різних елементів =* 3.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

Вказівники

### У мові С++ є операція визначення адреси — &, за допомогою якої визначаються адреси комірки пам’яті, що містить задану змінну. Наприклад, якщо vr — ім’я змінної, то &vr — адреса цієї змінної.

У С++ також існують і змінні типу вказівник.

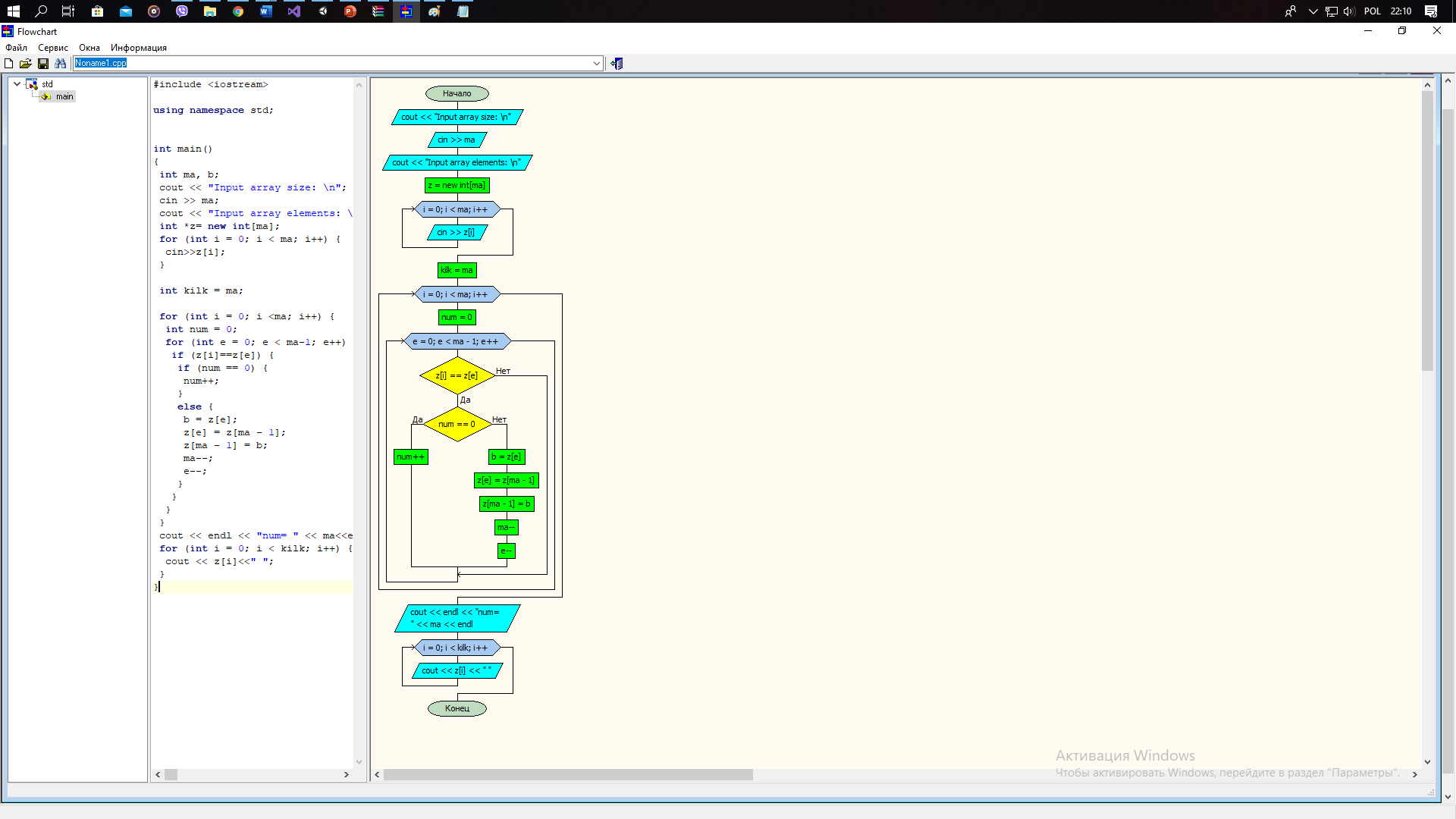
Вказівник — це тип даних, який використовується для зберігання адрес змінних і об'єктів.

Значенням змінної типу вказівник є адреса змінної або об'єкта.

Опис змінних типу вказівник здійснюється за допомогою операторів наступної форми:

<тип> \*<ім'я вказівника на змінну заданого типу>;

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ



## Висновок

1. Під час виконання данного завданнь отримала навички робити з вказіаниками та динамічною пам’ятю
2. **Додаток(а)**

#include <iostream>

using namespace std;

int main()

{

int ma, b;

cout << "Input array size: \n";

cin >> ma;

cout << "Input array elements: \n";

int \*z = new int[ma];

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

cin >> z[i];

}

int kilk = ma;

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

int num = 0;

for (int e = 0; e < ma - 1; e++) {

if (z[i] == z[e]) {

if (num == 0) {

num++;

}

else {

b = z[e];

z[e] = z[ma - 1];

z[ma - 1] = b;

ma--;

e--;

}

}

}

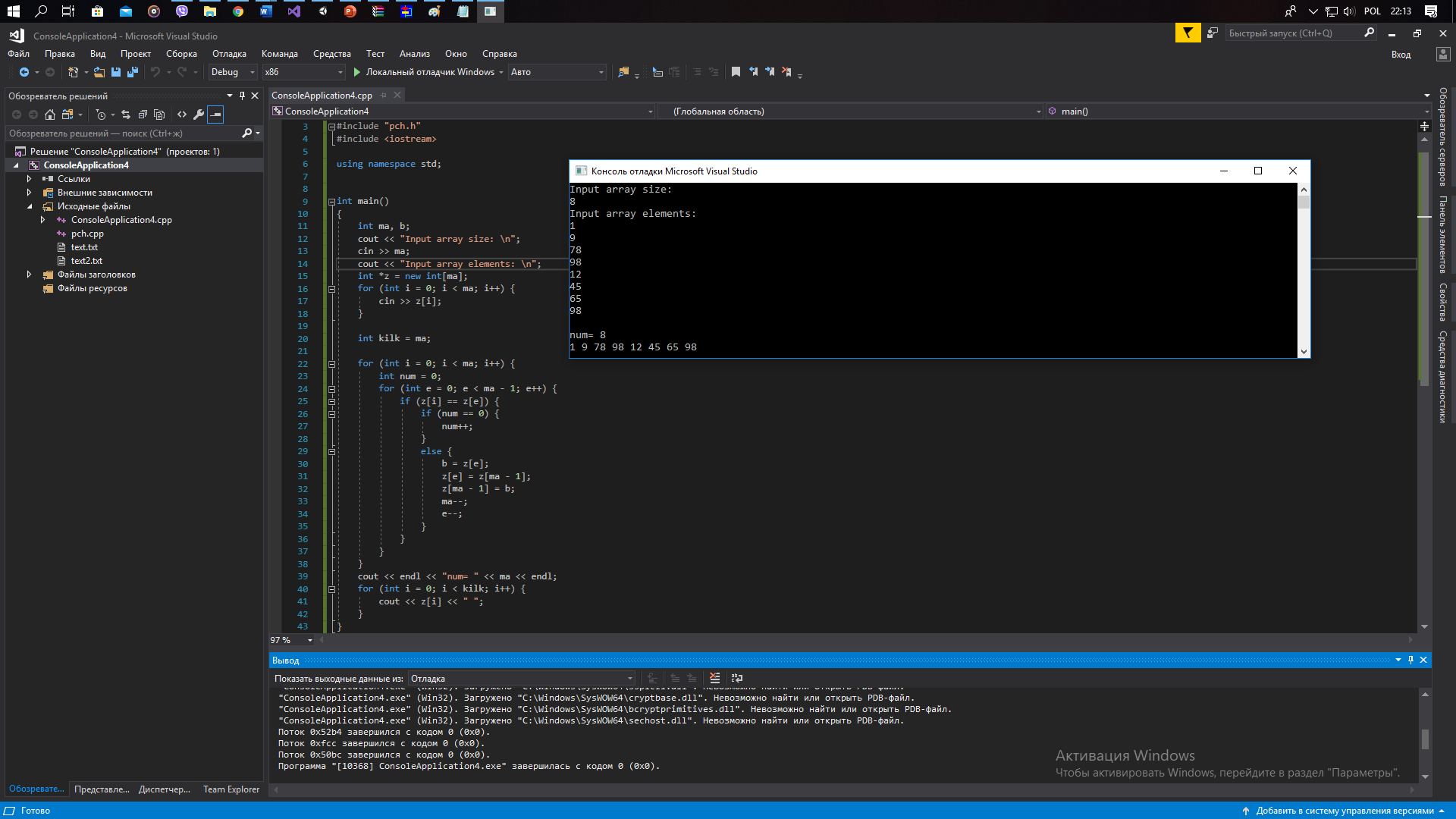
}

cout << endl << "num= " << ma << endl;

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

cout << z[i] << " ";

}

1. }
2. **Додаток(б)**
3. 

**Завдання 6**

**Лабораторна робота №6. Вказівники. Динамічна пам'ять. Двовимірні масиви.**

**Завдання:**

Для наведених нижче задач

1. зробити математичну постановку;
2. скласти алгоритм програми;
3. скласти програму для обчислення;
4. проаналізувати результати роботи програми;
5. оформити протокол.

**Задача 1. Двовимірні масиви. Задача про вибори.**

Нехай шість населених пунктів позначені номерами від 1 до 6 (величина **k**), а п’ять кандидатів – номерами від 1 до 5 (величина **п**). Кількість голосів, набраних кандидатами у кожному пункті визначається формулою ***akn***=random(10і+50), де і - номер варіанта. (Функція random(*п*) описана у модулі stdlib.h. Перед використанням функції random(*п*) треба записати на початку програми функцію randomize() щоб під час виконання програми кожного разу отримувати різні випадкові числа.) Вивести на екран таблицю результатів голосування, де у рядках є дані з населених пунктів, а у стовпцях - дані щодо конкретних кандидатів. Визначити і вивести значення величин з додаткового завдання. Створити одновимірний масив з шуканими даними.

1. Хто з кандидатів набрав максимальну кількість голосів у друго­му населеному пункті?

## ТЕОРЕТИЧНІ ВІДОМОСТІ

Оператор адреси &

При виконанні ініціалізації змінної, їй автоматично привласнюється вільний адреса пам'яті, і, будь-яке значення, яке ми присвоюємо змінної, зберігається в цю адресу пам'яті. наприклад:

int b;

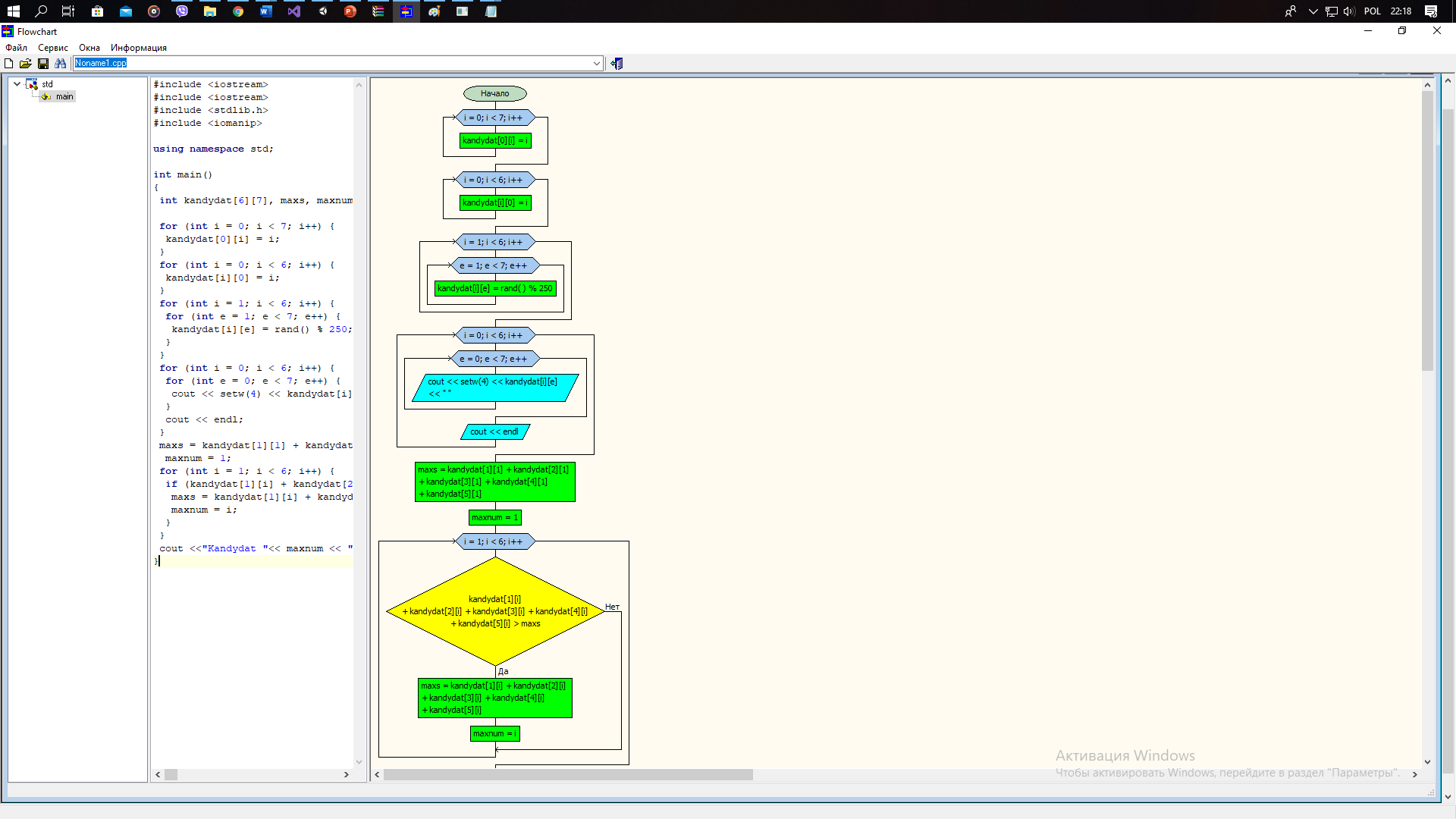
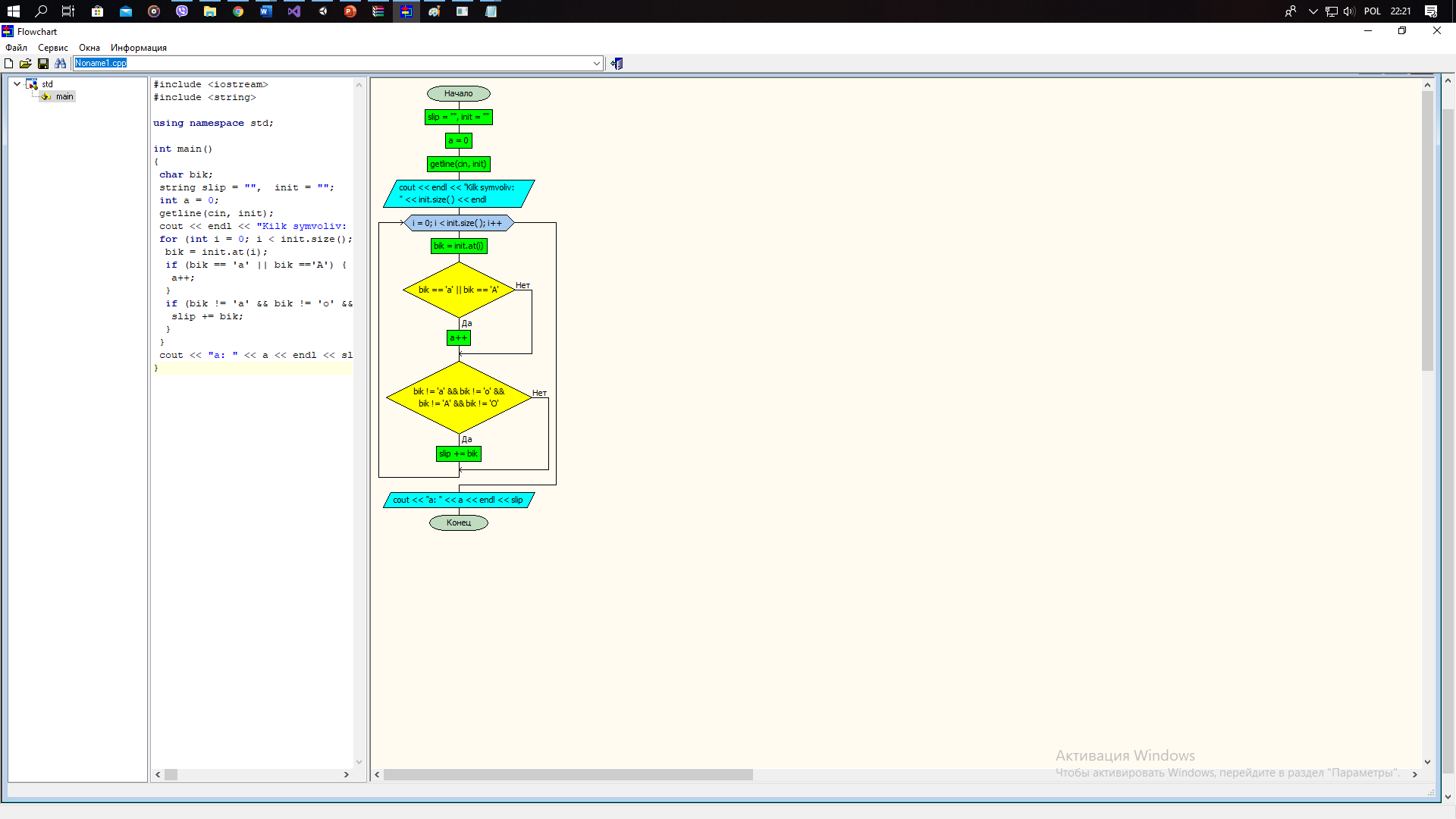
1

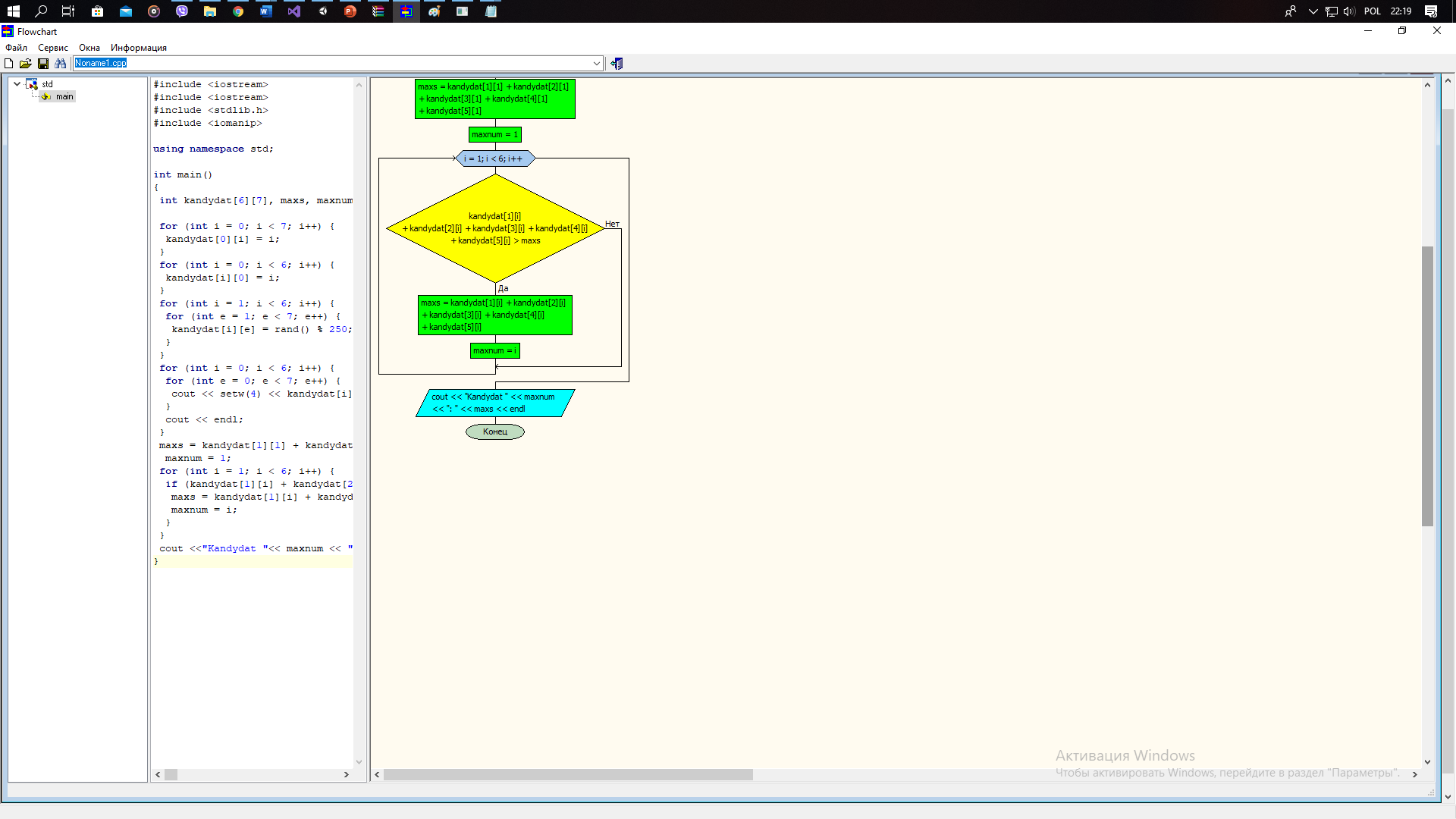
int b;

При виконанні цього стейтмента процесором, виділяється частина оперативної пам'яті. Як приклад припустимо, що змінної b присвоюється осередок пам'яті під номером 150. Всякий раз, коли програма зустрічає змінну b в вираженні або в стейтменте, вона розуміє, що для того, щоб отримати значення - їй потрібно заглянути в комірку пам'яті під номером 150.

Хороша новина: нам не потрібно турбуватися про те, які конкретно адреси пам'яті виділені для певних змінних. Ми просто посилаємося на змінну через присвоєний їй ідентифікатор, а компілятор конвертує це ім'я в відповідний адресу пам'яті.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ



## Висновок

1. Під час виконання данного завданнь отримала навички робити з вказіаниками та динамічною пам’ятю
2. **Додаток(а)**

#include <iostream>

#include <iostream>

#include <stdlib.h>

#include <iomanip>

using namespace std;

int main()

{

int kandydat[6][7], maxs, maxnum;

for (int i = 0; i < 7; i++) {

kandydat[0][i] = i;

}

for (int i = 0; i < 6; i++) {

kandydat[i][0] = i;

}

for (int i = 1; i < 6; i++) {

for (int e = 1; e < 7; e++) {

kandydat[i][e] = rand() % 250;

}

}

for (int i = 0; i < 6; i++) {

for (int e = 0; e < 7; e++) {

cout << setw(4) << kandydat[i][e] << " ";

}

cout << endl;

}

maxs = kandydat[1][1] + kandydat[2][1] + kandydat[3][1] + kandydat[4][1] + kandydat[5][1];

maxnum = 1;

for (int i = 1; i < 6; i++) {

if (kandydat[1][i] + kandydat[2][i] + kandydat[3][i] + kandydat[4][i] + kandydat[5][i] > maxs) {

maxs = kandydat[1][i] + kandydat[2][i] + kandydat[3][i] + kandydat[4][i] + kandydat[5][i];

maxnum = i;

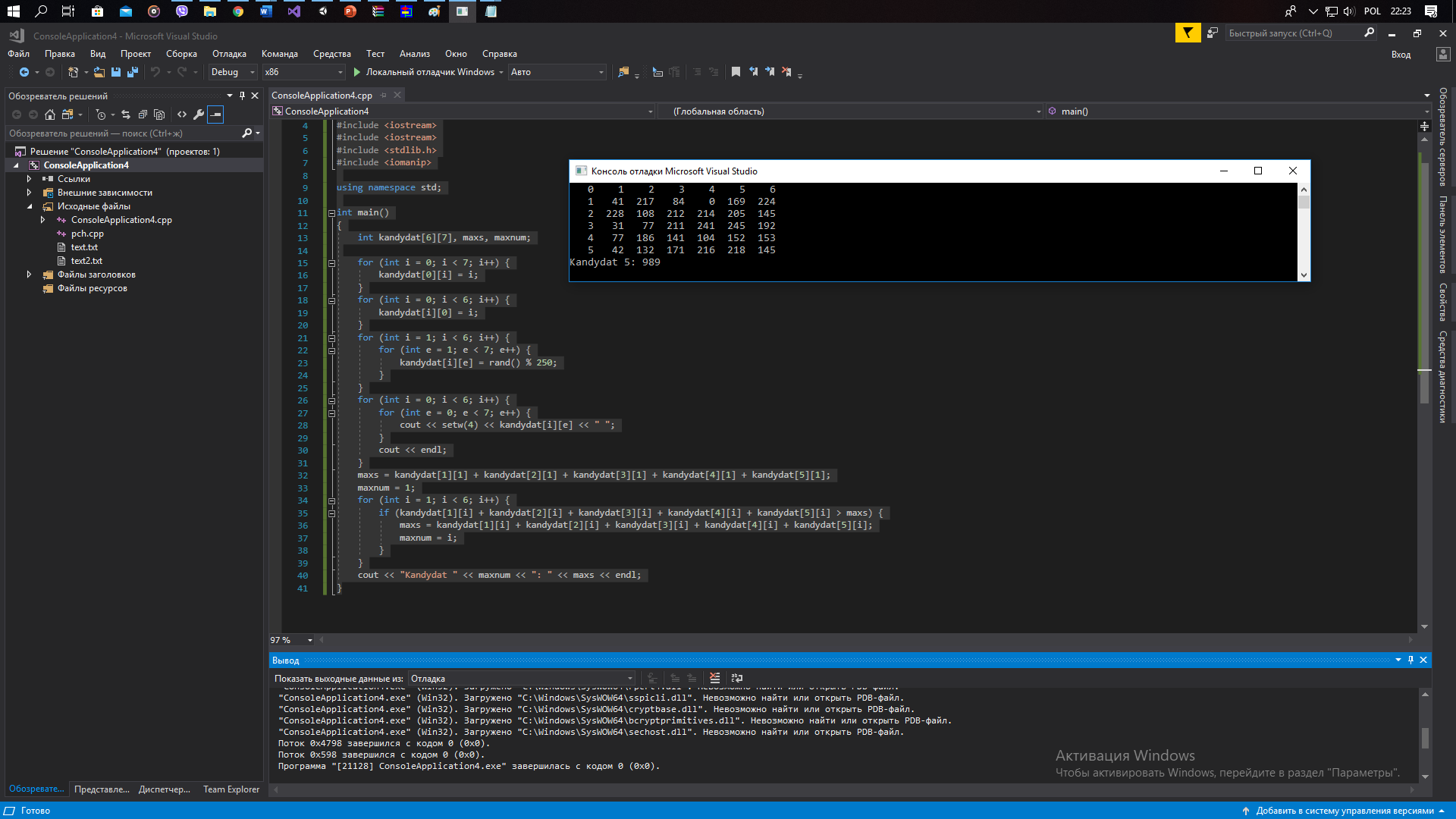
}

}

cout << "Kandydat " << maxnum << ": " << maxs << endl;

}

1. **Додаток(б)**



**Задача 2. Дані типу рядки.**

Ввести прізвище, ім’я та по батькові як одне дане типу рядок. Визначити довжину рядка і кількість букв «а» у ньому. Виконати додатково завдання свого варіанта двома способами: а) розглядаючи рядок як масив символів;

Вилучити усі букви «а» та «о» з прізвища.

**Додаток(в)**

#include <iostream>

#include <string>

using namespace std;

int main()

{

char bik;

string slip = "", init = "";

int a = 0;

getline(cin, init);

cout << endl << "Kilk symvoliv: " << init.size() << endl;

for (int i = 0; i < init.size(); i++) {

bik = init.at(i);

if (bik == 'a' || bik == 'A') {

a++;

}

if (bik != 'a' && bik != 'o' && bik != 'A' && bik != 'O') {

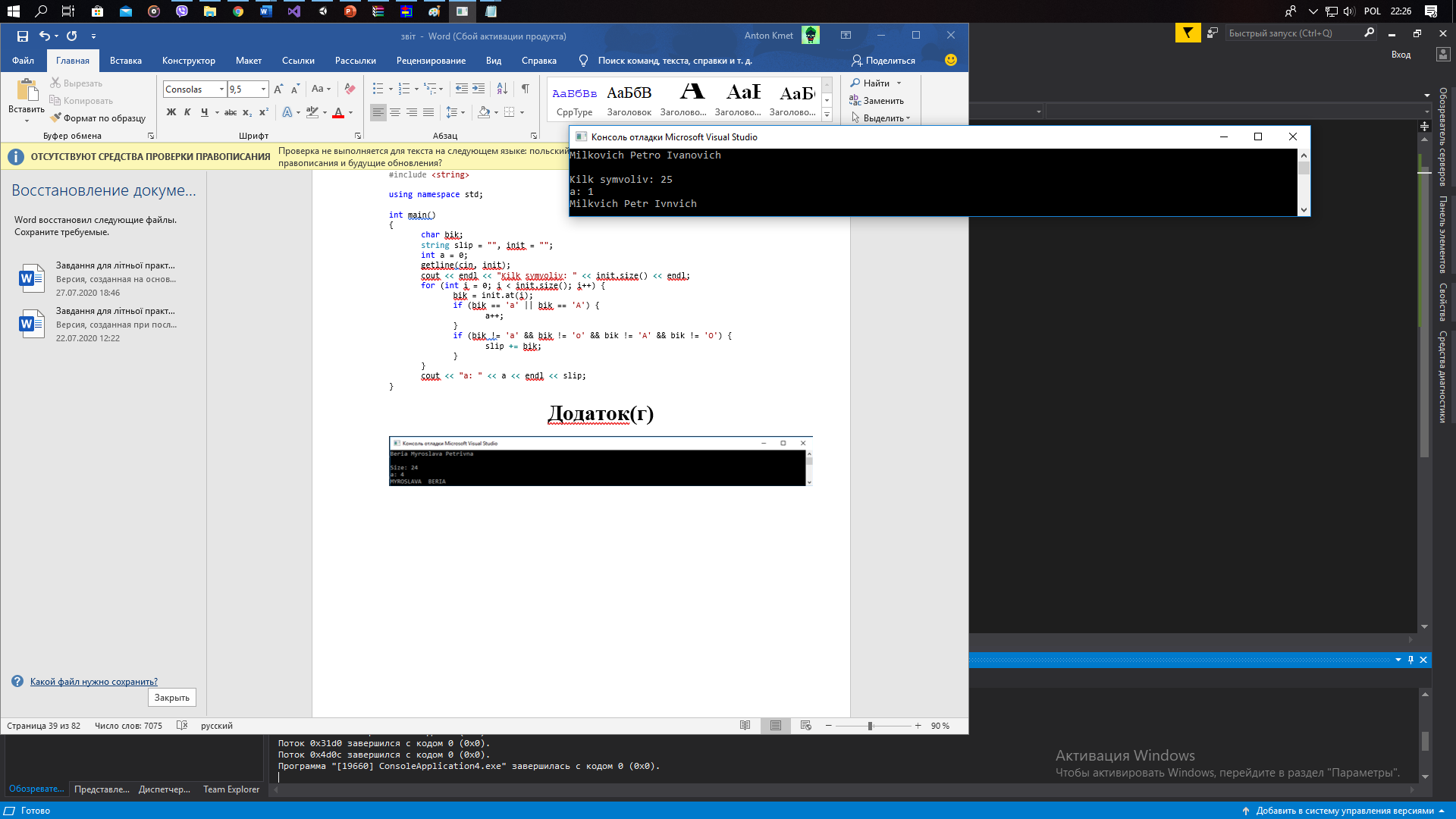
slip += bik;

}

}

cout << "a: " << a << endl << slip;

}

1. **Додаток(г)**
2. 

**Завдання 7**

**Лабораторна робота №7 ФУнкції.**

**Функції користувача**

**Завдання:**

Для наведених нижче задач

1. зробити математичну постановку;
2. скласти алгоритм програми;
3. скласти програму для обчислення;
4. проаналізувати результати роботи програми;
5. оформити протокол.

