**КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ  
ІМЕНІ ТАРАСА ШЕВЧЕНКА**

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**Кафедра інтелектуальних технологій**

Лабораторна робота №8  
*(вид роботи: лабораторна робота, індивідуальне завдання, курсова робота тощо)*

з дисципліни «Алгоритмізація та програмування»Тема роботи: «Динамічне розподілення пам’яті»  
**Варіант № 4**

Виконав(-ла) студент(-ка)  
групи АнД - 11  
Яковкін Микола Андрійович

Перевірив(-ла):  
ПІП викладача

Київ – 2022

**Завдання 1**

**1.1 Математична постановка задачі (МПЗ).**

*Вхідні дані:*

size – к-сть елементів масиву, цілого типу.

value – елементи масиву, дійсного типу.

*Вихідні дані:*

value – дробові частини елементів масиву, дійсного типу.

product – добуток всіх дробових частин, дійсного типу.

*Математична модель задачі:*

size = |size|

Для i = 0, size – 1:

value = |value|

Ai = value

value = Ai – floor(Ai)

product = product \* value

Подання математичної постановки задачі у вигляді таблиці:

Таблиця 1.1

|  |  |  |
| --- | --- | --- |
| **Вхідні дані** | **Дії** | **Вихідні дані** |
| size – к-сть елементів масиву, цілого типу.  value – елементи масиву, дійсного типу. | size = |size|  Для i = 0, size – 1:  value = |value|  Ai = value  value = Ai – floor(Ai)  product = product \* value | value – дробові частини елементів масиву, дійсного типу.  product – добуток всіх дробових частин, дійсного типу. |

