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

**Лабораторна робота №5**

з дисципліни  
«Алгоритми і структури даних»

Виконав: Перевірила:

студент групи ІМ-31 Молчанова А. А.  
Литвиненко Сергій Андрійович  
номер у списку групи: 14

Київ 2023

**Завдання**

1. Написати програму розв’язання задачі пошуку (за варіантом) у двовимірному масиві (матриці) методом двійкового пошуку. Алгоритм двійкового пошуку задається варіантом завдання.
2. Розміри матриці m та n взяти самостійно у межах від 7 до 10.
3. При тестуванні програми необхідно підбирати такі вхідні набори початкових значеннь матриці, щоб можна було легко відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Варіант 14:

Задано матрицю дійсних чисел A[m,n]. Окремо у першому рядку і останньому стовпчику визначити присутність заданого дійсного числа X і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №2), якщо елементи цього рядка і стовпчика впорядковані за незменшенням.

**Текст програми:**

#include <stdio.h>

#define ROWS 7

#define COLS 10

typedef double matrix[ROWS][COLS];

typedef double (\*accessFn)(const matrix, int, int);

int binarySearch(const matrix mat, int size, accessFn accesser, int accessersArg, double x) {

int left = 0, right = size - 1;

while (left < right) {

const int index = (left + right) / 2;

const double element = accesser(mat, index, accessersArg);

if (x <= element) right = index;

else left = index + 1;

}

return accesser(mat, left, accessersArg) == x ? left : -1;

}

double nextByRow(const matrix mat, int index, int row) {

return mat[row][index];

}

double nextByCol(const matrix mat, int index, int col) {

return mat[index][col];

}

void printMatrix(const matrix mat) {

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

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

printf("%8.2lf", mat[i][j]);

}

printf("\n");

}

}

int main(int argc, char\* argv[]) {

const matrix mat = {

{20.00,22.01,22.01,45.54,50.96,70.01,71.16,83.40,83.40,90.58},

{74.32,5.38,28.88,27.41,41.06,40.96,46.77,42.23,6.56,90.58},

{15.64,71.35,32.62,35.16,52.19,70.83,77.72,9.39,72.48,90.58},

{38.31,49.97,23.60,77.53,48.31,62.62,2.62,62.76,71.57,95.34},

{22.01,13.63,27.27,44.88,68.88,86.72,8.45,81.72,59.98,100.0},

{66.33,59.14,96.47,56.10,51.74,5.25,36.04,65.02,34.44,147.3},

{74.82,93.74,17.70,77.79,14.42,16.78,56.84,39.76,4.06,150.0},

};

printf("Initial matrix:\n");

printMatrix(mat);

double x;

printf("Enter an element to search: ");

scanf("%lf", &x);

const int pos1 = binarySearch(mat, COLS, nextByRow, 0, x);

const int pos2 = binarySearch(mat, ROWS, nextByCol, COLS - 1, x);

if (pos1 == -1) printf("Element %.2lf not found in the first line\n", x);

else printf("In the first row element %.2lf is located at (0, %d)\n", x, pos1);

if (pos2 == -1) printf("Element %.2lf not found in the last column\n", x);

else printf("In last column element %.2lf is located at (%d, %d)\n", x, pos2, COLS - 1);

return 0;

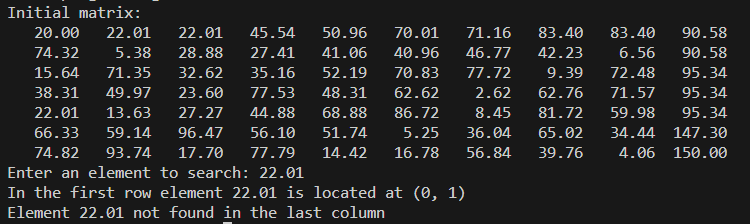
}

**Тестування програми**

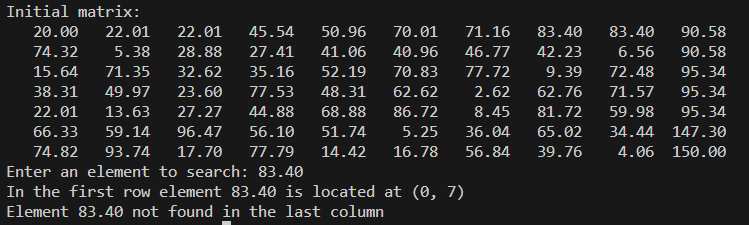
1. Матриця:



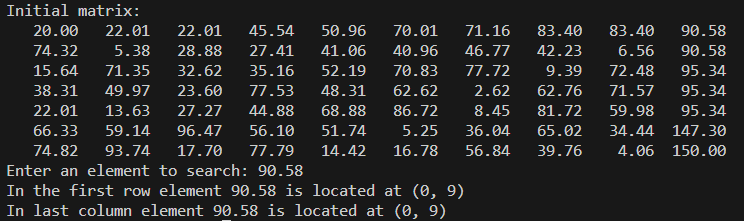
Результат пошуку числа *22.01*:



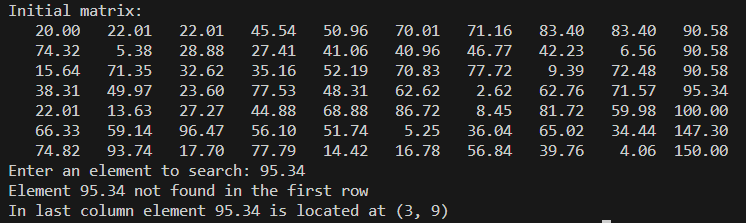
Результат пошуку числа *83.40*:



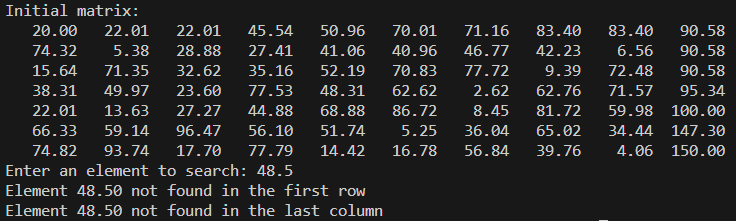
Результат пошуку числа *90.58*:



Результат пошуку числа *95.34*:



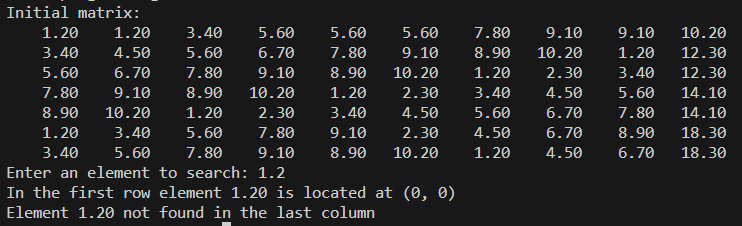
Результат пошуку числа *48.5*:



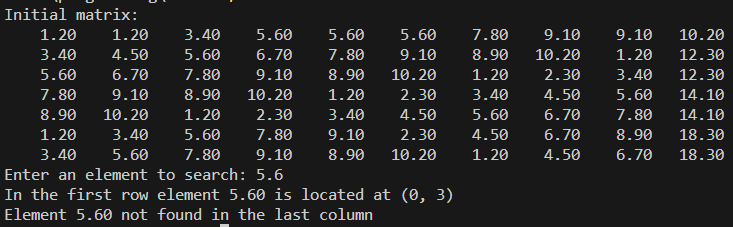
1. Матриця:



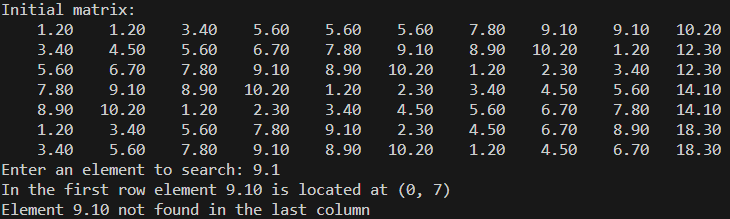
Результат пошуку числа *1.2*:



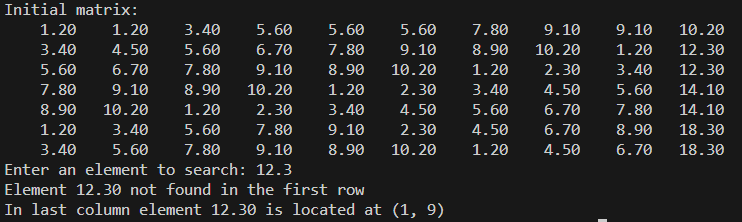
Результат пошуку числа *5.6*:



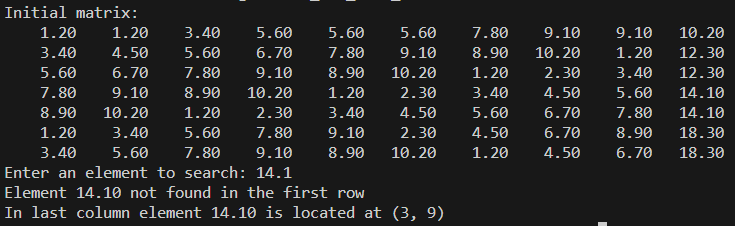
Результат пошуку числа *9.1*:



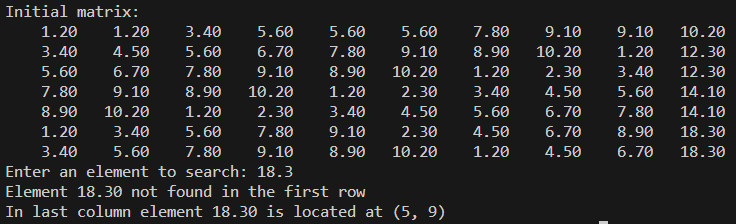
Результат пошуку числа *12.3*:



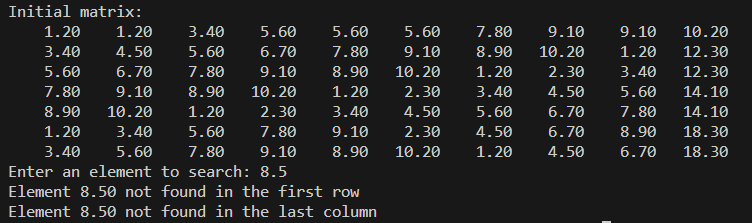
Результат пошуку числа *14.1*:



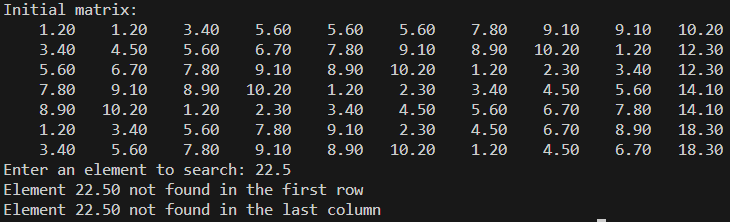
Результат пошуку числа *18.3*:



Результат пошуку числа *8.5*:



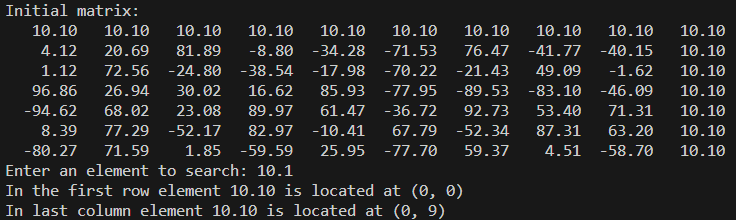
Результат пошуку числа *22.5*:



1. Матриця:



Результат пошуку числа *10.1*:



Результат пошуку числа *20.1*:

