

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт
з лабораторної роботи № 9 з дисципліни
«Алгоритми та структури даних-1.
Основи алгоритмізації»
«Дослідження алгоритмів обходу масивів»

Варіант __16__

Виконав студент __ІП-15, Куманецька І. В.____
Перевірив _Вечерковська А. С._____

Київ 2021

Лабораторна робота 9

Дослідження алгоритмів обходу масивів

Мета – дослідити алгоритми обходу масивів, набути практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Індивідуальне завдання

Варіант 16

Завдання

Розробити алгоритм та написати програму, яка складається з наступних дій:

1. Опису змінної індексованого типу (двовимірний масив) згідно з варіантом (табл. 1).
2. Ініціювання змінної, що описана в п.1 даного завдання.
3. Обчислення змінної, що описана в п.1, згідно з варіантом (табл. 1).

16	Задано матрицю дійсних чисел $A[m,n]$. В кожному стовпчику матриці знайти перший від'ємний елемент X і його місцезнаходження. Обміняти знайдене значення X з елементом останнього рядка.
----	---

Постановка задачі

Створити двовимірний масив з довільними елементами. Для кожного з стовпців вирахувати перший від'ємний елемент та поміняти його місцями з елементом останнього рядка.

Побудова математичної моделі

Змінна	Тип	Ім'я	Призначення
Задана матриця	Дійсне число	A	Вхідні та кінцеві дані
Кількість рядків у матриці	Ціле число	m	Вхідні дані
Кількість стовпців у матриці	Ціле число	n	Вхідні дані

Матриця, що розглядається в функції	Дійсне число	matrix	Проміжні дані
Номер стовпця матриці, що розглядається в функції	Ціле число	column	Проміжні дані
Перший від'ємний елемент в стовпчику	Дійсне число	elem	Проміжні дані
Номер рядка, що використовується в функції	Ціле число	row	Проміжні дані
Номер стовпця, ітератор циклу	Ціле число	i	Проміжні дані
Номер рядка з першим від'ємним елементом	Ціле число	change	Проміжні дані

Створюємо матрицю за введеною кількістю стовпців та рядків і заповнюємо її випадковими числами. Функція `find_negative(matrix, column)` повертає номер рядка `matrix` з першим від'ємним числом у стовпчику під номером `column`. У циклі з передумовою перевіряється, чи є конкретний елемент стовпця від'ємним, поки його не буде знайдено. У разі, якщо від'ємного числа не знайдено, функція повертає 0. В основному за допомогою арифметичного циклу переглядаються усі стовпці матриці, де результат функції передається змінній `change`. Якщо `change` не дорівнює 0, тобто в стовпці є хоча б 1 від'ємний елемент, його міняють місцями з елементом останнього рядка.

Через `a += b` позначається операція $a = a + b$, через `a != b` позначається «не дорівнює».

Розв'язання

Програмні специфікації запишемо у псевдокодi, графічній формi у вигляді блок-схеми та у вигляді коду.

Крок 1. Визначення основних дій.

Крок 2. Деталізація пошуку першого від'ємного значення в кожному стовпці.

Крок 3. Деталізація обміну місцями знайденого елемента з елементом останнього рядка.

Псевдокод

початок

введення m та n

введення A

повторити для i від 1 до n

$change := \text{find_negative}(A, i)$

якщо $change \neq 0$

то $A[change][i] := A[change][i] + A[m][i]$

$A[m][i] := A[change][i] - A[m][i]$

$A[change][i] := A[change][i] - A[m][i]$

все якщо

все повторити

кінець

Підпрограми

find_negative(matrix, row, column)