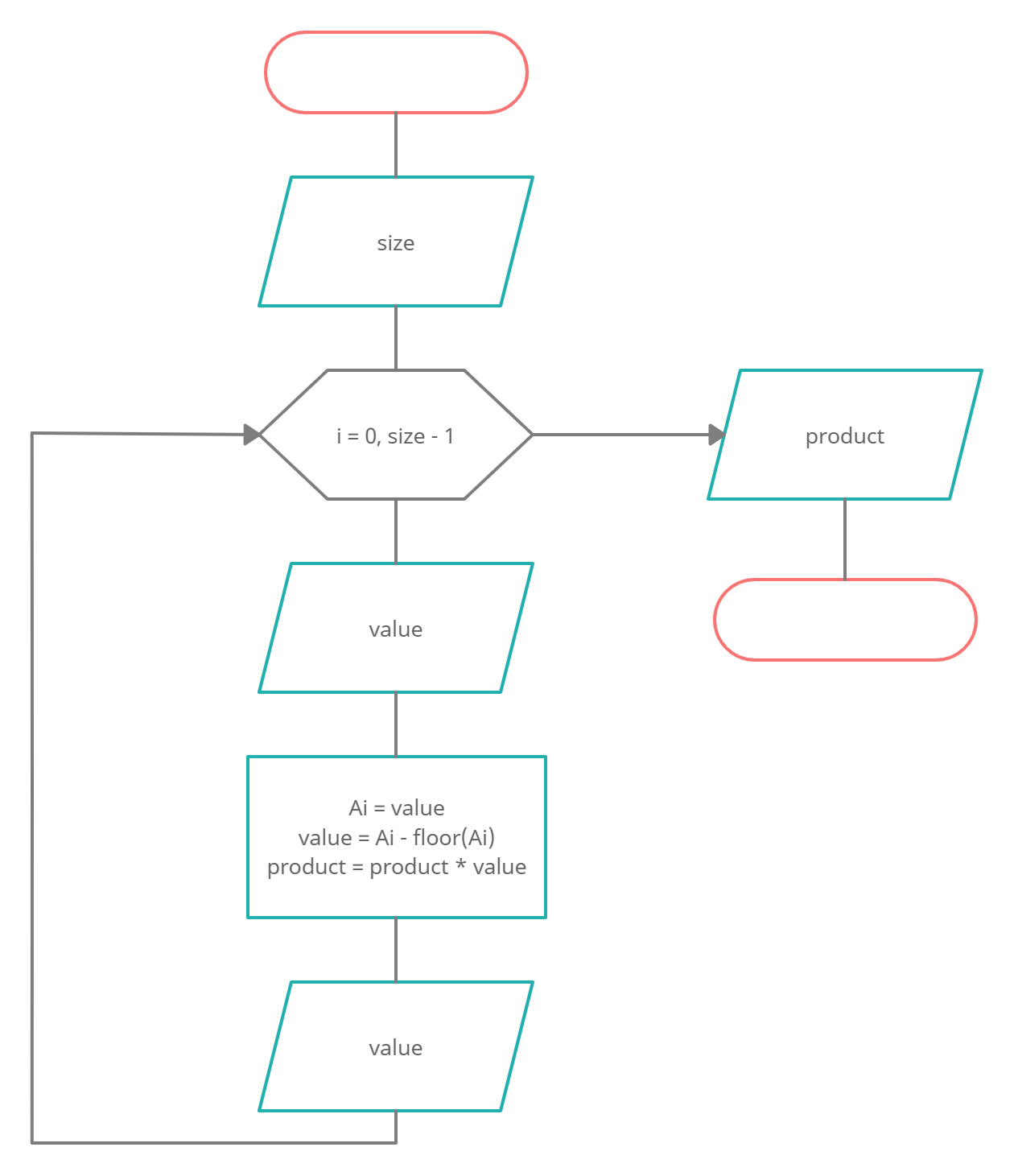


Рисунок 1.1 – Схема алгоритму для задачі 1

**1.2 Тестування програми**

Для доведення працездатності роботи програми проведемо таку перевірку:

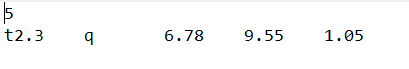


Рисунок 1.2 – Вхідні дані для тестування.

Маємо отримати:

0.78, 0.55, 0.05.

product = 0.78 \* 0.55 \* 0.05 = 0,02145

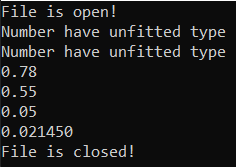


Рисунок 1.3 – Тестування програми.

**1.3 Текст програмною мовою С++.**

#**include** <iostream>

#**include** <string>

#**include** <fstream>

**using namespace** std;

**void** main()

{

**string** path = "valuesEx1.txt";

**double** product = 1;

**double** \*pProduct = &product;

**ifstream** fin;

fin.open(path);

**char** ch;

**if** (!fin.is\_open())

{

cout << "Error! File can`t be open. Try again" << endl;

}

**else**

{

cout << "File is open!" << endl;

**int** size;

fin >> size;

**double**\* array = **new** **double**[size];

//cout << size;

for (int i = 0; i < size; i++)

{

**double** value;

fin >> value;

**if** (fin.fail())

{

cerr << "Number have unfitted type" << endl;

fin.clear();

fin.ignore(32767, '\t');

**continue;**

}

**else**

{

array[i] = value;

value = array[i] - floor(array[i]);

cout << value << endl;

\*pProduct \*= value;

}

//cout << array[i] << endl;

}

cout << fixed << product << endl;

**delete**[] array;

fin.**close**();

cout << "File is closed!" << endl;

}

}

**Завдання 2**

**2.1 Математична постановка задачі (МПЗ).**

*Вхідні дані:*

value – елемент цілого типу.

K – число цілого типу.

*Вихідні дані:*

value – елемент цілого типу.

і – порядковий номер елемента цілого типу

*Математична модель задачі:*

K = |K|

Доки не кінець файлу:

temp\_value = |temp\_value|

temp\_i = temp\_i + 1

Якщо temp\_value > K, то:

value = temp\_value

i = temp\_i

Якщо i ≠ 0, то:

Виведення: і та value

Інакше:

Виведення: 0

Подання математичної постановки задачі у вигляді таблиці:

Таблиця 2.1

|  |  |  |
| --- | --- | --- |
| **Вхідні дані** | **Дії** | **Вихідні дані** |
| value – елемент цілого типу.  K – число цілого типу. | K = |K|  Доки не кінець файлу:  temp\_value = |temp\_value|  temp\_i = temp\_i + 1  Якщо temp\_value > K, то:  value = temp\_value  i = temp\_i  Якщо i ≠ 0, то:  Виведення: і та value  Інакше:  Виведення: 0 | value – елемент цілого типу.  і – порядковий номер елемента цілого типу |

