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:

Студент групи ІМ-42

Лобань Михайло Юрійович

номер у списку групи: 21

Перевірила:

Молчанова А. А.

Київ 2024

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 відстежити коректність виконання пошуку і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Завдання для варіанту

Варіант № 21

Задано квадратну матрицю дійсних чисел ***A***[***n***,***n***]. Визначити присутність у побічній діагоналі матриці будь-якого з чисел діапазону [**0**,**5**] і його місцезнаходження (координати) методом

двійкового пошуку (Алгоритм №1), якщо елементи цієї діагоналі впорядковані за незбільшенням.

Текст програми

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main() {
```

```
srand(time(0));

int n;
printf("enter amount of rows and columns: ");
scanf("%d", &n);
float matrix[n][n];



//generation of matrix
float random_number;
float prev;
int step = 5;

for (int i = 0; i < n; i++) {
    float r = (float)rand()/RAND_MAX;
    random_number += r;
    float repeat_chance = (float)rand()/RAND_MAX;
    for (int j = 0; j < n; j++) {
        if (i != n - j - 1) {
            matrix[i][j] = 0;
        } else {
            if(i != 0) {
                if (repeat_chance < 0.6 && repeat_chance > 0.3) {
                matrix[i][j] = prev;
                } else {
                matrix[i][j] = -step * j + 7 * random_number;
                prev = matrix[i][j];
                }
            } else {
```

```
            matrix[i][j] = -step * j + 7 * random_number;
            prev = matrix[i][j];
        }
    }
}

//matrix print
printf("\nmatrix:\n");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        printf("%8.3f ", matrix[i][j]);
    }
    printf("\n");
}

//binary search
int left = 0;
int right = n - 1;
int res_index = -1;

while (left <= right) {

    int mid = (right + left) / 2;

    if (matrix[mid][n - mid - 1] <= 5.0 && matrix[mid][n - mid - 1] >= 0.0) {
        res_index = mid;
        break;
    } else if (matrix[mid][n - mid - 1] > 5.0) {
        right = mid - 1;
```

```
        } else {
            left = mid + 1;
        }
    }

    //result output
    if (res_index != -1) {
        printf("\n(row, column): (%d, %d)\n", res_index, n - res_index - 1);
        printf("value: %.3f\n", matrix[res_index][n - res_index - 1]);
    } else {
        printf("\nno elements were found in the range [0, 5] on the secondary diagonal\n");
    }

    return 0;
}
```

Результати тестування

```
enter amount of rows and columns: 7

matrix:
   0.000    0.000    0.000    0.000    0.000    0.000  -25.332
   0.000    0.000    0.000    0.000    0.000  -18.856    0.000
   0.000    0.000    0.000    0.000  -12.666    0.000    0.000
   0.000    0.000    0.000   -7.394    0.000    0.000    0.000
   0.000    0.000   -1.901    0.000    0.000    0.000    0.000
   0.000    3.101    0.000    0.000    0.000    0.000    0.000
  13.840    0.000    0.000    0.000    0.000    0.000    0.000

(row, column): (5, 1)
value: 3.101

Process returned 0 (0x0)   execution time : 2.831 s
Press any key to continue.
```

```
enter amount of rows and columns: 8

matrix:
   0.000    0.000    0.000    0.000    0.000    0.000    0.000  -32.649
   0.000    0.000    0.000    0.000    0.000    0.000  -32.649    0.000
   0.000    0.000    0.000    0.000    0.000  -32.649    0.000    0.000
   0.000    0.000    0.000    0.000   -8.497    0.000    0.000    0.000
   0.000    0.000    0.000   -2.576    0.000    0.000    0.000    0.000
   0.000    0.000    7.373    0.000    0.000    0.000    0.000    0.000
   0.000   15.461    0.000    0.000    0.000    0.000    0.000    0.000
  24.853    0.000    0.000    0.000    0.000    0.000    0.000    0.000

no elements were found in the range [0, 5] on the secondary diagonal

Process returned 0 (0x0)   execution time : 2.054 s
Press any key to continue.
```

```
enter amount of rows and columns: 9

matrix:
   0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000  -35.249
   0.000    0.000    0.000    0.000    0.000    0.000    0.000  -25.829    0.000
   0.000    0.000    0.000    0.000    0.000    0.000  -25.829    0.000    0.000
   0.000    0.000    0.000    0.000    0.000  -25.829    0.000    0.000    0.000
   0.000    0.000    0.000    0.000    2.597    0.000    0.000    0.000    0.000
   0.000    0.000    0.000   10.807    0.000    0.000    0.000    0.000    0.000
   0.000    0.000   22.071    0.000    0.000    0.000    0.000    0.000    0.000
   0.000   29.407    0.000    0.000    0.000    0.000    0.000    0.000    0.000
  37.740    0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000

(row, column): (4, 4)
value: 2.597

Process returned 0 (0x0)   execution time : 1.009 s
Press any key to continue.
```

```
enter amount of rows and columns: 10

matrix:
   0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000  -40.228
   0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000  -28.322    0.000
   0.000    0.000    0.000    0.000    0.000    0.000    0.000  -20.617    0.000    0.000
   0.000    0.000    0.000    0.000    0.000    0.000  -13.611    0.000    0.000    0.000
   0.000    0.000    0.000    0.000    0.000  -13.611    0.000    0.000    0.000    0.000
   0.000    0.000    0.000    0.000    3.233    0.000    0.000    0.000    0.000    0.000
   0.000    0.000    0.000    9.379    0.000    0.000    0.000    0.000    0.000    0.000
   0.000    0.000   18.938    0.000    0.000    0.000    0.000    0.000    0.000    0.000
   0.000   18.938    0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000
  31.971    0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000    0.000

(row, column): (5, 4)
value: 3.233

Process returned 0 (0x0)   execution time : 1.901 s
Press any key to continue.
```

```
enter amount of rows and columns: 5

matrix:
   0.000    0.000    0.000    0.000  -15.206
   0.000    0.000    0.000   -3.998    0.000
   0.000    0.000    2.013    0.000    0.000
   0.000    8.631    0.000    0.000    0.000
  15.557    0.000    0.000    0.000    0.000

(row, column): (2, 2)
value: 2.013

Process returned 0 (0x0)   execution time : 1.013 s
Press any key to continue.
```

Висновки

В результаті виконання завдання було створено програму для розв'язання задачі пошуку елемента у двовимірному масиві методом двійкового пошуку. У ході тестування програма продемонструвала стабільну роботу, забезпечуючи правильне виконання пошуку для різних вхідних даних.