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 18

Виконав студент ІП-12 Кушнір Ганна Вікторівна

(шифр, прізвище, ім'я, по батькові)

Перевірив \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( прізвище, ім'я, по батькові)

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.

**Варіант 18**

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

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

Індивідуальне завдання:

Задано матрицю дійсних чисел A[m,n]. При обході матриці по стовпчиках визначити в ній присутність заданого дійсного числа X і його місцезнаходження. Обміняти знайдене значення Х з елементом середнього рядка.

1. *Постановка задачі.* Початковими даними є розмірність m×n двовимірного масиву та дійсне число X, яке потрібно буде знайти в утвореному масиві A[m,n]; ці дані вводяться користувачем з клавіатури. Результатом виконання алгоритму є або двовимірний масив A[m,n], у якому знайдений елемент X переставлений місцями з елементом середнього рядка, або повідомлення про те, що введений елемент X не було знайдено в масиві A[m,n].
2. *Побудова математичної моделі.* Складемо таблицю імен змінних.

|  |  |  |  |
| --- | --- | --- | --- |
| **Змінна** | **Тип** | **Ім’я** | **Призначення** |
| Кількість рядків двовимірного масиву | Цілий | m | Початкове дане |
| Кількість стовпців двовимірного масиву | Цілий | n | Початкове дане |
| Двовимірний масив | Дійсний | A[m,n] | Допоміжна змінна та результат |
| Шукане число X | Дійсний | X | Початкове дане |
| Номер рядка, у якому виявлено шукане число X | Цілий | iX | Допоміжна змінна |
| Номер стовпця, у якому виявлено шукане число X | Цілий | jX | Допоміжна змінна |
| Параметр арифметичного циклу | Цілий, послідовний | i | Лічильник |
| Параметр арифметичного циклу | Цілий, послідовний | j | Лічильник |
| Формальний параметр для передачі двовимірного масиву у функцію | Дійсний | arr[] | Допоміжна змінна |
| Формальний параметр для передачі кількості рядків двовимірного масиву у функцію | Цілий | m1 | Допоміжна змінна |
| Формальний параметр для передачі кількості стовпців двовимірного масиву у функцію | Цілий | n1 | Допоміжна змінна |
| Формальний параметр для передачі значення X у функцію | Дійсний | X1 | Допоміжна змінна |
| Посилання на номер рядка, у якому вперше виявлено число X | Цілий | iX1 | Допоміжна змінна |
| Посилання на номер стовпця, у якому вперше виявлено число X | Цілий | jX1 | Допоміжна змінна |
| Змінна для переривання пошуку X у випадку його знаходження | Логічний | R | Допоміжна змінна |

Складемо таблицю імен допоміжних алгоритмів (функцій).

|  |  |  |
| --- | --- | --- |
| **Функція** | **Тип результату** | **Ім’я** |
| Генерування двовимірного масиву | – | input() |
| Виведення двовимірного масиву | – | output() |
| Пошук заданого числа X у масиві A[m,n] | – | find\_X() |
| Обмін знайденого числа X з елементом середнього рядка | – | replace() |

Таким чином, математичне формулювання задачі зводиться до виконання наступних дій:

1. Введення m та n – розмірності масиву A[m,n].
2. Генерація двовимірного масиву A[m,n] та його виведення за допомогою функцій input(A, m, n) та output(A, m, n).
3. Пошук у згенерованому масиві введеного числа X за допомогою виклику функції find\_X(A, m, n, iX, jX). При цьому змінні iX та jX змінюють у функції своє значення.
4. Перевірка, чи було виявлено число X у масиві A[m,n] за допомогою альтернативної форми оператора вибору з умовою iX != -1. У випадку істинності умови, виводяться індекси числа X у двовимірному масиві A[m,n] та відбувається переставлення місцями знайденого числа X з елементом середнього рядка згенерованого масиву A[m,n] за допомогою виклику функції replace(A, m, iX, jX). Інакше, якщо хибність, – виводиться повідомлення про відсутність шуканого числа X у масиві.