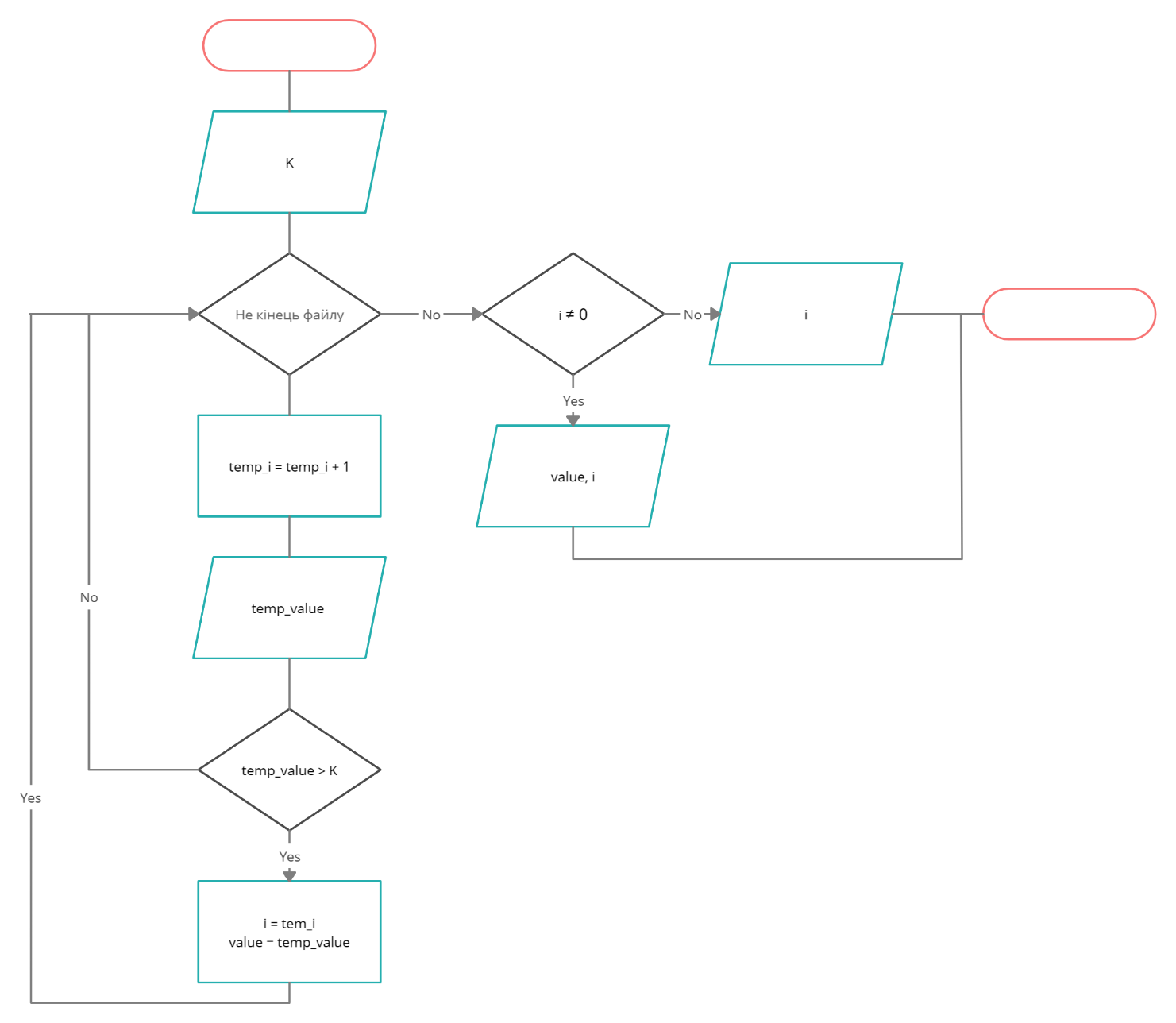


Рисунок 2.1 – Схема алгоритму для задачі 1

**2.2 Тестування програми**

Для доведення працездатності роботи програми проведемо таку перевірку:



Рисунок 2.2 – Вхідні дані для тестування.

Маємо отримати:

value = 13 та i =10

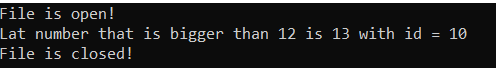


Рисунок 2.3 – Тестування програми.