початок

$r := 1$

$elem := 0$

поки $r \leq \text{row} \ \&\& \ \text{elem} = 0$

повторити

якщо $\text{matrix}[r][\text{column}] < 0$

то $\text{elem} := \text{matrix}[r][\text{column}]$

$r += 1$

все якщо

все повторити

якщо $\text{elem} = 0$

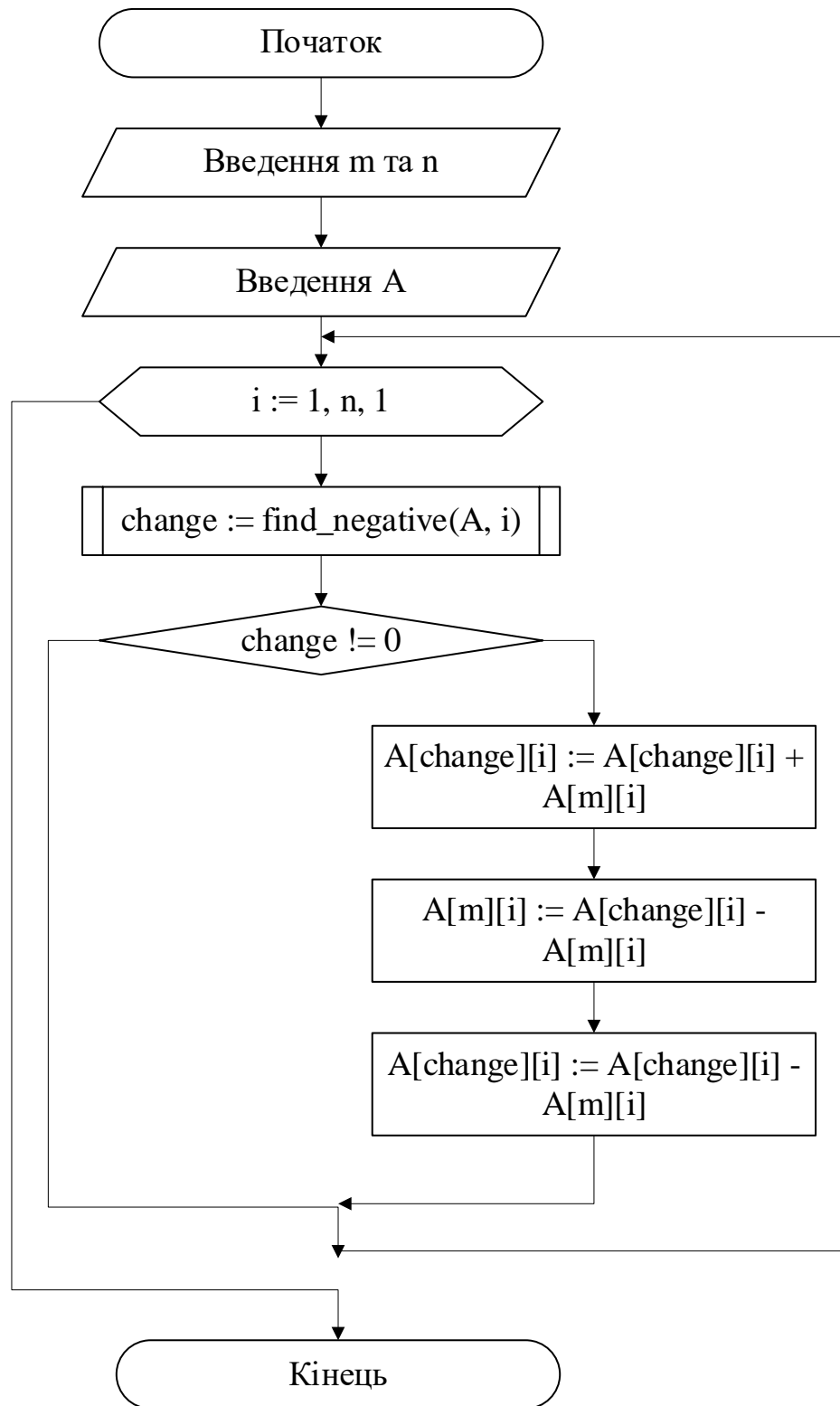
то $r := 1$

все якщо

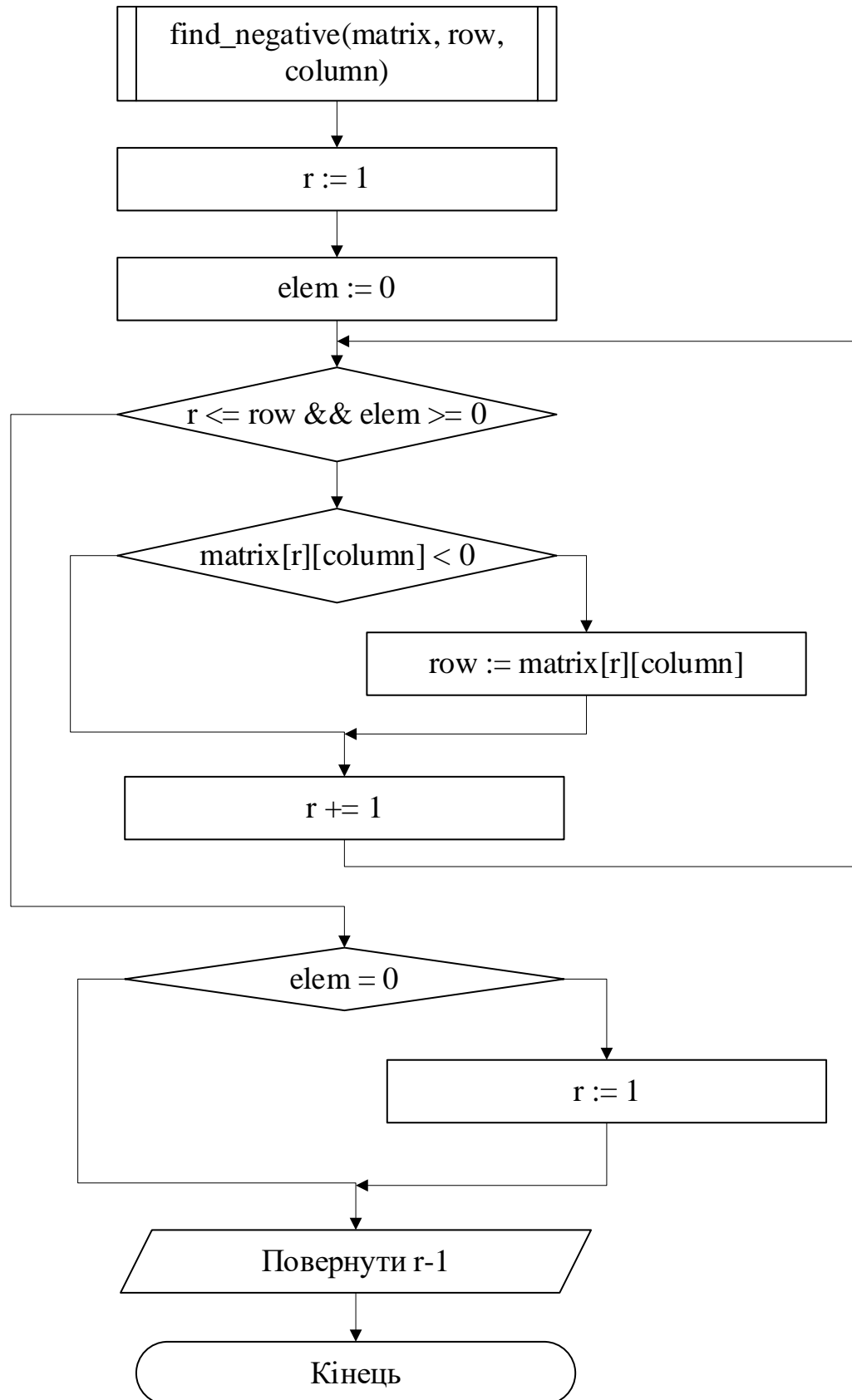
повернути $r-1$

кінець

Блок-схема



Підпрограма



Код

```
#include <stdio.h>
#include <math.h>
#include <iostream>

using namespace std;
float** gen_matrix(int, int);
void print_matrix(float**, int, int);
int find_negative(float**, int, int);

int main()
{
    int m, n;
    cout << "Enter number of rows: ";
    cin >> m;
    cout << "Enter number of columns: ";
    cin >> n;

    float** A = gen_matrix(m, n);
    cout << "The first matrix:" << endl;
    print_matrix(A, m, n);

    for (int i = 0; i < n; i++)
    {
        int change;
        change = find_negative(A, m-1, i);
        if (change != -1)
        {
            A[change][i] = A[change][i] + A[m - 1][i];
            A[m - 1][i] = A[change][i] - A[m - 1][i];
            A[change][i] = A[change][i] - A[m - 1][i];
        }
    }

    cout << "Changed matrix:" << endl;
    print_matrix(A, m, n);
}
```



```

float** gen_matrix(int row, int column) {
    srand(time(NULL));
    float** matrix = new float* [row];
    for (int i = 0; i < row; i++)
    {
        matrix[i] = new float [column];

        for (int j = 0; j < column; j++)
        {
            matrix[i][j] = -100 + (rand() % 200) + ((float)(-100 + rand() % 100)) / 100;
        }
    }
    return matrix;
}

void print_matrix(float** matrix, int row, int column) {
    for (int i = 0; i < row; i++)
    {
        for (int j = 0; j < column; j++)
        {
            cout << matrix[i][j] << " ";
        }
        cout << endl;
    }
}