* input(arr[], m1, n1) – функція, яка генерує двовимірний масив за допомогою арифметичного циклу з параметром i (i від 1 до m1 включно) з вкладеним у нього арифметичним циклом з параметром j (j від 1 до n1 включно); на кожній з ітерацій цього циклу випадковим чином генерується дійсне число arr[i,j] в межах від -100 до 100 з точністю до 2 цифр після крапки за формулою arr[i,j] := rand(). Конкретна працююча формула створюється в залежності від мови програмування.
* output(arr[], m1, n1) – функція, яка виводить переданий через параметр двовимірний масив на екран, використовуючи арифметичний цикл з параметром i (i від 1 до m1 включно), з вкладеним у нього арифметичним циклом з параметром j (j від 1 до n1 включно), і виводячи на кожній ітерації змінну arr[], яка відповідає індексу i,j (arr[i,j]).
* find\_X(arr[], X1, m1, n1, iX1, jX1) – функція, яка проводить пошук серед елементів масиву arr[m1,n1] за допомогою обходу «змійкою» по стовпцях. Змінні iX1, jX1 – посилання на змінні iX та jX відповідно головного алгоритму, тобто якщо в даній функції значення змінних iX1 або jX1 будуть змінюватися, то будуть змінюватися і значення змінних, переданих відповідно підпрограмі з головної програми.
* replace(arr[], m1, iX1, jX1) – функція, яка міняє місцями елемент arr[iX1, jX1] масиву arr[] та елемент масиву цього самого стовпця і рядка, який знаходиться посередині, тобто має індекс (m1 / 2 + m1 % 2), тобто відбувається обмін значеннями елементів arr[iX1,jX1] та arr[(m1 / 2 + m1 % 2), jX1]. Значення змінної arr[] безпосередньо впливає на значення змінної, переданої цій змінній з головної програми, тобто його зміни ведуть за собою зміни значення відповідного параметру.

Програмні специфікації запишемо у псевдокоді та графічній формі у вигляді блок-схеми.

1. Визначимо основні дії.
2. Деталізуємо дію створення та виведення двовимірного масиву A[m,n].
3. Деталізуємо ініціалізацію змінних iX та jX.
4. Деталізуємо пошук елемента X у масиві A[m,n].
5. Деталізуємо перевірку на наявність X у масиві A[m,n].
6. Деталізуємо дію перестановки елементів масиву та виведення зміненого масиву.
7. Деталізуємо функцію input().
8. Деталізуємо функцію output().
9. Деталізуємо функцію find\_X().
10. Деталізуємо функцію replace().
11. *Псевдокод алгоритму.*

*Крок 1*

**початок**

введення m та n

створення та виведення масиву A[m,n]

введення X

ініціалізація змінних iX та jX

пошук X у масиві A[m,n]

перевірка на наявність X у масиві A[m,n]

**кінець**

*Крок 2*

**початок**

введення m та n

input(A, m, n)

output(A, m, n)

введення X

ініціалізація змінних iX та jX

пошук X у масиві A[m,n]

перевірка на наявність X у масиві A[m,n]

**кінець**

*Крок 3*

**початок**

введення m та n

input(A, m, n)

output(A, m, n)

введення X

iX := -1

jX := -1

пошук X у масиві A[m,n]

перевірка на наявність X у масиві A[m,n]

**кінець**

*Крок 4*

**початок**

введення m та n

input(A, m, n)

output(A, m, n)

введення X

iX := -1

jX := -1

find\_X(A, X, m, n, iX, jX)

перевірка на наявність X у масиві A[m,n]

**кінець**

*Крок 5*

**початок**

введення m та n

input(A, m, n)

output(A, m, n)

введення X

iX := -1

jX := -1

find\_X(A, X, m, n, iX, jX)

**якщо** iX != -1

**то**

виведення iX та jX

перестановка елементів масиву та виведення зміненого масиву

**інакше**

виведення “Масив A не містить елемент X”

**все якщо**

**кінець**

*Крок 6*

**початок**

введення m та n

input(A, m, n)

output(A, m, n)

введення X

iX := -1

jX := -1

find\_X(A, X, m, n, iX, jX)

**якщо** iX != -1

**то**

виведення iX та jX

replace(A, m, iX, jX)

output(A, m, n)

**інакше**

виведення “Масив A не містить елемент X”

**все якщо**

**кінець**

1. *Псевдокод допоміжних алгоритмів (функцій).*

*Крок 7*

**початок input(arr[], m1, n1)**

**для** i від 1 до m1

**повторити**

**для** j від 1 до n1

**повторити**

arr[i,j] := rand()

**все повторити**

**все повторити**

**кінець input()**

*Крок 8*

**початок output(arr[], m1, n1)**

**для** i від 1 до m1

**повторити**

**для** j від 1 до n1

**повторити**

виведення arr[i,j]

**все повторити**

**все повторити**

**кінець output()**

*Крок 9*

**початок find\_X(arr[],X1,m1,n1,iX1,jX1)**

R := 1

j := 1

**поки** j <= n1 && R==1

**повторити**

**якщо** j%2 == 1

**то**

i := 1

**поки** i <= m1 && R==1

**повторити**

**якщо** arr[i,j]==X1

**то**

iX1 := i

jX1 := j

R := 0

**все якщо**

i++

**все повторити**

**інакше**

i := m1

**поки** i >= 1 && R==1

**повторити**

**якщо** arr[i,j]==X1

**то**

iX1 := i

jX1 := j

R := 0

**все якщо**

i--

**все повторити**

**все якщо**

j++

**все повторити**

**кінець find\_X()**

*Крок 10*

**початок replace(arr[], m1, iX1, jX1)**

cop := arr[iX1, jX1]

arr[iX1, jX1] := arr[(m1 / 2 + m1 % 2 ), jX1]

arr[(m1 / 2 + m1 % 2), jX1] := cop

**кінець replace()**

* 1. *Блок-схема алгоритму.*

***Крок 1 Крок 2***

***Крок 3 Крок 4***

***Крок 5***



***Крок 6***



* 1. *Блок-схеми допоміжних алгоритмів (функцій).*

 ***Крок 7******Крок 8***

***Крок 10***

***Крок 9***

1. *Код програми (на мові програмування C++).*

#include <iostream>

#include <stdlib.h>

#include <ctime>

#include <iomanip>

using namespace std;

void input(float\*\*, int, int);

void output(float\*\*, int, int);

void find\_X(float\*\*, float, int, int, int&, int&);

void replace(float\*\*, int, int, int);

int main()

{

int m, n;

cout << "Enter the number of rows in the two-dimensional array: "; cin >> m;

cout << "Enter the number of columns in the two-dimensional array: "; cin >> n;

float\*\* A;

A = new float\* [m];

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

A[i] = new float[n];

}

cout << "The array A:" << endl;

input(A, m, n);

output(A, m, n);

float X;

cout << "Enter the number you want to find: ";

cin >> X;

int iX = -1, jX = -1;

find\_X(A, X, m, n, iX, jX);

if (iX != -1) {

cout << "The first entry of the desired element X into the array has an index: " << iX << "; " << jX << endl;

replace(A, m, iX, jX);

cout << "New array:" << endl;

output(A, m, n);

}

else {

cout << "No X element was found in the array!" << endl;

}

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

delete[] A[i];

}

delete[] A;

system("pause");

}

void input(float\*\* arr, int m1, int n1)

{

srand(time(NULL));

for (int i = 0; i < m1; i++) {

for (int j = 0; j < n1; j++) {

arr[i][j] = -100 + (rand() % (int)pow(10, 3)) / pow(10, 3) \* 200;

}

}

}

void output(float\*\* arr, int m1, int n1)

{

for (int i = 0; i < m1; i++) {

for (int j = 0; j < n1; j++) {

cout << setw(10) << arr[i][j];

}

cout << endl;

}

}

void find\_X(float\*\* arr, float X1, int m1, int n1, int &iX1, int &jX1)

{

bool R = 1;

int i, j = 0;

while (j < n1 && R) {

if (j % 2 == 0) {

i = 0;

while (i < m1 && R) {

if (arr[i][j] == X1) {

iX1 = i;

jX1 = j;

R = 0;

}

i++;

}

}

else {

i = m1 - 1;

while (i >= 0 && R) {

if (arr[i][j] == X1) {

iX1 = i;

jX1 = j;

R = 0;

}

i--;

}

}

j++;

}

}

void replace(float\*\* arr, int m1, int iX1, int jX1)

{

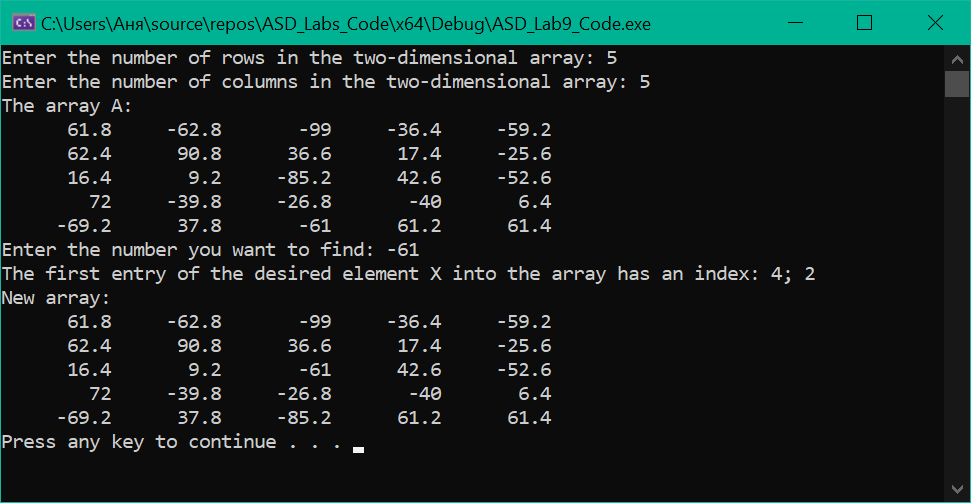
float cop;

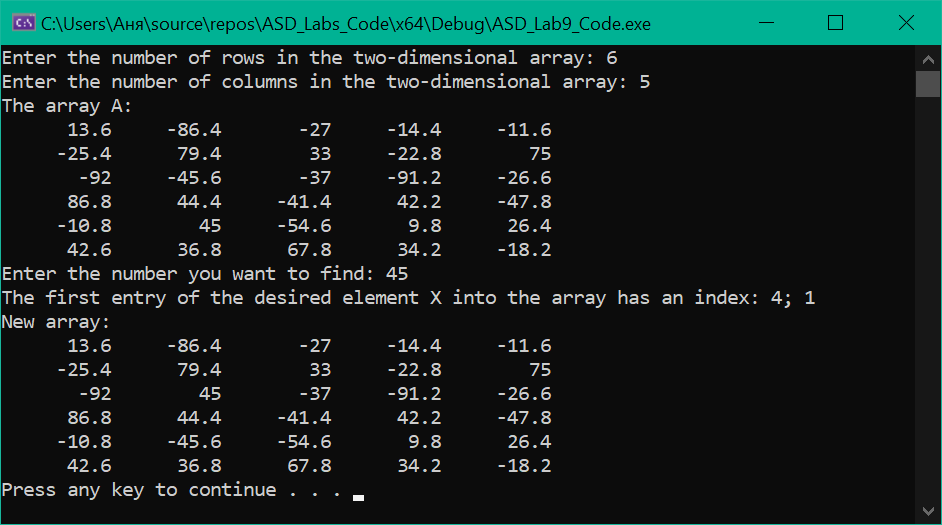
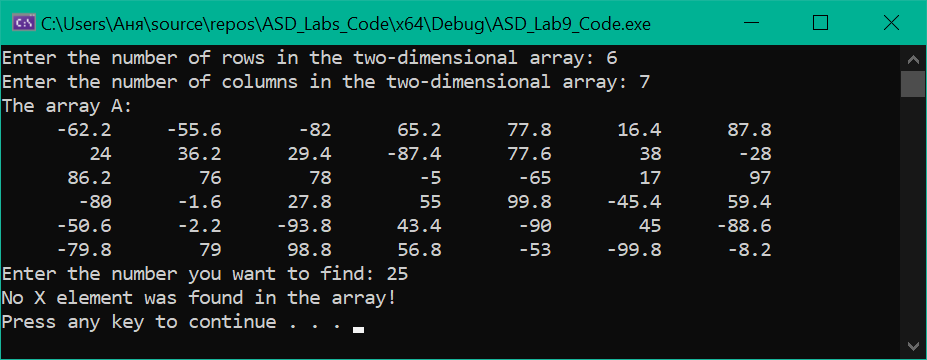
cop = arr[iX1][jX1];