**Функції користувача.**

Утворити і вивести масив А, елементи якого описані формулою , m,n=1…4, i – номер варіанта. Скласти функцію користувача для пошуку даних у цьому масиві. Критерії пошуку взяти відповідно свого варіанту, якщо критерії пошуку не відповідають вашим даним вибрати свої значення для пошуку. Шукані дані вивести в головній функції.



1. Обчислити добуток від’ємних елементів. Визначити індекси максимального елемента.

**ТЕОРЕТИЧНІ ВІДОМОСТІ**

Функція - це самостійна одиниця програми, яка спроектована для реалізації конкретної підзадачі.

Функція є підпрограмою, яка може міститися в основній програмі, а може бути створена окремо (в бібліотеці). Кожна функція виконує в програмі певні дії.

Сигнатура функції визначає правила використання функції. Зазвичай сигнатура є опис функції, що включає ім'я функції, перелік формальних параметрів з їх типами і тип значення, що повертається.

Семантика функції визначає спосіб реалізації функції. Зазвичай являє собою тіло функції.Для функ. , яка не залежить від жодних параметрів, у круглих дужках записують службове слово void. Назву функ. Надає користувач за правилом створення ідентифікаторів. У списку формальних параметрів через кому записують змінні, зазначаючи їх тип(для кожної змінної окремо!). Імена змінних можна опускати. Якщо функ. Не набуває жодних значень, то список формальних параметрів може бути відсутній(круглі дужки опускати не можна). Приклад: float S(int k, float c); - функ. S типу float, залежить від 2 змінних типів int і float.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ

## 

## Висновок

Під час виконання данного завданнь отримала навички з створення та використання Функцій користувача

1. **Додаток(а)**

#include <iostream>

#include <iostream>

#include<cmath>

using namespace std;

void cinn(int t[4][4]) {

for (int i = 0; i < 4; i++) {

for (int e = 0; e < 4; e++) {

t[i][e] = 2 \* rand() % 8 \* i + 8 \* (e + 1);

cout << t[i][e] << " ";

}

cout << endl;

}

cout << endl;

}

void sort(int array[4][4]) {

int max = array[0][0], x = 0, y = 0;

for (int i = 0; i < 4; i++) {

for (int e = 0; e < 4; e++) {

if (array[i][e] > max) {

max = array[i][e];

x = i;

y = e;

}

}

}

int dob = 1;

for (int i = 0; i < 4; i++) {

for (int e = 0; e < 4; e++) {

if (array[i][e] < 0) {

dob \*= array[i][e];

}

}

}

cout << "max [" << x << ";" << y << "]=" << max << endl << "dob elem<0= " << dob;

}

int main()

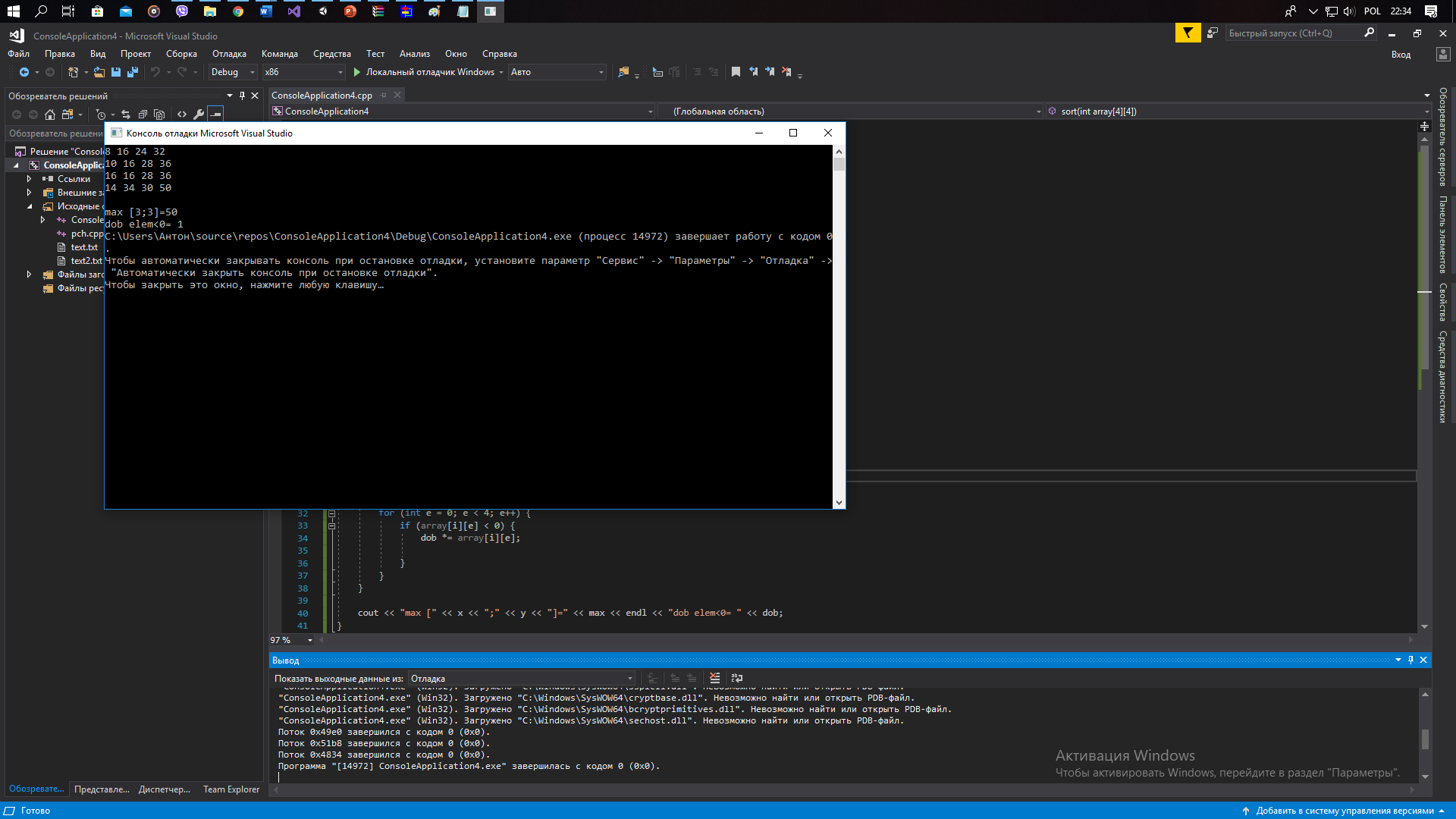
{

int re[4][4];

cinn(re);

sort(re);

}

1. **Додаток(б)**
2. 

**Завдання 8**

1. **Лабораторна робота №8 ФУнкції. Функції користувача**
2. **Завдання:** *Реалізувати алгоритм та програму для розв’язання задачі відповідного варіанту*
3. Дано координати вершин двох трикутників. Написавши функцію *Area*, що визначає площу трикутника, заданого координатами вершин, визначити який з них має більшу площу.
4. *Вхідні дані:* *А1*(0, 0); *В1*(3, 3); *С1*(6, 0); *А2*(0, 0); *В2*(4, 4); *С2*(8, 0).
5. *Вихідні дані:* *Більшу площу має другий трикутник*.
6. **ТЕОРЕТИЧНІ ВІДОМОСТІ**
7. Функція - це самостійна одиниця програми, яка спроектована для реалізації конкретної підзадачі.
8. Функція є підпрограмою, яка може міститися в основній програмі, а може бути створена окремо (в бібліотеці). Кожна функція виконує в програмі певні дії.
9. Сигнатура функції визначає правила використання функції. Зазвичай сигнатура є опис функції, що включає ім'я функції, перелік формальних параметрів з їх типами і тип значення, що повертається.
10. Семантика функції визначає спосіб реалізації функції. Зазвичай являє собою тіло функції.Для функ. , яка не залежить від жодних параметрів, у круглих дужках записують службове слово void. Назву функ. Надає користувач за правилом створення ідентифікаторів. У списку формальних параметрів через кому записують змінні, зазначаючи їх тип(для кожної змінної окремо!). Імена змінних можна опускати. Якщо функ. Не набуває жодних значень, то список формальних параметрів може бути відсутній(круглі дужки опускати не можна). Приклад: float S(int k, float c); - функ. S типу float, залежить від 2 змінних типів int і float.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ

## 

## Висновок

Під час виконання данного завданнь отримала навички з створення та використання Функцій користувача

1. **Додаток(а)**

#include <iostream>

#include<cmath>

using namespace std;

void area(int x1, int x2, int x3, int y1, int y2, int y3, int &s) {

int a = sqrt(pow(x1 - x2, 2) + pow(y1 - y2, 2));

int b = sqrt(pow(x2 - x3, 2) + pow(y2 - y3, 2));

int c = sqrt(pow(x3 - x1, 2) + pow(y3 - y1, 2));

s = sqrt((a + b + c) \* (b + c - a) \* (a + c - b) \* (a + b - c)) \* 1 / 4;

}

int main()

{

int x1, x2, x3, x11, x22, x33, y1, y2, y3, y11, y22, y33, s1 = 0, s2 = 0;

cout << "A1(x y) \n";

cin >> x1 >> y1;

cout << "B1(x y) \n";

cin >> x2 >> y2;

cout << "C1(x y) \n";

cin >> x3 >> y3;

cout << "A2(x y) \n";

cin >> x11 >> y11;

cout << "B2(x y) \n";

cin >> x22 >> y22;

cout << "C2(x y) \n";

cin >> x33 >> y33;

area(x1, x2, x3, y1, y2, y3, s1);

area(x11, x22, x33, y11, y22, y33, s2);

if (s1 > s2) {

cout << "triangles are similar";

}

if (s1 == s2) {

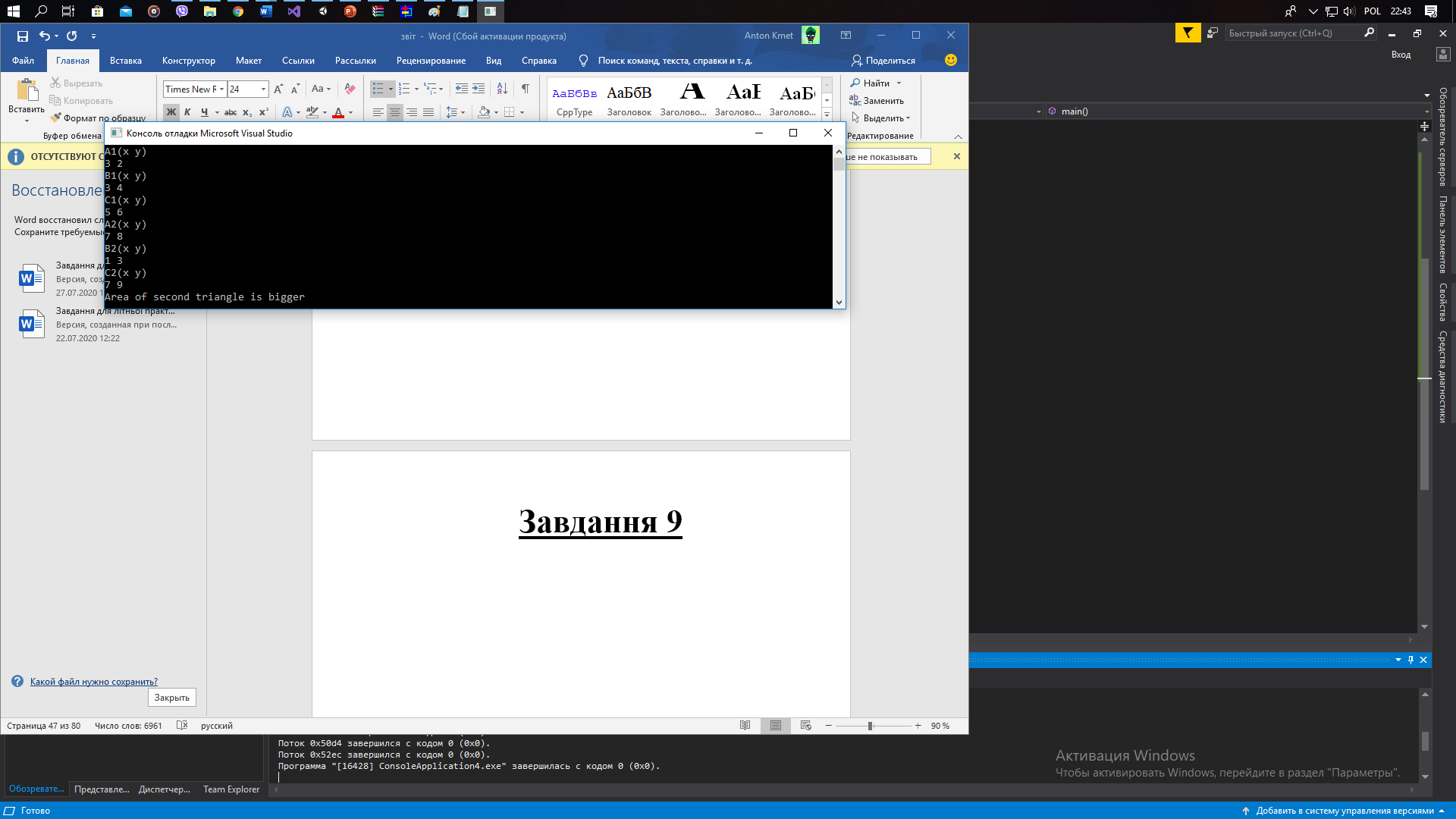
cout << "Area of second triangle is bigger";

}

else {

cout << "Area of second triangle is bigger";

}

1. }
2. **Додаток(б)**
3. 

**Завдання 9**

1. **Лабораторна робота №9 Динамічні Структури.**
2. **Списки. Стеки. Черги**
3. Дано масив цілих чисел *a*[*i*], *і* = 1, … , *n, n –* парне. Виділіть для нього необхідний обсяг динамічної пам'яті. Заповнити його за допомогою генератора псевдовипадкових чисел з відрізка [-10, -3]. Поміняти місцями його половини, а саме: перший елемент з останнім, другий елемент з передостаннім і т.д. Вивільніть динамічну пам'ять.

*Вхідні дані:* *n* = 6, *a*[*i*] = {-6, -3, -7, -10, -3, -5}.