```

```

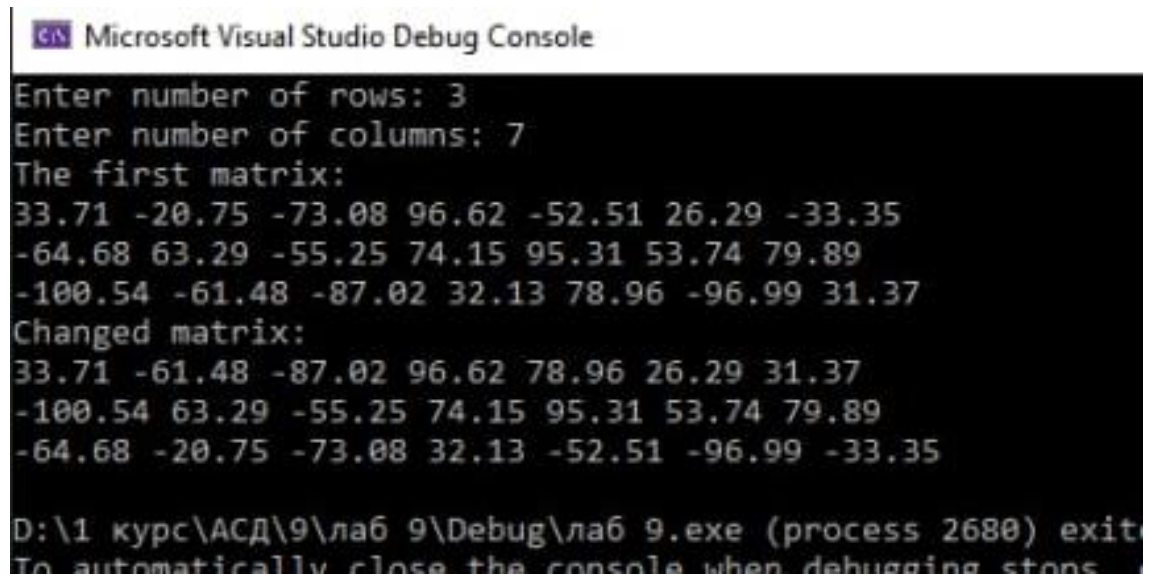
int find_negative(float** matrix, int row, int column) {
    int r = 0;
    float elem = 0;

    while (r < row and elem == 0)
    {
        if (matrix[r][column] < 0) elem = matrix[r][column];
        r += 1;
    }

    if (elem == 0) r = 0;
    return r-1;
}

```

Тестування коду



```
Microsoft Visual Studio Debug Console
Enter number of rows: 3
Enter number of columns: 7
The first matrix:
33.71 -20.75 -73.08 96.62 -52.51 26.29 -33.35
-64.68 63.29 -55.25 74.15 95.31 53.74 79.89
-100.54 -61.48 -87.02 32.13 78.96 -96.99 31.37
Changed matrix:
33.71 -61.48 -87.02 96.62 78.96 26.29 31.37
-100.54 63.29 -55.25 74.15 95.31 53.74 79.89
-64.68 -20.75 -73.08 32.13 -52.51 -96.99 -33.35
D:\1 курс\АСД\9\лаб 9\Debug\лаб 9.exe (process 2680) exit
To automatically close the console when debugging stops...
```

Висновок

Було досліджено алгоритми обходу матриць та набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій. В результаті виконання лабораторної роботи було ініційовано двовимірний масив, у кожному стовпці знайдено перший від'ємний елемент та поміняно його місцями з елементом останнього рядка, розділивши задачу на 3 кроки: визначення основних дій, деталізація пошуку першого від'ємного значення в кожному стовпці, деталізація обміну місцями знайденого елемента з елементом останнього рядка.