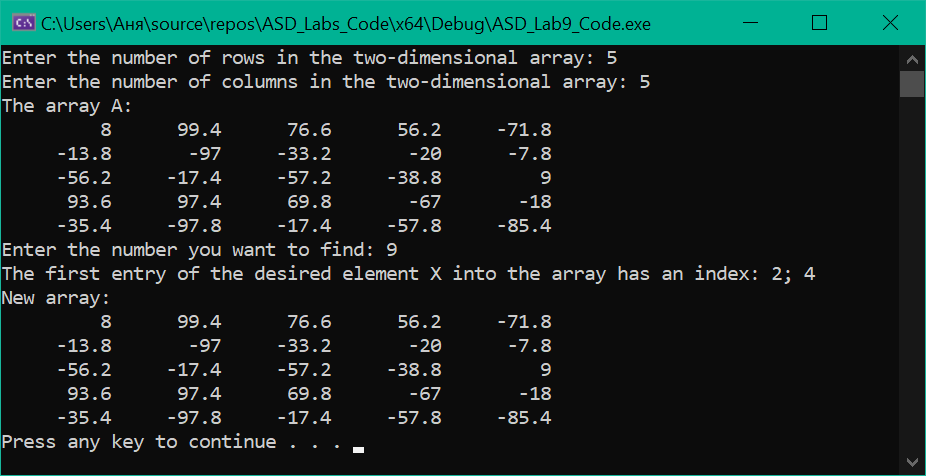
arr[iX1][jX1] = arr[m1 / 2 + m1 % 2 - 1][jX1];

arr[m1 / 2 + m1 % 2 - 1][jX1] = cop;

}

1. *Тестування програми.*

**



1. *Висновки.* На цій лабораторній роботі було досліджено алгоритми обходу масивів та було набуто практичних навичок використання цих алгоритмів під час складання програмних специфікацій.

Побудований алгоритм було покладено на мову програмування C++ та написано код, який опрацьовує виконання заданих дій. Готову програму було випробувано з уведенням п’ятьох різних початкових даних.

У першому тестуванні було введено розмірність двовимірного масиву 5×5, на що програма згенерувала та вивела масив такої розмірності. Далі було введено значення X=-61. Даний елемент був знайдений під індексом 4;2, тобто в 5-му рядку 2-го стовпця. Далі його було переставлено місцями з елементом цього ж (2-го) стовпця середнього рядка (в даному випадку середнім є 3-ій рядок). Змінений двовимірний масив було виведено на екран.

У другому тестуванні було введено розмірність 6×5, на що програма згенерувала та вивела масив заданої розмірності. Далі було введено значення X=45. Даний елемент було знайдено під індексом 4;1, тобто в 5-му рядку 1-го стовпця. Далі його було переставлено місцями з елементом цього ж (1-го) стовпця середнього рядка (в даному випадку введена кількість рядків (6) – парне число, тому єдиного середнього рядка немає, отже, на етапі побудови алгоритму було вирішено в таких випадках за середній брати рядок, який знаходиться вище половини рядків, тобто в даному випадку – середнім буде 3-ій рядок). Змінений двовимірний масив було виведено на екран.

У третьому тестуванні програми було введено розмірність 6×7, на що програма згенерувала та вивела масив уведеної розмірності. Далі було введено значення X=25. Такий елемент не було знайдено у згенерованому масиві, тому було виведено відповідне повідомлення.

У останньому тестуванні було введено розмірність 5×5, на що програма знову ж таки згенерувала та вивела двовимірний масив заданої розмірності. Далі було введено значення X=9. Даний елемент був знайдений під індексом 2;4, тобто в 3-му рядку 5-го стовпця. Оскільки шуканий елемент знаходився в 3-му рядку, який є середнім серед 5-ти рядків, то необхідності переставляти цей елемент не виникло. Тому двовимірний масив було виведено без змін.

Отже, побудований алгоритм працює правильно і виконує поставлену задачу.