**2.3 Текст програмною мовою С++.**

#**include** <iostream>

#**include** <fstream>

#**include** <string>

**using namespace** std;

**int** main()

{

**string** path = "valuesEx2.txt";

**ifstream** fin;

fin.open(path);

**if** (!fin.is\_open())

{

cout << "Error! File can`t be open. Try again" << endl;

**return** -1;

}

**else**

{

cout << "File is open!" << endl;

**int** i = 0, temp\_i = 0;

**int** temp\_value, value;

**int** K;

fin >> K;

**while** (!fin.eof())

{

fin >> temp\_value;

temp\_i++;

**if** (temp\_value > K)

{

i = temp\_i;

value = temp\_value;

}

}

**if** (i != 0) cout << "Lat number that is bigger than " << K << " is " << value << " with id = " << i << endl;

**else** cout << 0 << endl;

fin.close();

cout << "File is closed!" << endl;

}

**return** 0;

}

**Завдання 3**

**3.1 Математична постановка задачі (МПЗ).**

*Вхідні дані:*

rows – к-сть рядків у масиві, цілого типу.

cols – к-сть стовпчиків у масиві, цілого типу.

Aij – елементи масиву, цілого типу.

*Вихідні дані:*

count – к-сть двійок у масиві, цілого типу.

*Математична модель задачі:*

Для i = 0, rows – 1:

Для j = 0, cols – 1:

Aij = |Aij|

Для i = 0, rows – 1:

Для j = 0, cols – 1:

Якщо Aij = 2, то:

count = count + 1

Подання математичної постановки задачі у вигляді таблиці:

Таблиця 3.1

|  |  |  |
| --- | --- | --- |
| **Вхідні дані** | **Дії** | **Вихідні дані** |
| rows – к-сть рядків у масиві, цілого типу.  cols – к-сть стовпчиків у масиві, цілого типу.  Aij – елементи масиву, цілого типу. | Для i = 0, rows – 1:  Для j = 0, cols – 1:  Aij = |Aij|  Для i = 0, rows – 1:  Для j = 0, cols – 1:  Якщо Aij = 2, то:  count = count + 1 | count – к-сть двійок у масиві, цілого типу. |

