МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дисципліни «Алгоритми та структури даних»

Виконав:

студент групи ІМ-o51

Машута Олександр Ігорович

Перевірив:

ас. Череватенко О.В.

Київ 2025

**Мета лабораторної роботи**

Метою лабораторної роботи №4. «Алгоритми двійкового пошуку» є засвоєння теоретичного матеріалу та набуття практик- них навичок рішення задачі пошуку заданої категорії елементів за допомогою різних алгоритмів методу двійкового пошуку у двови- мірних масивах.

**Постановка задачі**

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 від- стежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування до- зволяється використовувати матриці меншого розміру.

Завдання

Варіант №9

Задано матрицю дійсних чисел ***A***[***m***,***n***]. Окремо у кожному стовпчику матриці визначити присутність заданого дійсного числа

***X*** і його місцезнаходження (координати) методом двійкового пошуку (Алгоритм №1), якщо елементи кожного стовпчика окремо впорядковані за незменшенням.

Текст програми

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <ctype.h>

int main()

{

int m = 0, n = 0;

printf("Введіть кількість рядків (m) та стовпців (n) (можна через кому або пробіл): \n");

char buf[128];

if (!fgets(buf, sizeof(buf), stdin)) {

fprintf(stderr, "Помилка вводу.\n");

return 1;

}

// Розбір введених розмірів матриці

char\* p = buf;

m = (int)strtol(p, &p, 10);

while (\*p && !isdigit((unsigned char)\*p) && \*p != '-') p++;

n = (int)strtol(p, NULL, 10);

if (m <= 0 || n <= 0) {

fprintf(stderr, "Неприпустимі розміри матриці (m і n мають бути додатні).\n");

return 1;

}

double\* matrix = malloc((size\_t)m \* n \* sizeof(double));

if (!matrix) {

perror("malloc");

return 1;

}

printf("Введіть елементи матриці (елементи в кожному стовпчику впорядковані за незменшенням):\n");

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

{

for (int j = 0; j < n; j++)

{

if (scanf("%lf", &matrix[i \* n + j]) != 1) {

fprintf(stderr, "Некоректне значення матриці.\n");

free(matrix);

return 1;

}

}

}

printf("Ваша матриця:\n");

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

{

for (int j = 0; j < n; j++)

{

printf("%.3lf\t", matrix[i \* n + j]);

}

printf("\n");

}

double x;

printf("Введіть число X, яке потрібно знайти в кожному стовпчику:\n");

if (scanf("%lf", &x) != 1) {

fprintf(stderr, "Некоректне значення X.\n");

free(matrix);

return 1;

}

const double EPS = 1e-9;

int found\_any = 0;

// Шукаємо x у кожному стовпчику окремо (бінарний пошук по кожному стовпчику)

for (int j = 0; j < n; j++)

{

int L = 0, R = m - 1;

int found = 0;

while (L <= R)

{

int mid = L + (R - L) / 2;

double val = matrix[mid \* n + j];

if (fabs(val - x) <= EPS)

{

printf("Значення X знайдено у стовпчику %d на позиції (%d, %d)\n", j, mid, j);

found = 1;

found\_any = 1;

break;

}

else if (val < x)

{

L = mid + 1;

}

else

{

R = mid - 1;

}

}

if (!found)

{

printf("Значення X не знайдено у стовпчику %d\n", j);

}

}

if (!found\_any)

{

printf("Значення X не знайдено в жодному стовпчику матриці.\n");

}

free(matrix);

return 0;

}

Тестування програми

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Блок схема

A white lines on a black background

AI-generated content may be incorrect.

**Висновок**

У результаті виконання лабораторної роботи №4 я засвоїв принципи бінарного пошуку у двовимірних масивах та реалізував алгоритм пошуку заданого числа **X по стовпцях матриці**, елементи яких впорядковані за незменшенням.

Пошук реалізовано через класичний цикл while з обчисленням середнього індексу mid та коригуванням меж L і R. Програма правильно обробляє як наявність, так і відсутність елемента в кожному стовпчику.

Також я виявив, що бінарний пошук знаходить **перше-ліпше входження**, але не обов’язково найверхніше. Це не є помилкою, проте вимагає додаткової обробки, якщо потрібно знайти саме першу позицію згори.

Отже, завдання виконано успішно, і алгоритм працює коректно для заданих умов.