*Вихідні дані:* *a*[*i*] = {-5, -3, -10, -7, -3, -6}.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

**Стек - це структура даних, яка працює за принципом FILO (first in - last out; перший прийшов - останній пішов). У C ++ вже є готовий шаблон - stack.**

**У стеці елемент, який увійшов найперший - вийде найостаннішим. Виходить, якщо ви додали три елементи в стек першим буде видалений останній доданий елемент.**

**стек картінкаНа малюнку 1 ви можете побачити 6 чисел: 6, 5, 1, 2, 5, 9. До речі витягати їх будемо в такому ж порядку. Наприклад щоб витягти число 1 нам доведеться спочатку витягти числа 6 і 5, а потім вже 1. Так як це стек, ці числа ми додавали в зворотному порядку. Якщо бути точним ось так: 9, 5, 2, 1, 5, 6.**

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ

## 

## Висновок

1. Під час виконання данного завданнь отримала навички з використання та створення динамічних Структур, Списків, Стеків та Черги
2. **Додаток(А)**

#include <iostream>

using namespace std;

unsigned int rt()

{

static unsigned int seed = -46869;

seed = (-45996 \* seed + 64568);

return seed % 13 - 10;

}

int main()

{

cout << "Input array size: " << endl;

int a;

cin >> a;

int \*mas = new int[a];

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

mas[i] = rt();

}

cout << endl;

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

cout << mas[i] << " ";

}

int bom = a - 1;

for (int i = 0; i <= a / 2; i++) {

int tepo = mas[i];

mas[i] = mas[bom];

mas[bom] = tepo;

bom--;

}

cout << endl << "Result: " << endl;

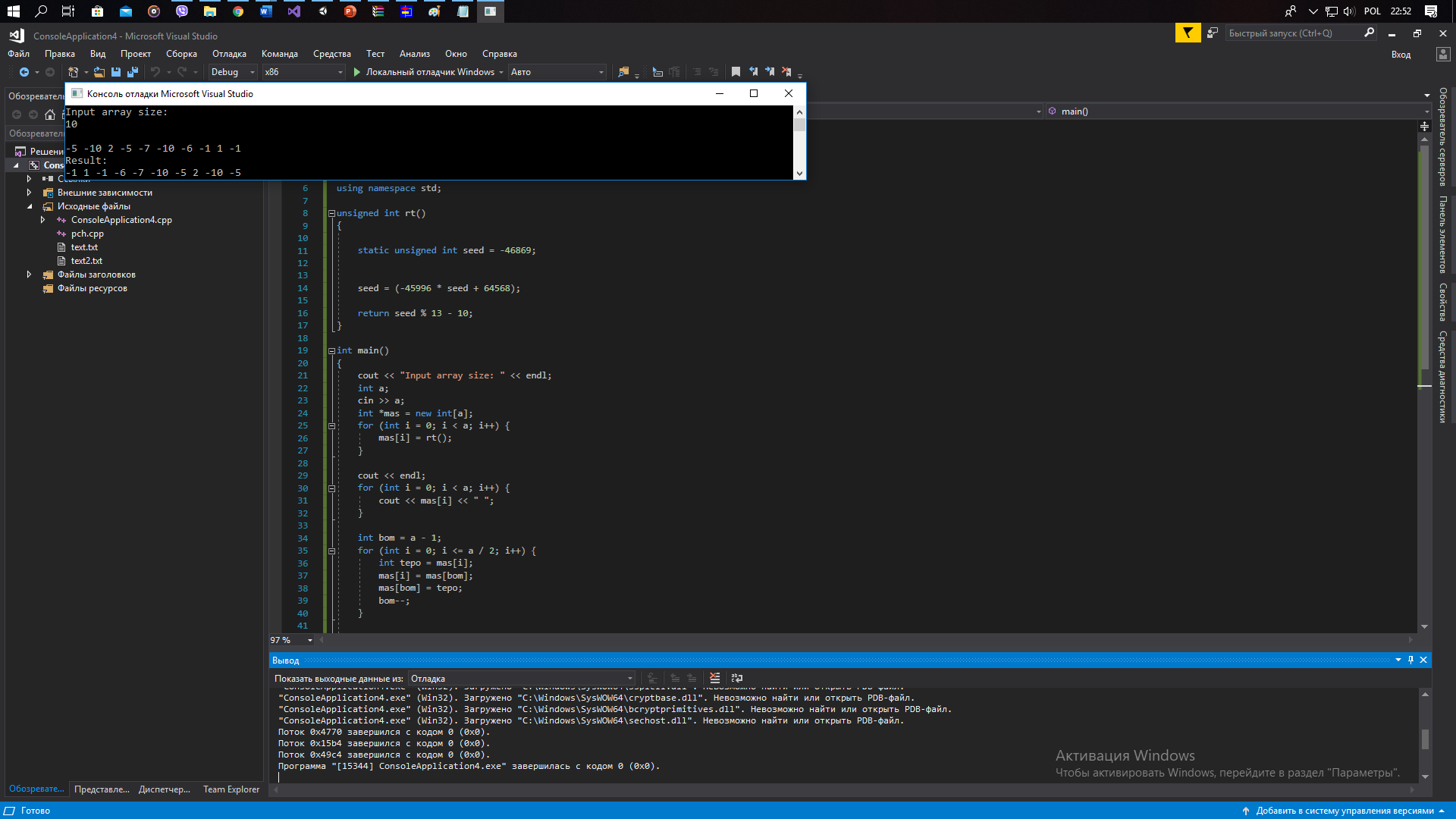
for (int i = 0; i < a; i++) {

cout << mas[i] << " ";

}

delete mas;

1. }**Додаток(Б)**



**Завдання 10**

1. **Лабораторна робота №10 ВИКОРИСТАННЯ СТРУКТУР**
2. **Завдання:** *Реалізувати алгоритм та програму для розв’язання задачі відповідного варіанту*
3. **Задачі для самостійної роботи.**
4. 8. Розклад поїздів містить такі відомості:
5. – номер поїзда;
6. – станція призначення;
7. – відстань до станції призначення;
8. – час відправлення (години, хвилини);
9. – час прибуття в кінцевий пункт (години, хвилини);
10. – час у дорозі.
11. Визначити: а) поїзд, що має найтриваліший час шляху (повідомити його номер і станцію призначення); б) чи є потяг до Києва, який відправляється не раніше 6 і не пізніше 10 годин ранку. Якщо так, то повідомити всі наявні відомості про нього. (Якщо таких поїздів декілька, то про кожен з них.)

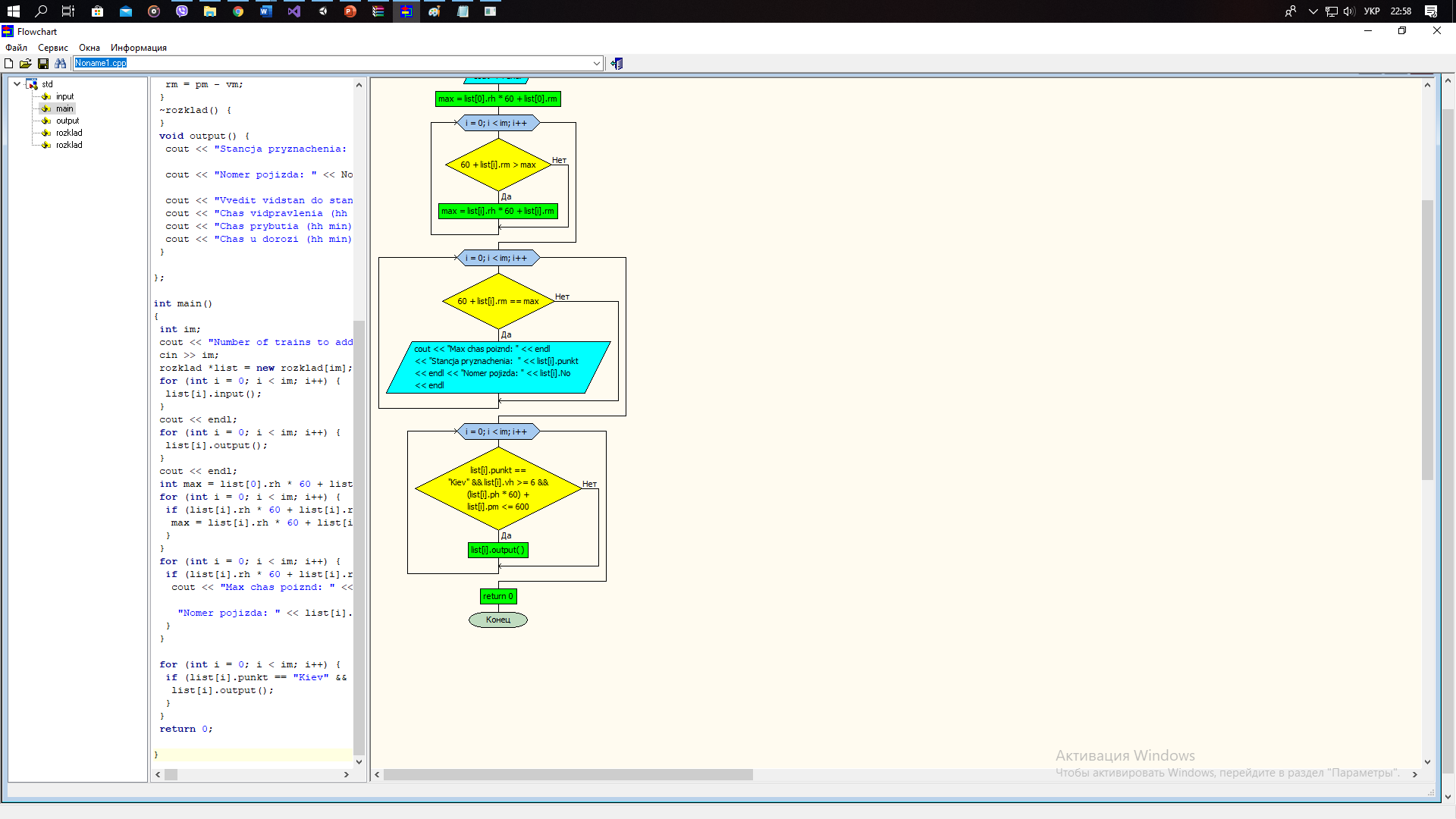
## ТЕОРЕТИЧНІ ВІДОМОСТІ

Раніше для визначення класів ми використовували ключове слово class. Однак C ++ надає ще один спосіб для визначення користувача типів, який полягає у використанні структур. Даний спосіб був успадкований мовою С ++ ще від мови Сі.

Структура в мові C ++ є похідний тип даних, який представляє якусь певну сутність, також як і клас. Нерідко структури пріменітелько до С ++ також називають класами. І в реальності відмінності між ними не такі великі.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ





## Висновок

1. Під час виконання данного завданнь отримала навички роботи з структурами
2. **Додаток(а)**

#include <iostream>

#include <string>

using namespace std;

struct rozklad {

string punkt;

int No, vdo, vh, vm, ph, pm, rh, rm;

rozklad() {

}

void input() {

cout << "Vvedit stancju pryznachenia: " << endl;

cin >> punkt;

cout << "Vvedit nomer pojizda: \n";

cin >> No;

cout << "Vvedit vidstan do stancji pryznachenia \n";

cin >> vdo;

cout << "Chas vidpravlenia (hh min) \n";

cin >> vh >> vm;

cout << "Chas prybutia (hh min) \n";

cin >> ph >> pm;

if (pm < vm) {

ph--;

pm += 60;

}

rh = ph - vh;

rm = pm - vm;

}

~rozklad() {

}

void output() {

cout << "Stancja pryznachenia: " << punkt << endl;

cout << "Nomer pojizda: " << No << endl;

cout << "Vvedit vidstan do stancji pryznachenia " << vdo << endl;

cout << "Chas vidpravlenia (hh min) " << vh << " " << vm << endl;

cout << "Chas prybutia (hh min) " << ph << " " << pm << endl;

cout << "Chas u dorozi (hh min) " << rh << " " << rm << endl;

}

};

int main()

{

int im;

cout << "Number of trains to add" << endl;

cin >> im;

rozklad \*list = new rozklad[im];

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

list[i].input();

}

cout << endl;

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

list[i].output();

}

cout << endl;

int max = list[0].rh \* 60 + list[0].rm;

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

if (list[i].rh \* 60 + list[i].rm > max) {

max = list[i].rh \* 60 + list[i].rm;

}

}

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

if (list[i].rh \* 60 + list[i].rm == max) {

cout << "Max chas poiznd: " << endl << "Stancja pryznachenia: " << list[i].punkt << endl <<

"Nomer pojizda: " << list[i].No << endl;

}

}

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

if (list[i].punkt == "Kiev" && list[i].vh >= 6 && (list[i].ph \* 60) + list[i].pm <= 600) {

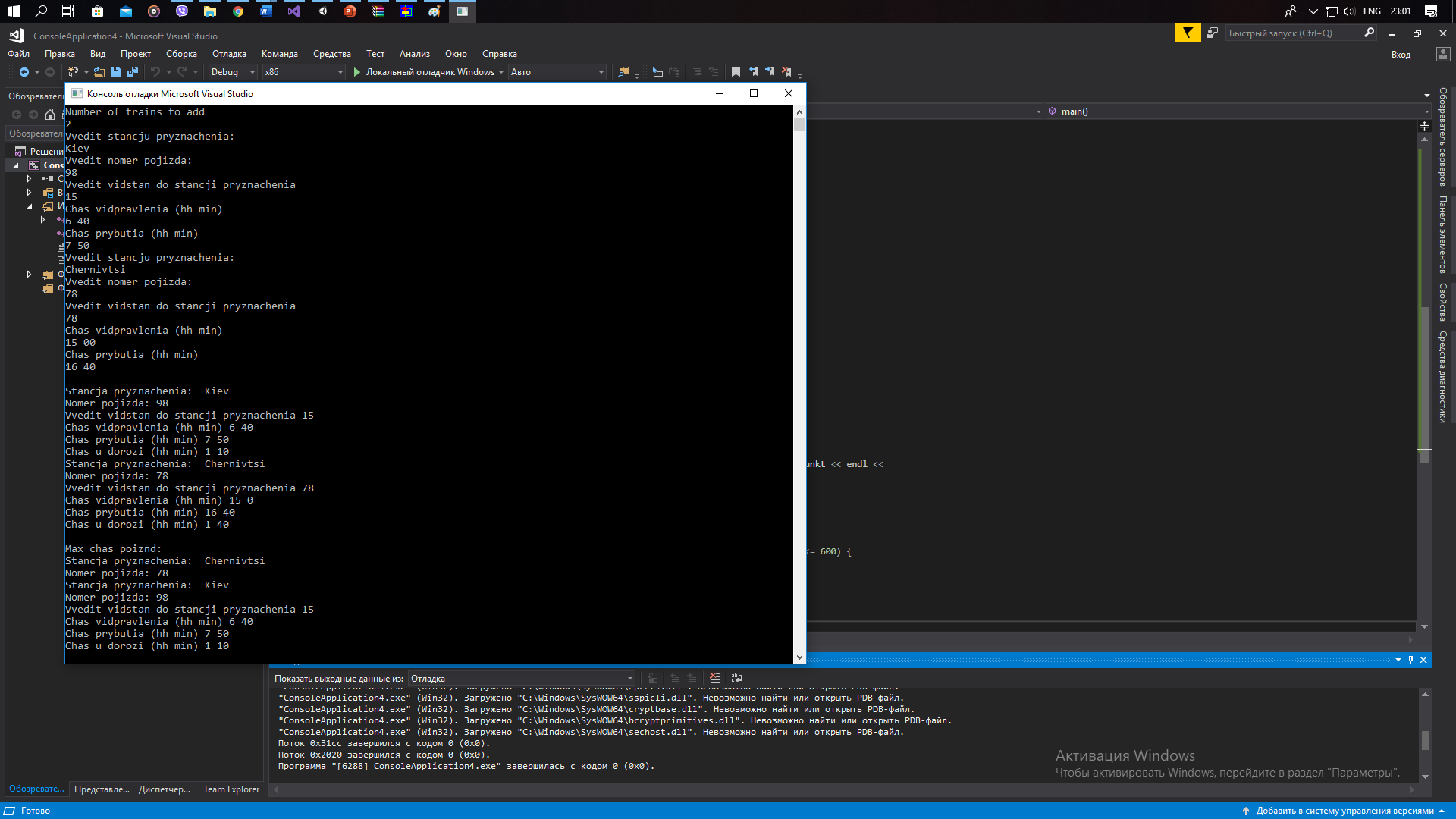
list[i].output();

}

}

return 0;

}

1. **Додаток(б)**

**Завдання 11**

**Лабораторна робота №11. Структури. Списки. Стеки. Черги.**

## *Завдання до лабораторної роботи*

*Лабораторну роботу виконати з використанням структурованих засобів представлення даних мови С(мови С++). Згідно варіанту виконати наступне:*

1. *Описати структуру представлення елементів даних;*
2. *Розробити програму введення даних з клавіатури та їх записати у файл;*
3. *Розробити програму розв’язання задачі, вхідні дані прочитати із файлу в динамічний масив, результати виконання вивести та екран та записати у файл;*
4. *Оформити звіт.*

Кількість м'ячів, забитих та пропущених футбольною командою в кожній з 22 ігор, записано в масиві, елементами якого є величини типу структури.

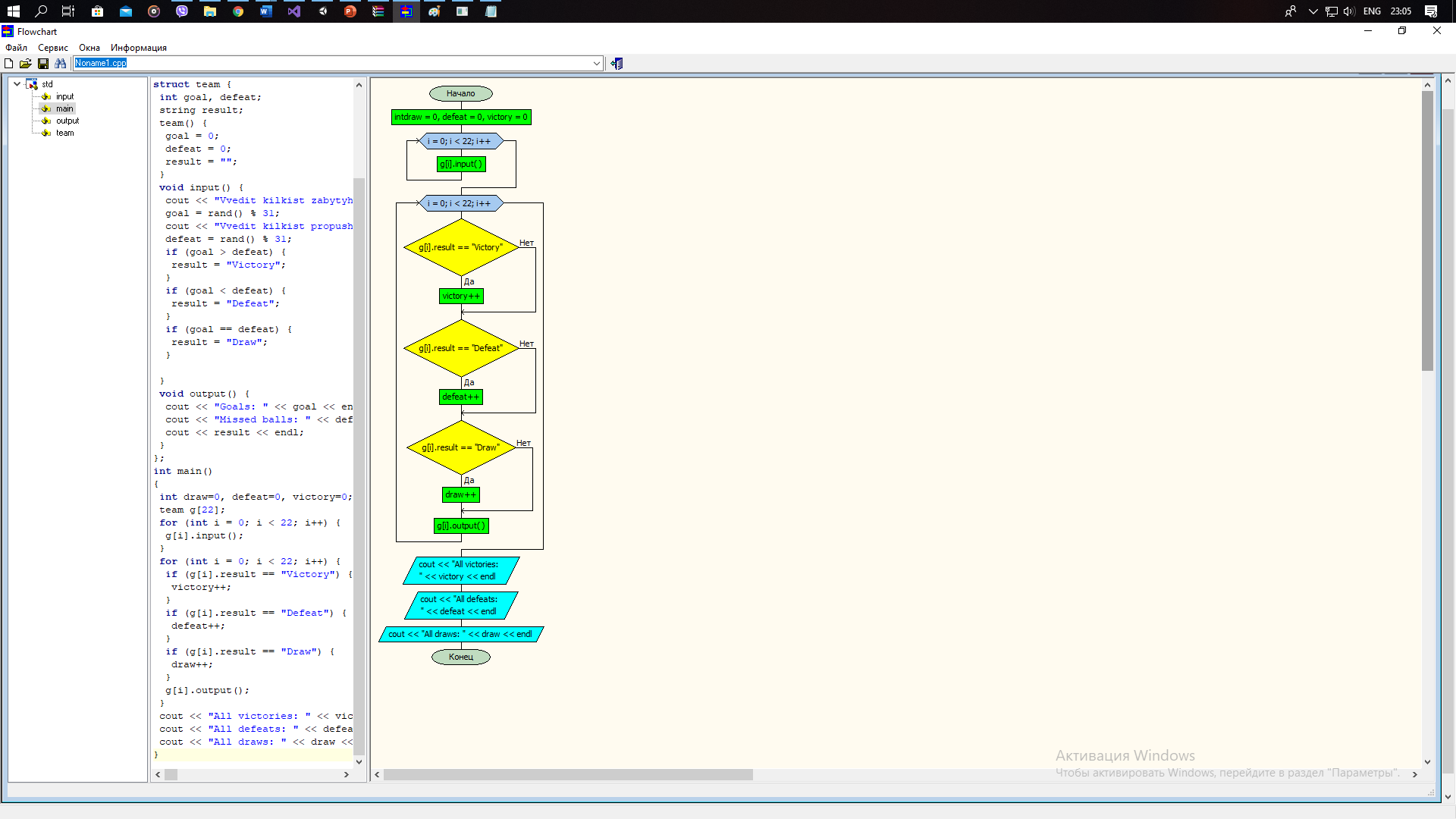
* 1. Для кожної проведеної гри надрукувати словесний результат: «виграш», «нічия» або «програш».
  2. Визначити кількість нічиїх даної команди.
  3. Визначити кількість виграшів і кількість програшів даної команди.
  4. Визначити кількість виграшів, кількість нічиїх і кількість програшів даної команди.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

Раніше для визначення класів ми використовували ключове слово class. Однак C ++ надає ще один спосіб для визначення користувача типів, який полягає у використанні структур. Даний спосіб був успадкований мовою С ++ ще від мови Сі.

Структура в мові C ++ є похідний тип даних, який представляє якусь певну сутність, також як і клас. Нерідко структури пріменітелько до С ++ також називають класами. І в реальності відмінності між ними не такі великі.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ



## Висновок

1. Під час виконання данного завданнь отримала навички з використання структур, стеків і черг.
2. **Додаток(а)**

#include "pch.h"

#include <iostream>

#include<cmath>

#include <string>

#include<sstream>

using namespace std;

struct team {

int goal, defeat;

string result;

team() {

goal = 0;

defeat = 0;

result = "";

}

void input() {

cout << "Vvedit kilkist zabytyh mjachiv: " << endl;

goal = rand() % 31;

cout << "Vvedit kilkist propushchenyh mjachiv: " << endl;

defeat = rand() % 31;

if (goal > defeat) {

result = "Victory";

}

if (goal < defeat) {

result = "Defeat";

}

if (goal == defeat) {

result = "Draw";

}

}

void output() {

cout << "Goals: " << goal << endl;

cout << "Missed balls: " << defeat << endl;

cout << result << endl;

}

};

int main()

{

int draw = 0, defeat = 0, victory = 0;

team g[22];

for (int i = 0; i < 22; i++) {

g[i].input();

}

for (int i = 0; i < 22; i++) {

if (g[i].result == "Victory") {

victory++;

}

if (g[i].result == "Defeat") {

defeat++;

}

if (g[i].result == "Draw") {

draw++;

}

g[i].output();

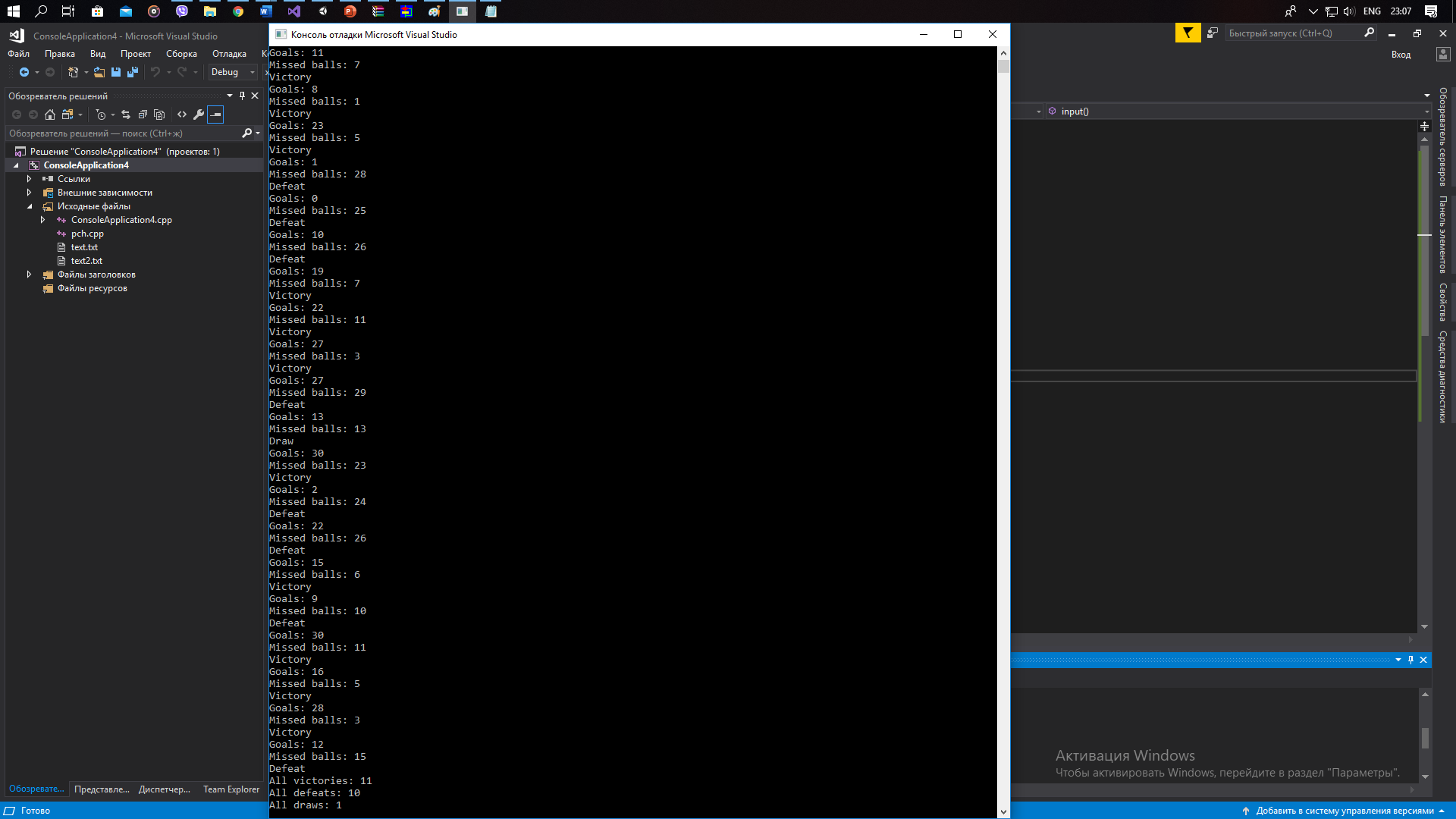
}

cout << "All victories: " << victory << endl;

cout << "All defeats: " << defeat << endl;

cout << "All draws: " << draw << endl;

}

1. **Додаток(Б)**
3. 

**Завдання 12**

**ЛАБОРАТОРНА РОБОТА №12. СТЕКИ ЧЕРГИ**

**Завдання:** *Реалізувати алгоритм та програму для розв’язання задачі відповідного варіанту*

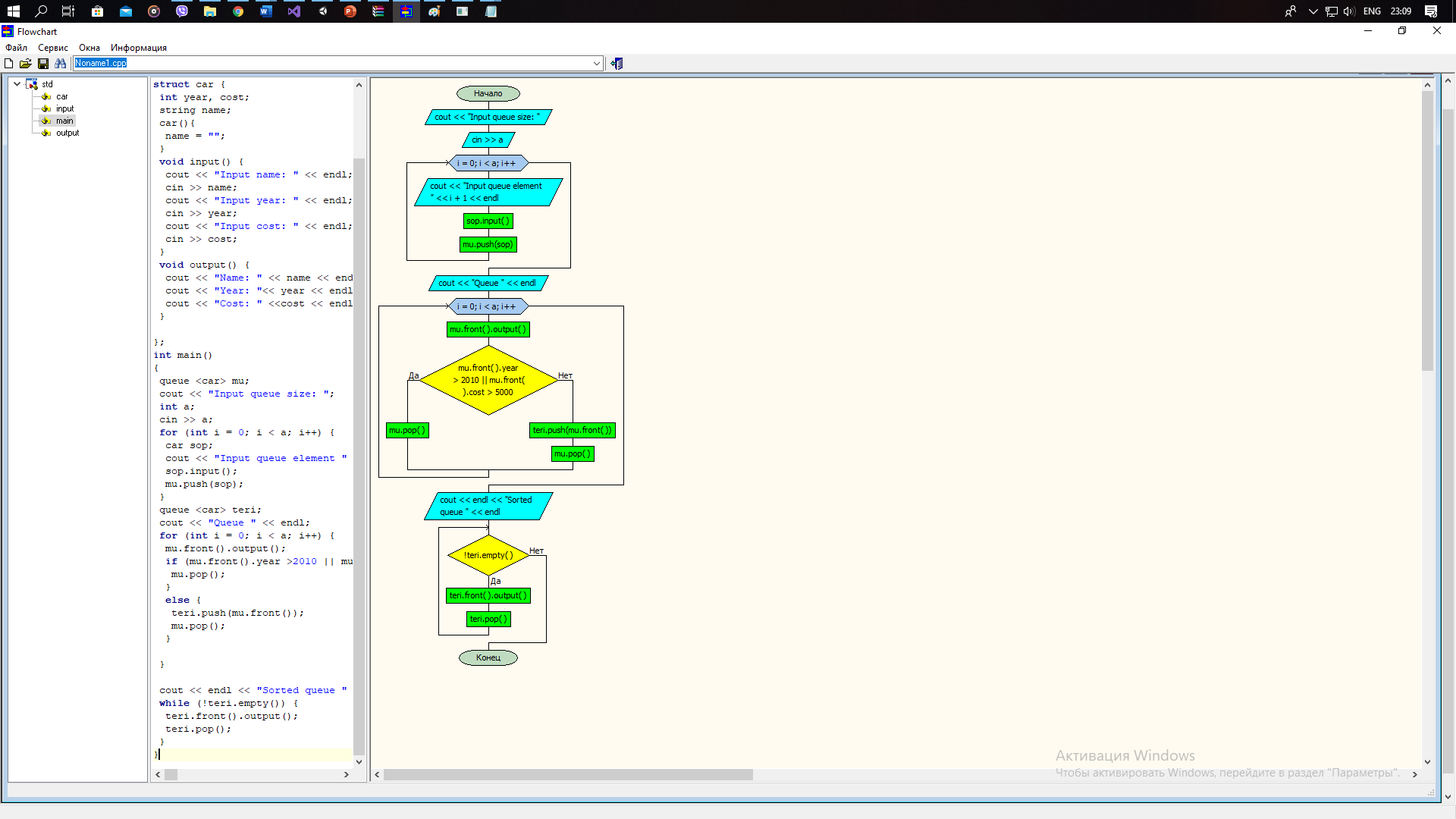
**Задачі для самостійної роботи.**

Створити чергу з n елементів, які мають три інформаційні поля: назва автомобіля, рік випуску та ціна. Вивести на екран перелік автомобілів, які старші 10 років та коштують менше 5000 у.о. Вивільнити динамічну пам’ять.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

1. ***Стек****— лінійний список, у якому всі операції вставки і знищення елементів виконуються лише на одному з кінців списку.****Черга****(одностороння черга) — лінійний список, у якому всі операції вставки здійснюються на одному з кінців списку, а всі операції знищення — на іншому.*

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ



## Висновок

1. Під час виконання данного завданнь отримала навички з використання стеків і черг.
2. **Додаток(а)**

#include <iostream>

#include <queue>

#include <string>

using namespace std;

struct car {

int year, cost;

string name;

car() {

name = "";

}

void input() {

cout << "Input name: " << endl;

cin >> name;

cout << "Input year: " << endl;

cin >> year;

cout << "Input cost: " << endl;

cin >> cost;

}

void output() {

cout << "Name: " << name << endl;

cout << "Year: " << year << endl;

cout << "Cost: " << cost << endl;

}

};

int main()

{

queue <car> mu;

cout << "Input queue size: ";

int a;

cin >> a;

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

car sop;

cout << "Input queue element " << i + 1 << endl;

sop.input();

mu.push(sop);

}

queue <car> teri;

cout << "Queue " << endl;

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

mu.front().output();

if (mu.front().year > 2010 || mu.front().cost > 5000) {

mu.pop();

}

else {

teri.push(mu.front());

mu.pop();

}

}

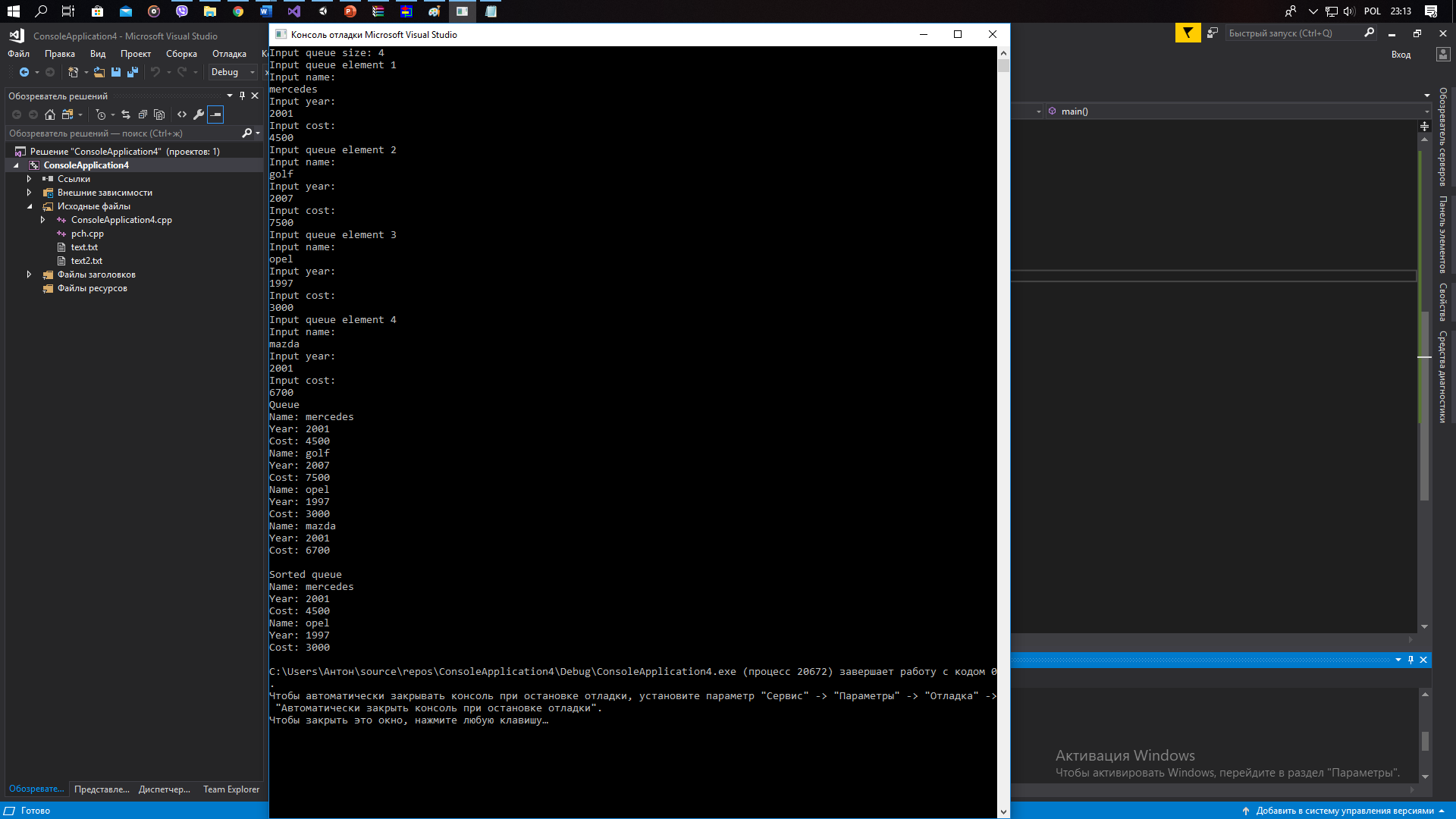
cout << endl << "Sorted queue " << endl;

while (!teri.empty()) {

teri.front().output();

teri.pop();

}

1. }
2. **Додаток(Б)**
3. 

# Завдання 13

1. **Лабораторна робота №13 робота з текстовими файлами**
2. **Завдання:** *Реалізувати алгоритм та програму для розв’язання задачі відповідного варіанту*
3. **Задачі для самостійної роботи.**

## На диску *D* створено символьний файл. Підрахувати кількість входжень в нього кожної з голосних букв.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

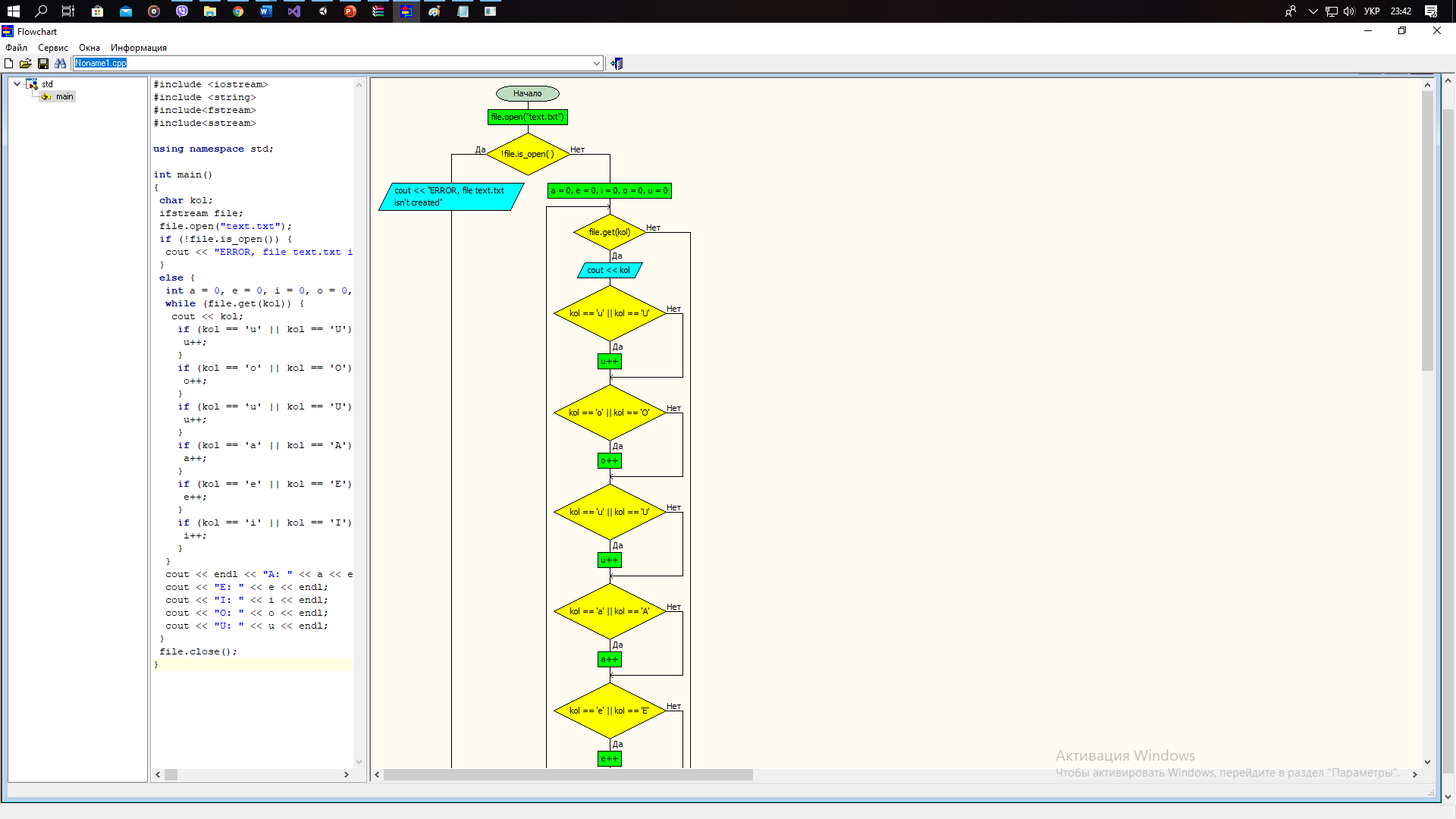
заголовок fstream надає функціонал для зчитування даних з файлу і для запису в файл. В цілому він дуже схожий на хедер iostream, який працює з консоллю, оскільки консоль це теж файл. Тому всі основні операції такі ж, за дрібними відмінностями, як в попередній темі по iostream.

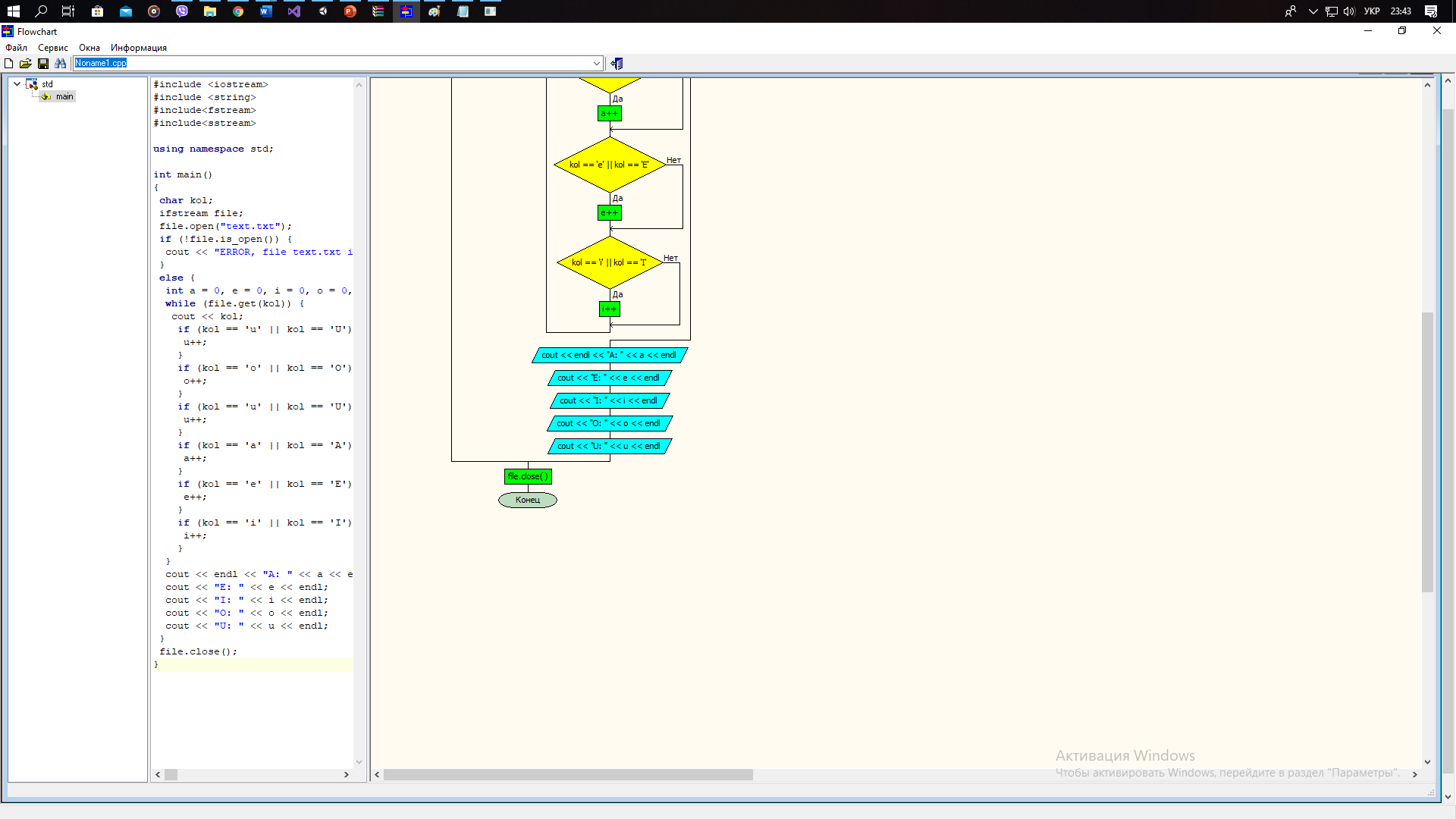
Це не всі можливості, які надає бібліотека fstream. Розглядати всі зараз ми не будемо, оскільки їх коло застосування досить вузький. Познайомимося з вищепереліченими. Почнемо з класу читання.

клас ifstream

Надає можливості для читання файлів. Відкрити файл можна двома способами: викликавши метод ВІДЧИНЕНО() або вказавши шлях до нього в конструкторі. Вам необхідно підготувати текстовий файл, перед тем, як почати набирати код. На диску d створіть папку з ім'ям 1  і в ній створіть файл з розширенням txt – “файл.txt”.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ ФУНКЦІЙ





## Висновок

1. Під час виконання данного завданнь отримала навички з роботи з текстовими файлами
2. **Додаток(а)**

#include <iostream>

#include <string>

#include<fstream>

#include<sstream>

using namespace std;

int main()

{

char kol;

ifstream file;

file.open("text.txt");

if (!file.is\_open()) {

cout << "ERROR, file text.txt isn't created";

}

else {

int a = 0, e = 0, i = 0, o = 0, u = 0;

while (file.get(kol)) {

cout << kol;

if (kol == 'u' || kol == 'U') {

u++;

}

if (kol == 'o' || kol == 'O') {

o++;

}

if (kol == 'u' || kol == 'U') {

u++;

}

if (kol == 'a' || kol == 'A') {

a++;

}

if (kol == 'e' || kol == 'E') {

e++;

}

if (kol == 'i' || kol == 'I') {

i++;

}

}

cout << endl << "A: " << a << endl;

cout << "E: " << e << endl;

cout << "I: " << i << endl;

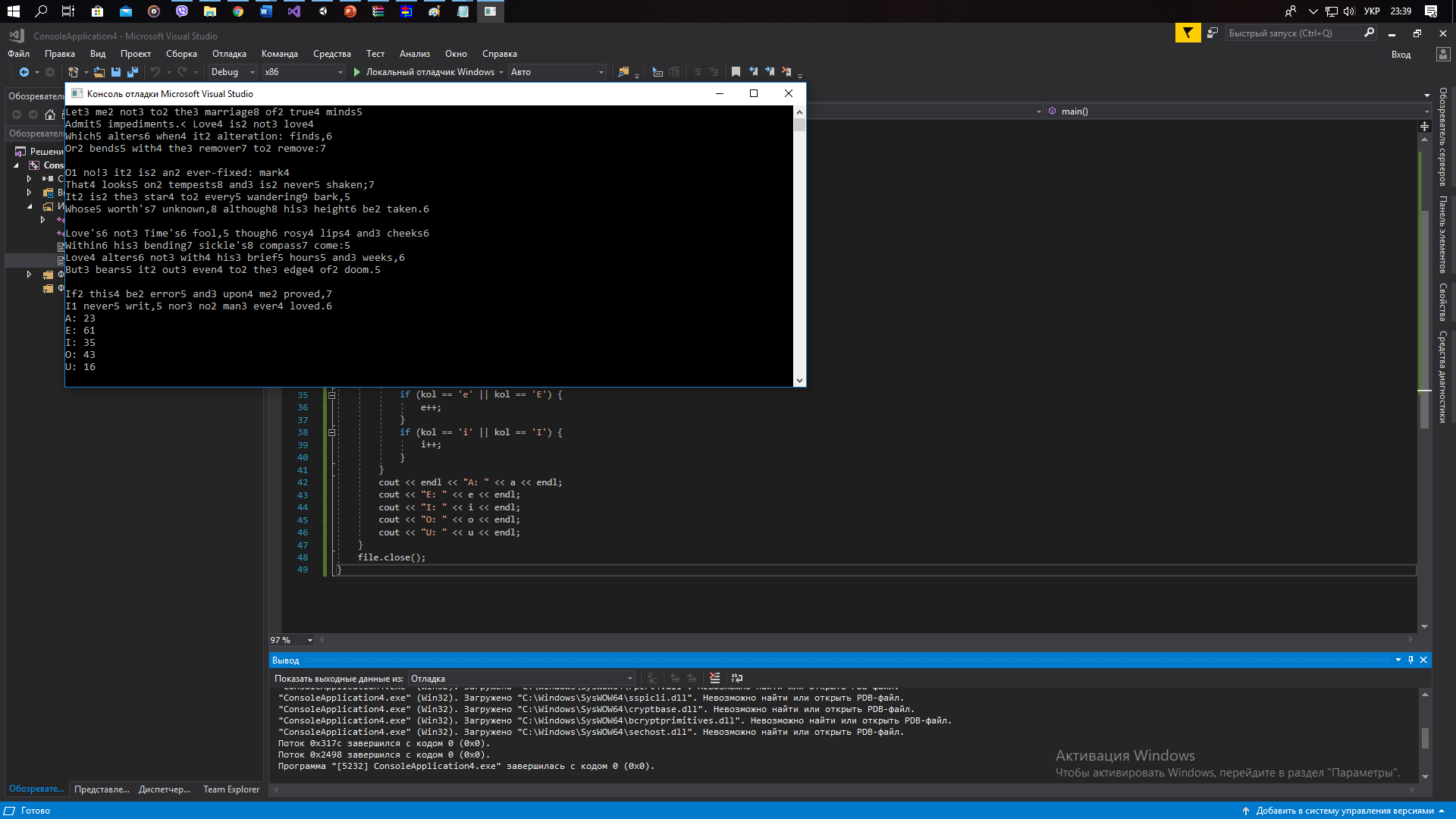
cout << "O: " << o << endl;

cout << "U: " << u << endl;

}

file.close();

}

1. **Додаток(б)**
2. 

**Завдання 14**

**ЛАБОРАТОРНА РОБОТА №14 РЯДКИ**

**Завдання:** *Реалізувати алгоритм та програму для розв’язання задачі відповідного варіанту*

**Задачі для самостійної роботи.**

1. Дано рядок *S*. В тексті замінити всі пробіли на символ "\_". Підрахувати кількість розділових знаків в тексті.

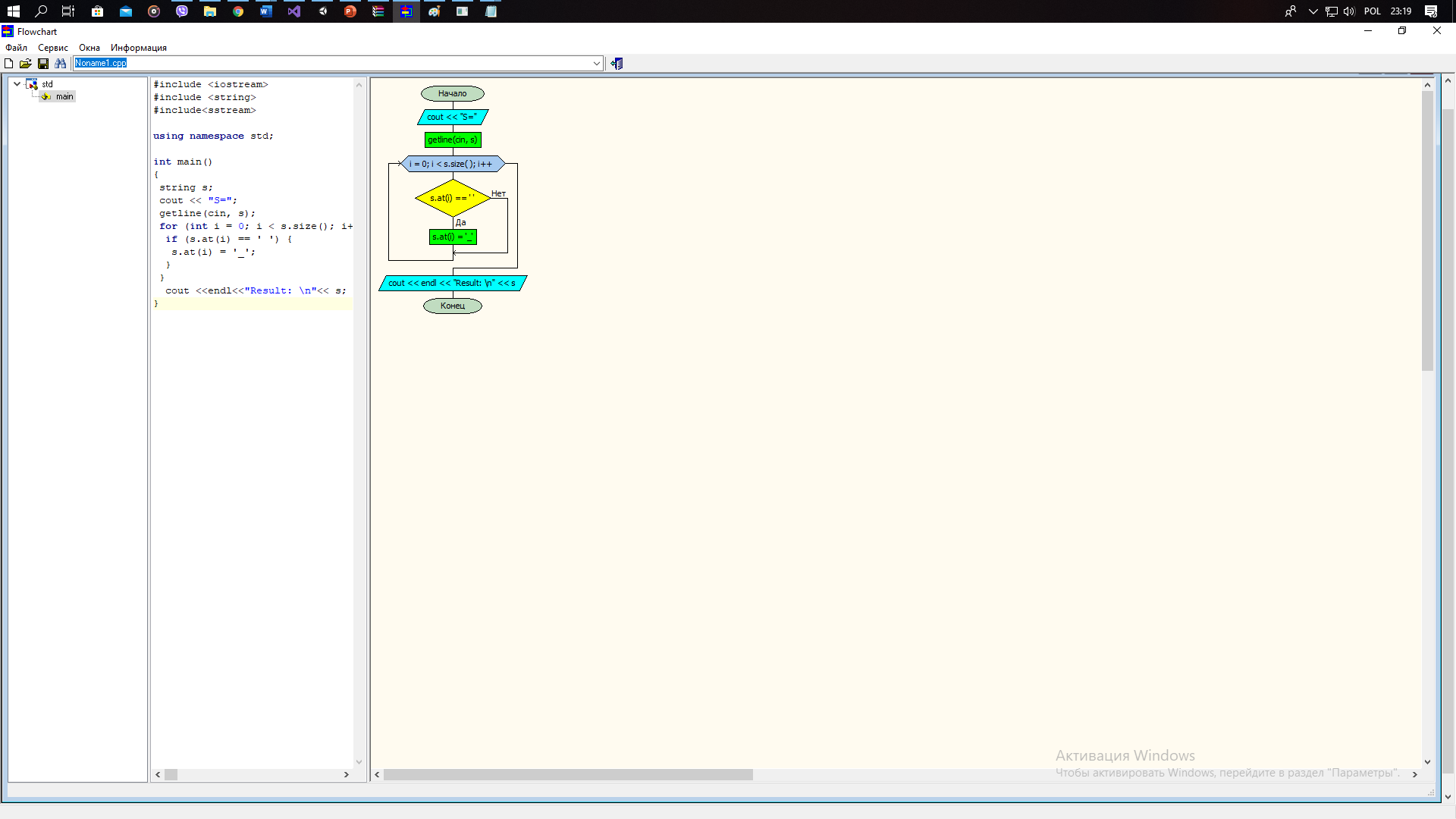
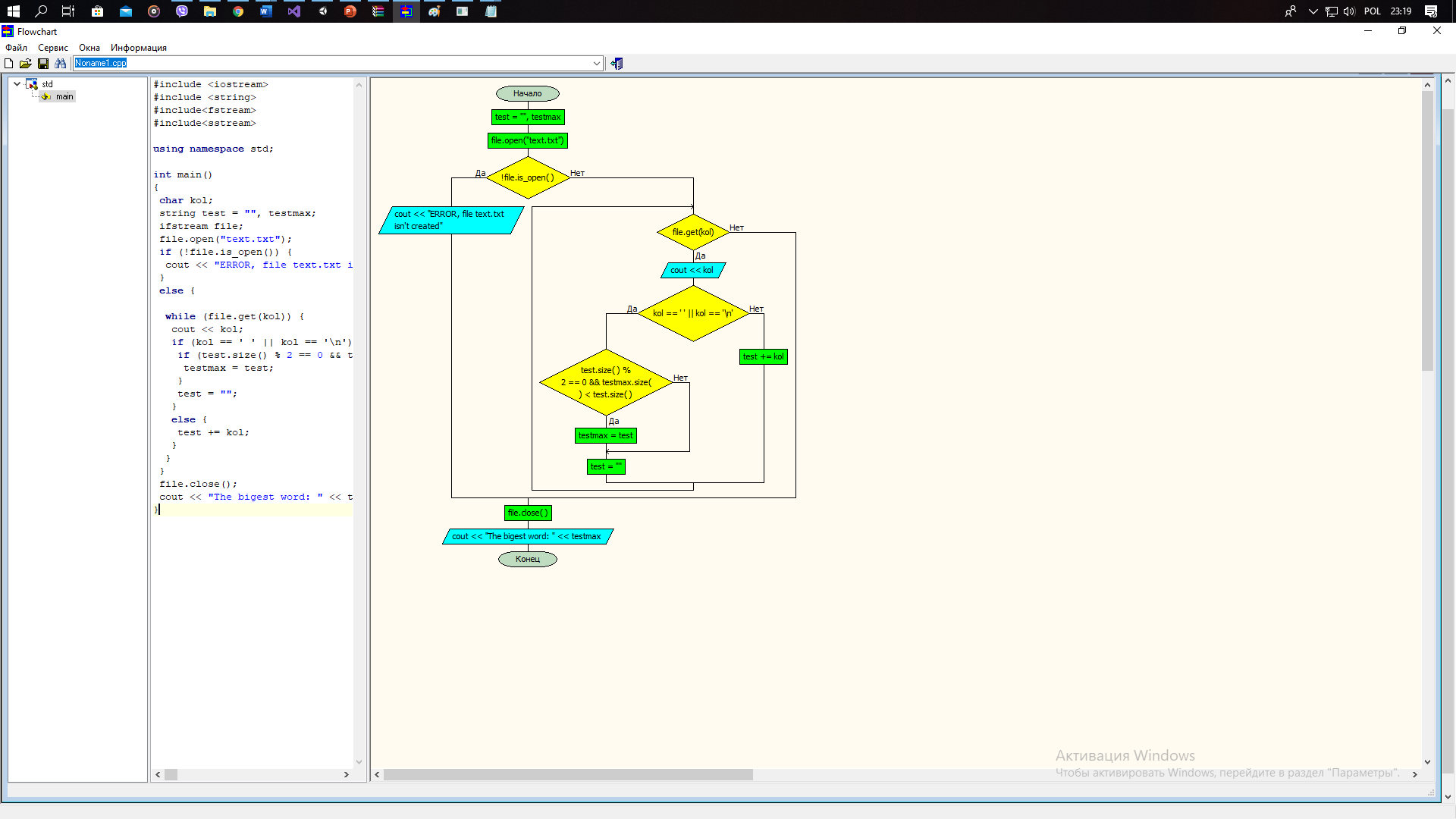
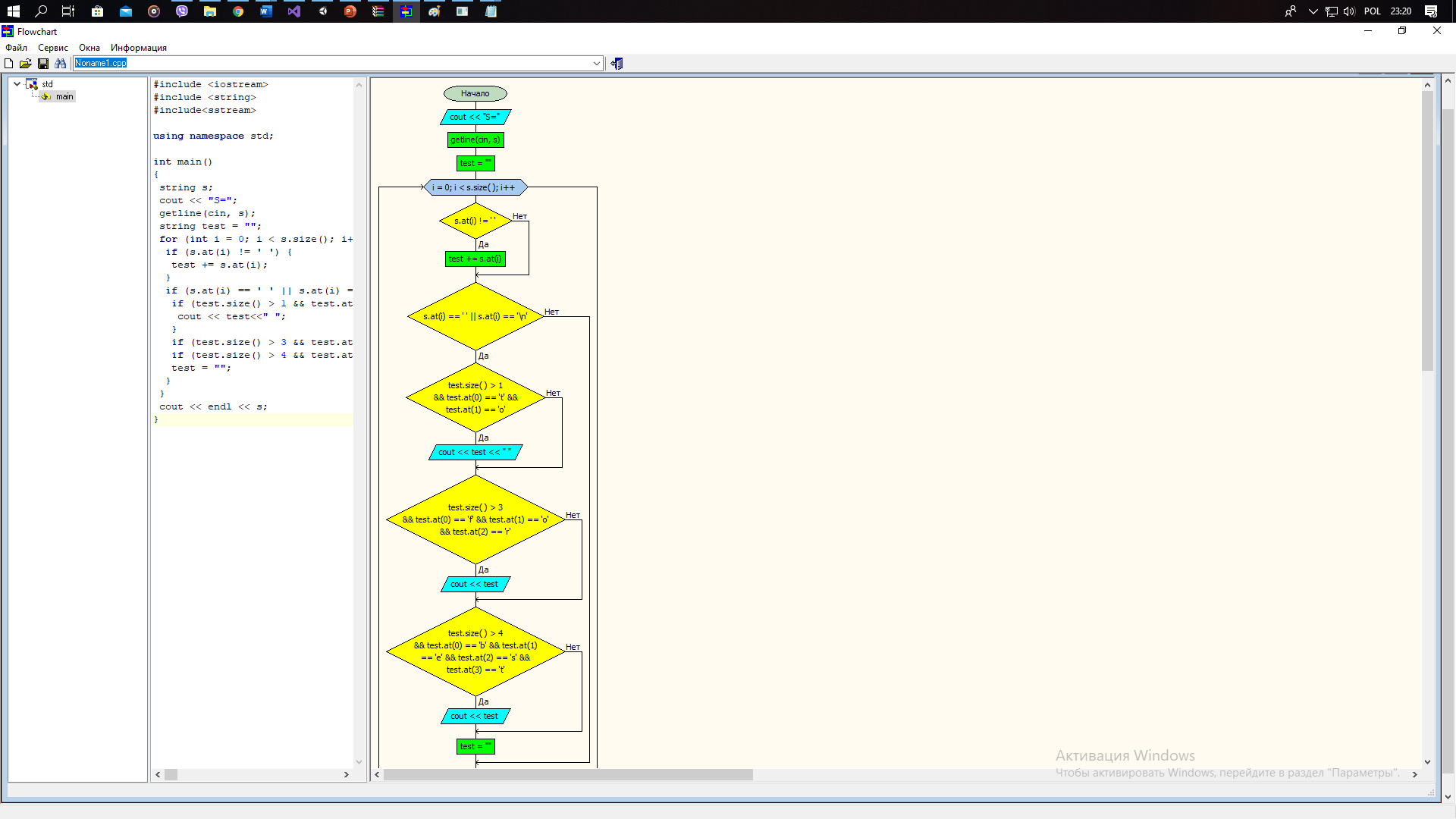
*Вхідні дані:* *S = "qw, e.!rty3 34sd?fg 3 we,3 4r"*.

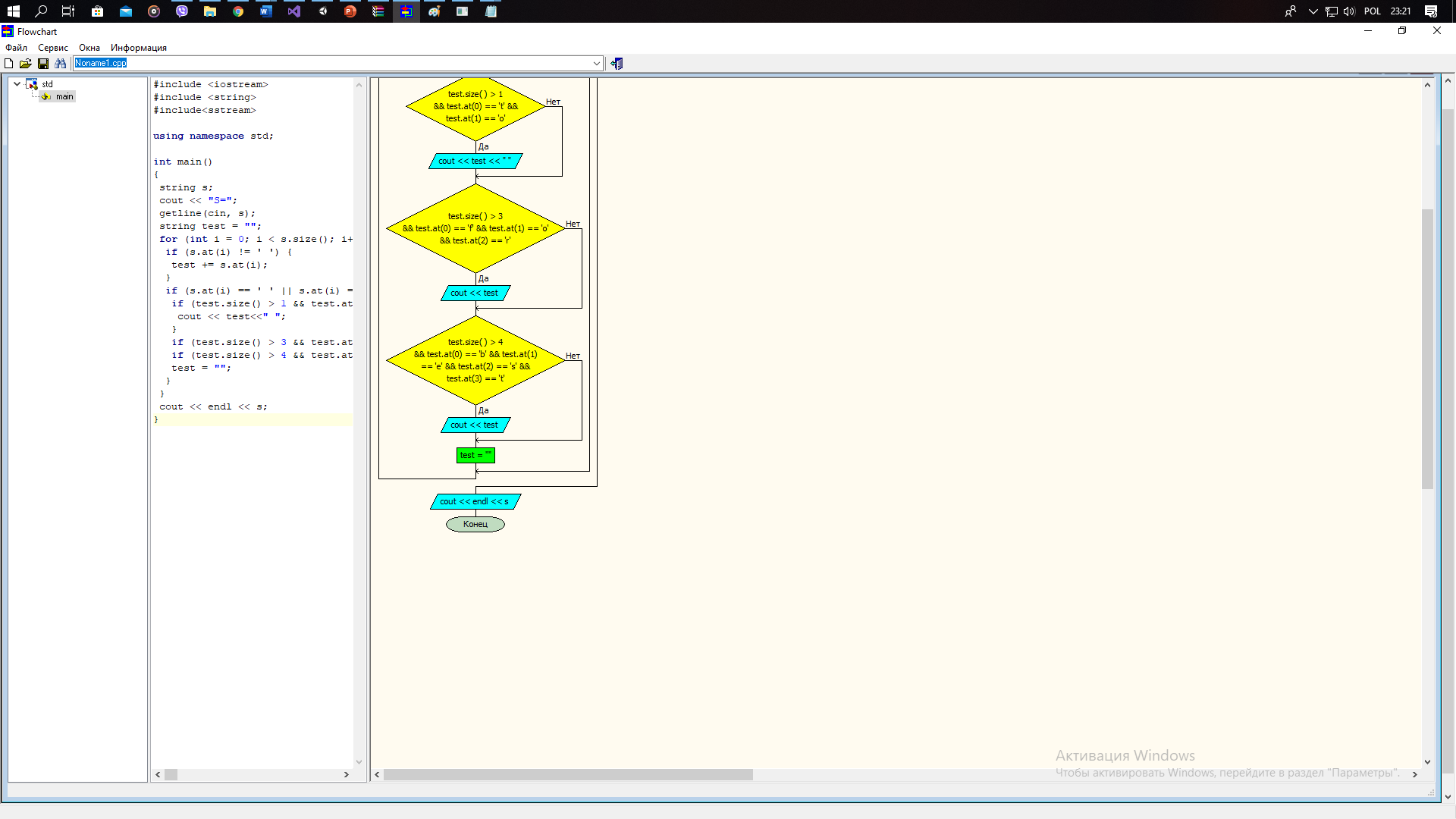
*Вихідні дані:* *S = "qw,\_e.!rty3\_34sd?fg\_3\_we,3\_4r"*. *Punctuation =* 5.

## ТЕОРЕТИЧНІ ВІДОМОСТІ

1. Рядки в С++ представляються як масиви елементів типу char, завершуються нуль-термінатором \0 називаються C рядками або рядками в стилі С.
2. \0 – символ нуль-термінатора.
3. Символьні рядки складаються з набору символьних констант поміщених в подвійні лапки. При оголошенні рядкового масиву необхідно враховувати наявність в кінці рядка нуль-термінатора, і відводити додатковий байт під нього.

## СХЕМИ АЛГОРИТМІВ ОСНОВНИХ





## Висновок

1. Під час виконання данного завданнь отримала навички з роботи з рядками
2. **Додаток(а)**

#include <iostream>

#include <string>

#include<sstream>

using namespace std;

int main()

{

string s;

cout << "S=";

getline(cin, s);

for (int i = 0; i < s.size(); i++) {

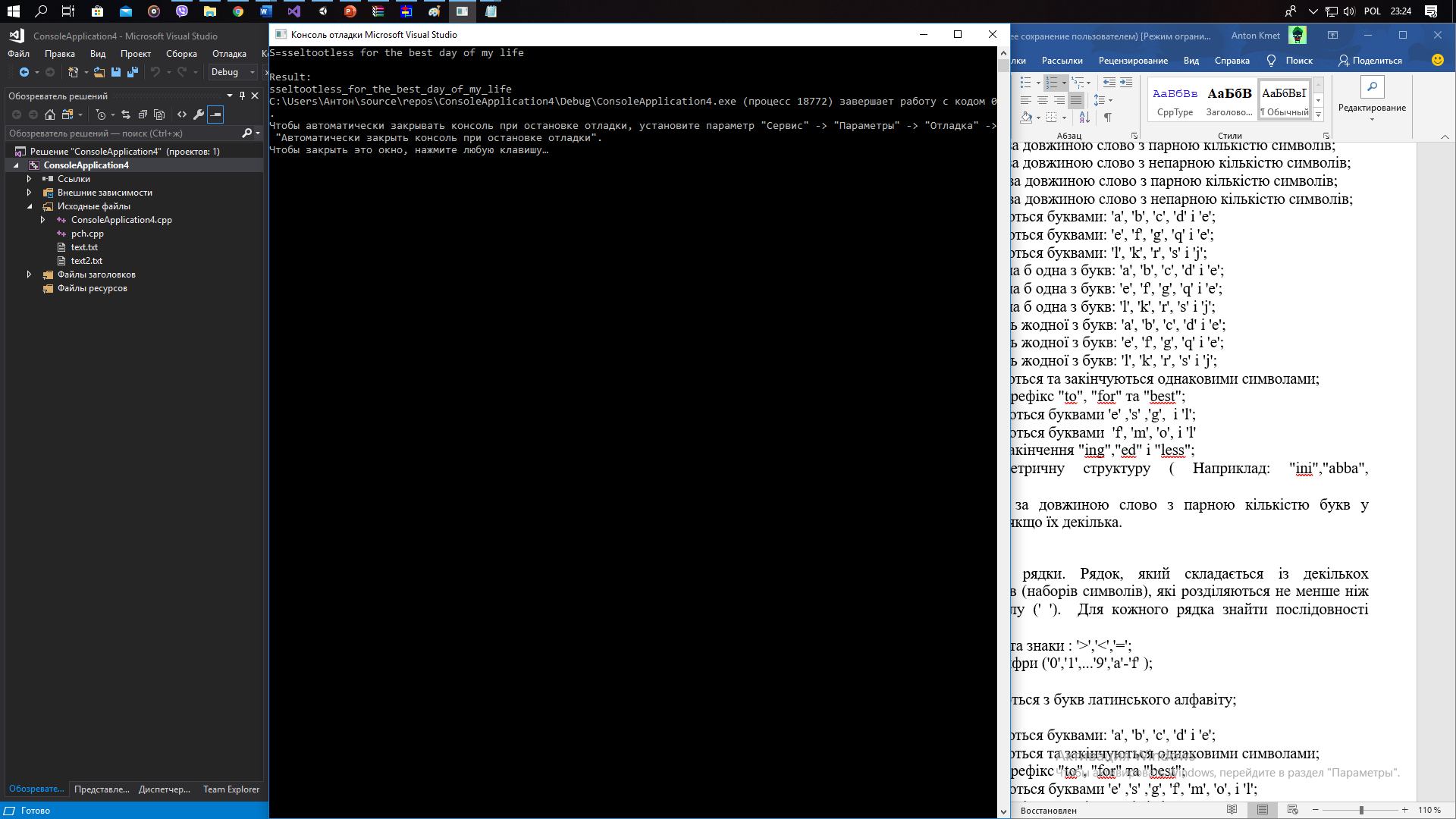
if (s.at(i) == ' ') {

s.at(i) = '\_';

}

}

cout << endl << "Result: \n" << s;

1. }
2. **Додаток(б)**
3. 

Завдання 2.

Розробити програми мовою С для розв’язання задачі згідно варіанту. Вхідні дані читаються з файлу, результат записується у новий файл. Ім’я файлів передаються через командний рядок, або вводяться з консолі.

* 1. перше найбільше за довжиною слово з парною кількістю символів;

1. Дано рядок *S*. Вивести всі слова, які містять рівно 3 літери «а».

*Вхідні дані:* *S = "qawartay daferf dasaffderda", S0 = “er”*.

*Вихідні дані:* *"qawartay dasaffderda".*

1. **Додаток(в)**

#include <iostream>

#include <string>

#include<fstream>

#include<sstream>

using namespace std;

int main()

{

char kol;

string test = "", testmax;

ifstream file;

file.open("text.txt");

if (!file.is\_open()) {

cout << "ERROR, file text.txt isn't created";

}

else {

while (file.get(kol)) {

cout << kol;

if (kol == ' ' || kol == '\n') {

if (test.size() % 2 == 0 && testmax.size() < test.size()) {

testmax = test;

}

test = "";

}

else {

test += kol;

}

}

}

file.close();

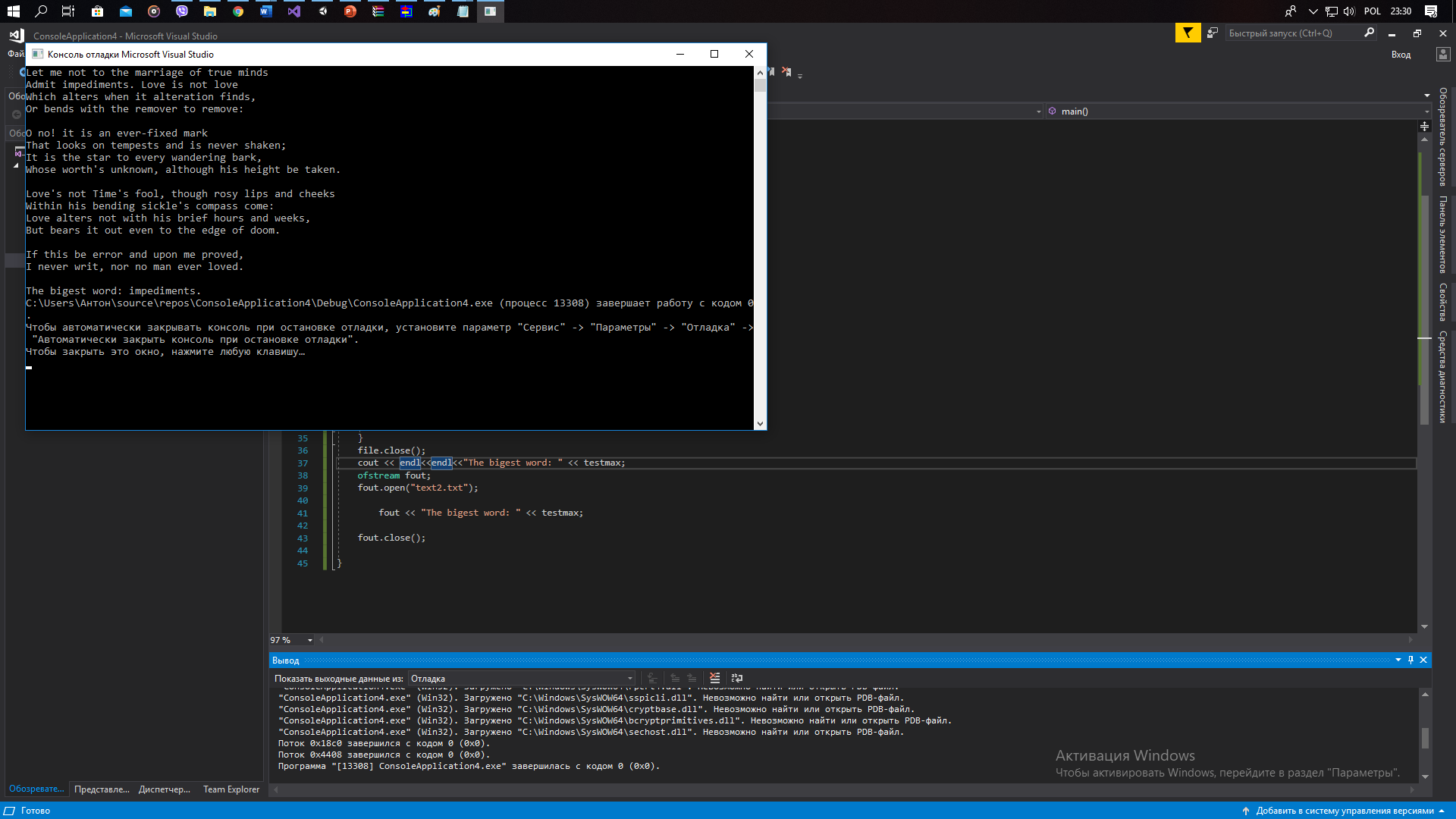
cout << endl<<endl<<"The bigest word: " << testmax;

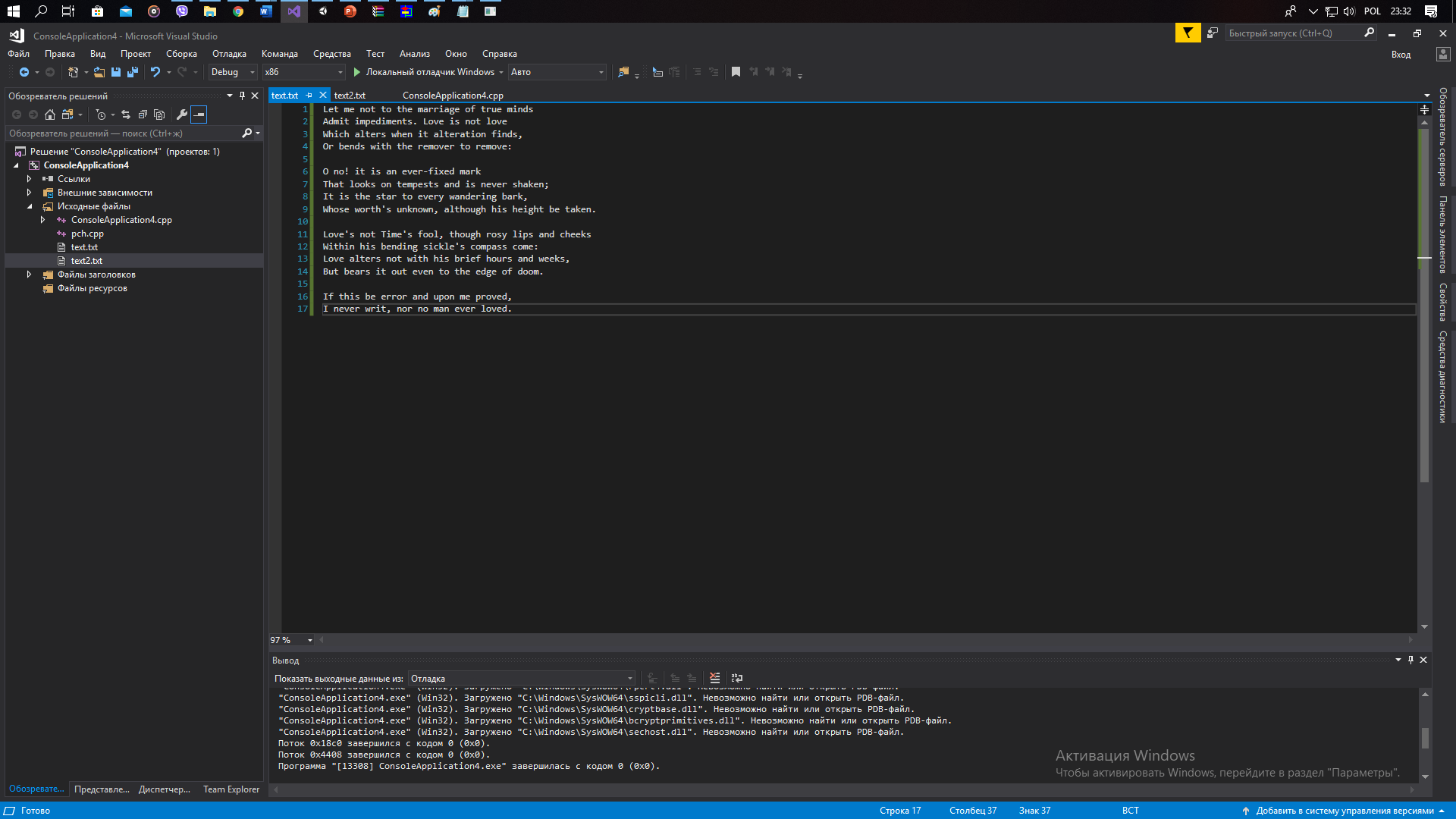
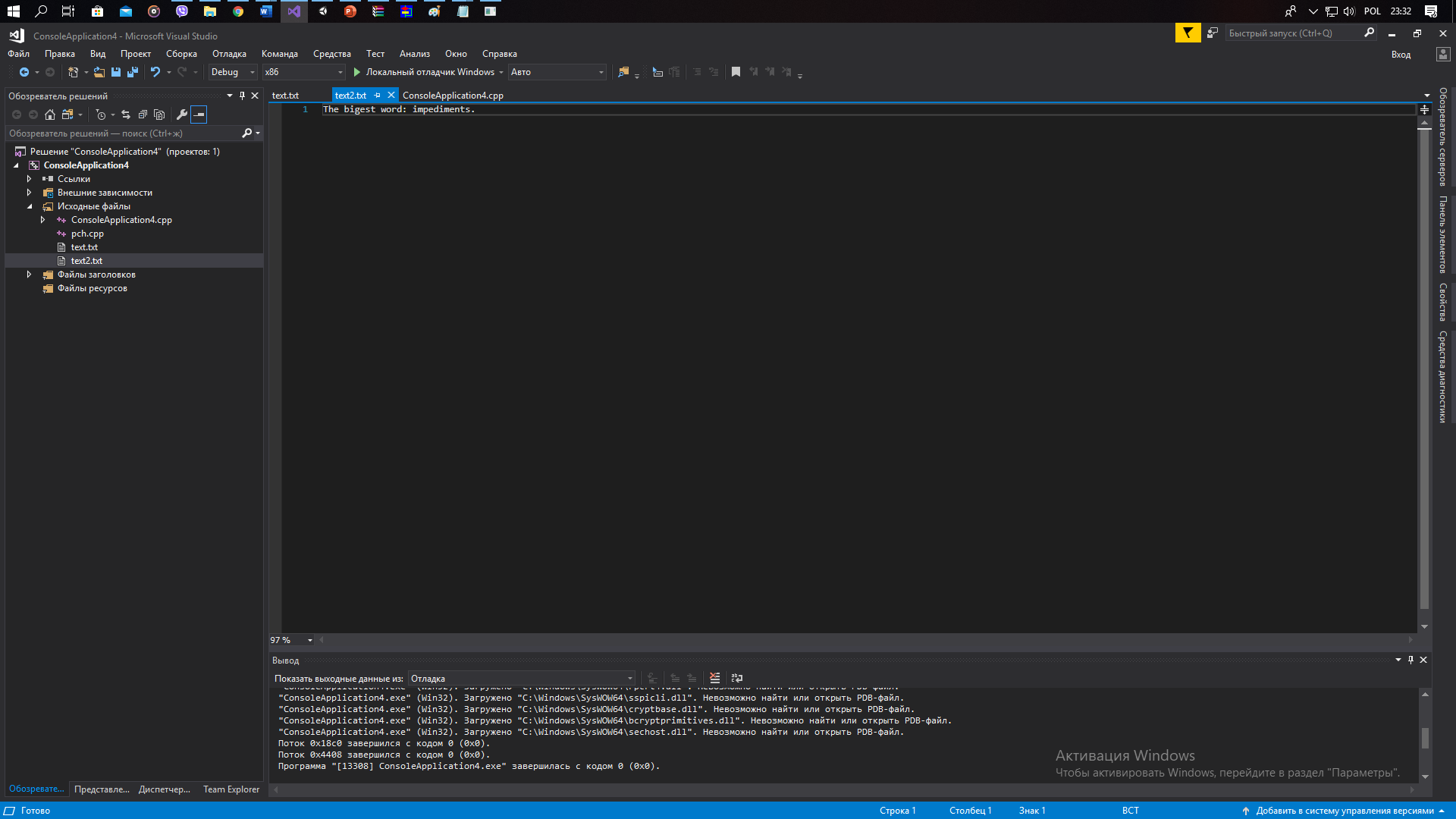
ofstream fout;

fout.open("text2.txt");

fout << "The bigest word: " << testmax;

fout.close();

1. }
2. **Додаток(г)**
3. 

Завдання 3.

Задано символьні рядки. Рядок, який складається із декількох послідовностей символів (наборів символів), які розділяються не менше ніж одним символом пробілу (' '). Для кожного рядка знайти послідовності символів у яких є:

слова, які мають префікс "to", "for" та "best";

**Додаток(д)**

#include <iostream>

#include <string>

#include<sstream>

using namespace std;

int main()

{

string s;

cout << "S=";

getline(cin, s);

string test = "";

for (int i = 0; i < s.size(); i++) {

if (s.at(i) != ' ') {

test += s.at(i);

}

if (s.at(i) == ' ' || s.at(i) == '\n') {

if (test.size() > 1 && test.at(0) == 't' && test.at(1) == 'o') {

cout << test << " ";

}

if (test.size() > 3 && test.at(0) == 'f' && test.at(1) == 'o' && test.at(2) == 'r') { cout << test; }

if (test.size() > 4 && test.at(0) == 'b' && test.at(1) == 'e' && test.at(2) == 's' && test.at(3) == 't') { cout << test; }

test = "";

}

}

cout << endl << s;

}**Додаток(е)**