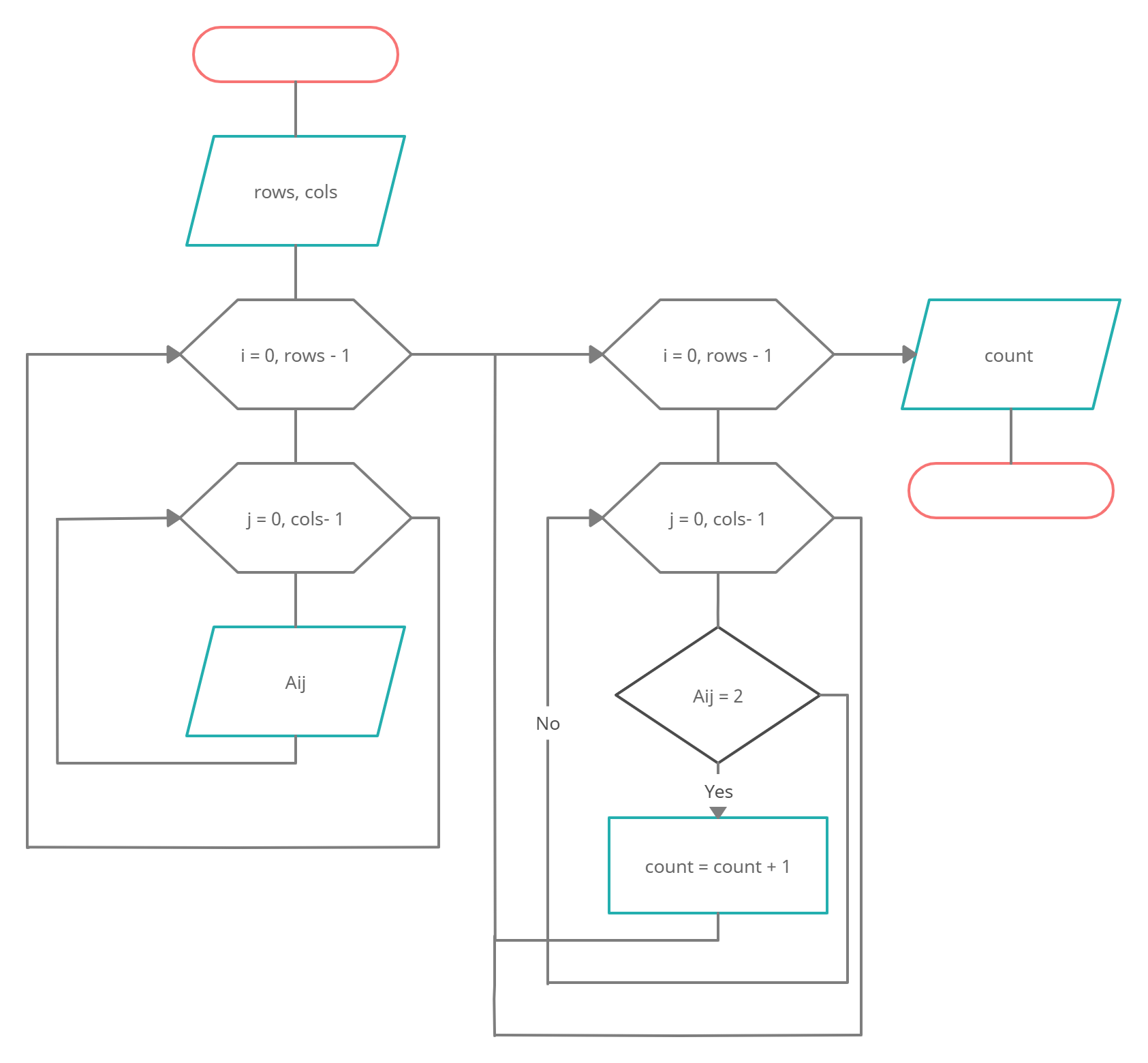


Рисунок 3.1 – Схема алгоритму для задачі 1

**3.2 Тестування програми**

Для доведення працездатності роботи програми проведемо таку перевірку:

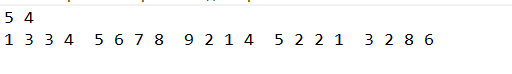


Рисунок 3.2 – Вхідні дані для тестування.

Маємо отримати:

count = 3

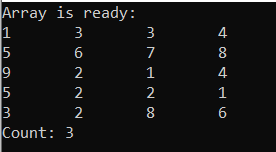


Рисунок 3.3 – Тестування програми.

**3.3 Текст програмною мовою С++.**

#**include** <iostream>

#**include** <fstream>

#**include** <string>

**using namespace** std;

**int** main()

{

**string** path = "valuesEx3.txt";

**ifstream** fin;

fin.open(path);

**if** (!fin.is\_open())

{

cout << "Error! File can`t be open. Try again" << endl;

**return** -1;

}

**else**

{

**int** count = 0;

**int** rows, cols;

fin >> rows >> cols;

**int**\*\* array = **new int**\* [rows];

**for** (**int** i = 0; i < rows; i++)

{

array[i] = **new int**[cols];

}

cout << "Array is ready: " << endl;

**for** (**int** i = 0; i < rows; i++)

{

**for** (**int** j = 0; j < cols; j++)

{

fin >> array[i][j];

cout << array[i][j] << '\t';

}

cout << endl;

}

**for** (**int** i = 0; i < rows; i++)

{

**for** (**int** j = 0; j < cols; j++)

{

**if** (array[i][j] == 2)

{

count++;

**break**;

}

}

}

cout << "Count: " << count << endl;

**for** (**int** i = 0; i < rows; i++)

{

**delete**[] array[i];

}

**delete**[] array;

fin.**close**();

}

**return** 0;

}

**Завдання 4**

**4.1 Математична постановка задачі (МПЗ).**

*Вхідні дані:*

K – к-сть наборів, цілого типу.

fiV, seV, thiV – елементи наборів, цілого типу.

*Вихідні дані:*

count – к-сть пилкоподібних наборів, цілого типу.

*Математична модель задачі:*

K = |K|

Для i = 0, K – 1:

waste = 1

fiV = |fiV|

seV = |seV|

thiV = |thiV|

Доки thiV ≠ 0:

Якщо fiV < seV та thiV < seV або seV < fiV та seV < thiV, то:

fiV = seV

seV = thiV

thiV = |thiV|

Інакше:

waste = 0

Доки thiV ≠ 0:

thiV = |thiV|

Якщо waste = 1, то:

count = count + 1

Подання математичної постановки задачі у вигляді таблиці:

Таблиця 4.1

|  |  |  |
| --- | --- | --- |
| **Вхідні дані** | **Дії** | **Вихідні дані** |
| K – к-сть наборів, цілого типу.  fiV, seV, thiV – елементи наборів, цілого типу. | K = |K|  Для i = 0, K – 1:  waste = 1  fiV = |fiV|  seV = |seV|  thiV = |thiV|  Доки thiV ≠ 0:  Якщо fiV < seV та thiV < seV або seV < fiV та seV < thiV, то:  fiV = seV  seV = thiV  thiV = |thiV|  Інакше:  waste = 0  Доки thiV ≠ 0:  thiV = |thiV|  Якщо waste = 1, то:  count = count + 1 | count – к-сть пилкоподібних наборів, цілого типу. |

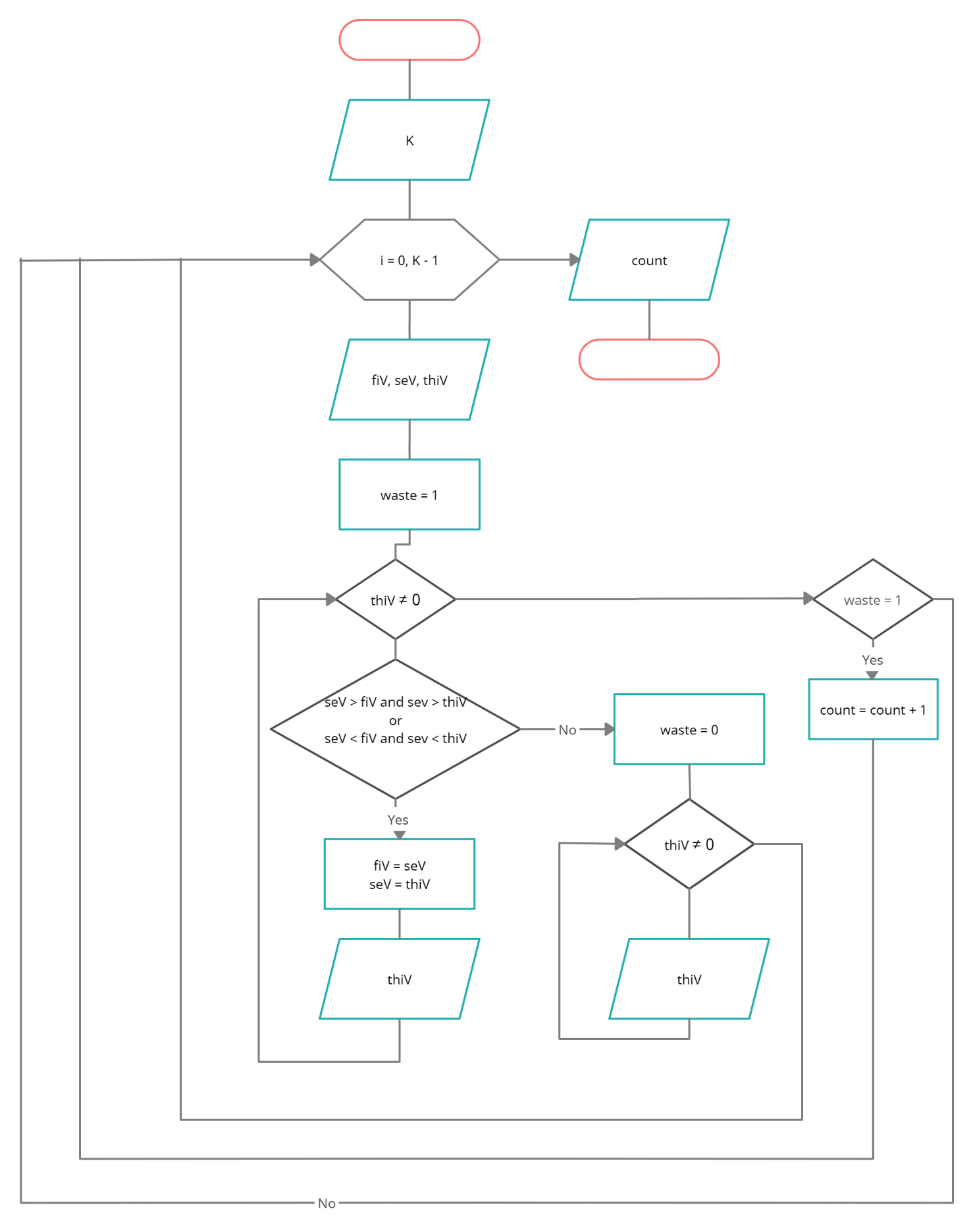


Рисунок 4.1 – Схема алгоритму для задачі 1

**4.2 Тестування програми**

Для доведення працездатності роботи програми проведемо таку перевірку:



Рисунок 4.2 – Вхідні дані для тестування.

Маємо отримати:

count = 3

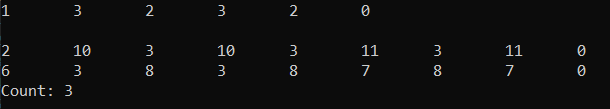


Рисунок 4.3 – Тестування програми.

**4.3 Текст програмною мовою С++.**

#**include** <iostream>

#**include** <string>

#**include** <fstream>

**using namespace** std;

**void** main()

{

**string** path = "valuesEx4.txt";

**ifstream** fin;

fin.open(path);

**if** (!fin.is\_open())

{

cout << "Error! File can`t be open." << endl;

}

**else**

{

**int** count = 0;

**int** waste = 1;

**int** K;

fin >> K;

//cout << K << endl;

**int** fiV, seV, thiV;

**for** (**int** i = 0; i < K; i++)

{

fin >> fiV >> seV >> thiV;

waste = 1;

**while** (thiV != 0)

{

//cout << fiV << '\t' << seV << '\t' << thiV << '\t';

**if** ((fiV < seV && thiV < seV) || (seV < fiV && seV < thiV))

{

fiV = seV;

seV = thiV;

fin >> thiV;

cout << fiV << '\t' << seV << '\t' << thiV << '\t';

}

**else**

{

waste = 0;

while (thiV != 0) fin >> thiV;

//cout << thiV << endl;

}

}

cout << endl;

**if** (waste) count++;

}

cout << "Count: " << count << endl;

fin.**close**();

}

}

**Завдання 4 modified**

**5.1 Математична постановка задачі (МПЗ).**

*Вхідні дані:*

fiV, seV, thiV – елементи наборів, цілого типу.

*Вихідні дані:*

count – к-сть пилкоподібних наборів, цілого типу.

*Математична модель задачі:*

Доки не кінець файлу:

waste = 1

fiV = |fiV|

seV = |seV|

thiV = |thiV|

Доки не кінець строки:

Якщо fiV < seV та thiV < seV або seV < fiV та seV < thiV, то:

fiV = seV

seV = thiV

thiV = |thiV|

Якщо не fiV < seV та thiV < seV або seV < fiV та seV < thiV, то:

waste = 0

Інакше:

waste = 0

Доки не кінець строки:

thiV = |thiV|

Якщо waste = 1, то:

count = count + 1

Подання математичної постановки задачі у вигляді таблиці:

Таблиця 5.1

|  |  |  |
| --- | --- | --- |
| **Вхідні дані** | **Дії** | **Вихідні дані** |
| fiV, seV, thiV – елементи наборів, цілого типу. | waste = 1  fiV = |fiV|  seV = |seV|  thiV = |thiV|  Доки не кінець строки:  Якщо fiV < seV та thiV < seV або seV < fiV та seV < thiV, то:  fiV = seV  seV = thiV  thiV = |thiV|  Якщо не fiV < seV та thiV < seV або seV < fiV та seV < thiV, то:  waste = 0  Інакше:  waste = 0  Доки не кінець строки:  thiV = |thiV|  Якщо waste = 1, то:  count = count + 1 | count – к-сть пилкоподібних наборів, цілого типу. |

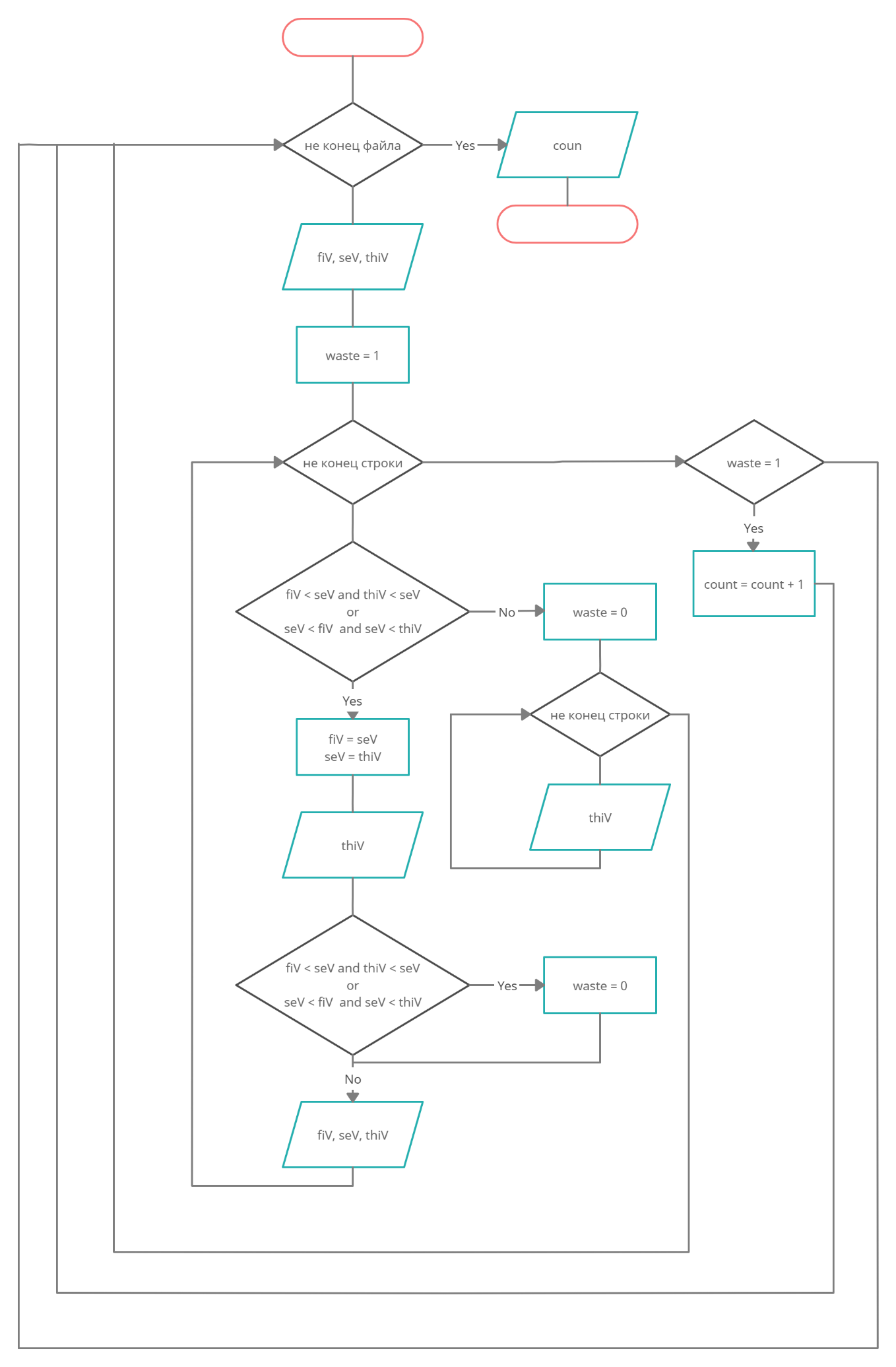


Рисунок 5.1 – Схема алгоритму для задачі 1

**5.2 Тестування програми**

Для доведення працездатності роботи програми проведемо таку перевірку:

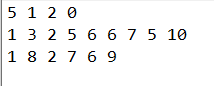


Рисунок 5.2 – Вхідні дані для тестування.

Маємо отримати:

count = 2

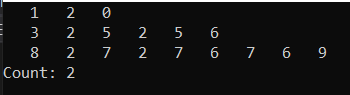


Рисунок 5.3 – Тестування програми.

**5.3 Текст програмною мовою С++.**

#**include** <iostream>

#**include** <string>

#**include** <fstream>

#**include** <sstream>

#**include** <iomanip>

**using namespace** std;

**void** main()

{

**string** path = "valuesEx4extra.txt";

**ifstream** fin;

fin.open(path);

**if** (!fin.is\_open())

{

cout << "Error! File can`t be open." << endl;

}

**else**

{

**int** count = 0;

**char** getString[20];

**int** fiV, seV, thiV;

**while** (!fin.eof())

{

fin.getline(getString, 20);

**stringstream** iss(getString);

**int** waste = 1;

iss >> fiV >> seV >> thiV;

**while** (!iss.eof())

{

**if** ((fiV < seV && thiV < seV) || (seV < fiV && seV < thiV))

{

fiV = seV;

seV = thiV;

iss >> thiV;

**if** (!((fiV < seV && thiV < seV) || (seV < fiV && seV < thiV))) waste = 0;

cout << setw(4) << fiV << setw(4) << seV << setw(4) << thiV;

}

**else**

{

waste = 0;

while (!iss.eof()) iss >> setw(4) >> thiV;

}

}

cout << endl;

**if** (waste) ++count;

}

cout << "Count: " << count << endl;

fin.**close**();

}

}