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 №2.2

з дисципліни

«Алгоритми і структури даних»

Виконала:

студентка групи ІМ-21

Рабійчук Дар’я Олександрівна

номер у списку групи: 18

Перевірила:

Молчанова А.А.

Київ 2022

Постановка задачі :

1) Задано двовимірний масив(матрицю) цілих чисел A[m, n] або A[n, n], де m та n – натуральні числа (константи), що визначають розміри двовимірного масиву. Виконати сортування цього масиву або заданої за варіантом його частини у заданому порядку заданим алгоритмом(методом).

Сортування повинно бути виконано безпосередньо у двовимірному масиві «на тому ж місці», тобто без переписування масиву та/або його будь-якої частини до інших одно- або двовимірних масивів, а також без використання спискових структур даних.

2) Розміри матриці m та n взяти самостійно у межах від 7 до 10.

3) При тестуванні програми необхідно підбирати такі вхідні набори початкових значень матриці, щоб можна було легко відстежити коректність виконання сортування і ця коректність була б протестована для всіх можливих випадків. З метою тестування дозволяється використовувати матриці меншого розміру.

Варіант № 18

Задано двовимірний масив (матрицю) цілих чисел $A[m,n]$. Відсортувати окремо кожен стовпчик масиву методом швидкого сортування (методом Хоара) за незбільшенням.

Текст програми:

```
#include <stdio.h>
#include <windows.h>
#define decrease 1
#define increase -1

int main(){

  const int matrix[7][8] = {
  {9, 2, 3, 8, 7, 6, 4, 7},
  {10, 8, 0, 3, 2, 7, 2, 1},
  {6, 9, 8, 2, 3, 10, 9, 2},
  {5, 3, 1, 10, 9, 5, 3, 0},
  {2, 1, 2, 1, 4, 4, 1, 4},
  {8, 7, 5, 6, 6, 8, 8, 5},
  {1, 10, 6, 7, 1, 1, 5, 8}
  };

  const int raw = 8;
  const int col = 7;
  COORD GetConsoleCursorPosition(HANDLE hConsoleOutput) {
    CONSOLE_SCREEN_BUFFER_INFO csbi;
  if(GetConsoleScreenBufferInfo(hConsoleOutput, &csbi)){
    return csbi.dwCursorPosition;
  }else{
    COORD invalid = {0, 0};
    return invalid;
    }
  }
  HANDLE hout = GetStdHandle(STD_OUTPUT_HANDLE);
  void gotoX(int x){
  x += GetConsoleCursorPosition(hout).X;
  COORD pos = {x, GetConsoleCursorPosition(hout).Y};
  SetConsoleCursorPosition(hout, pos);
  }

  void gotoY(int y){
  y += GetConsoleCursorPosition(hout).Y;
  COORD pos = {GetConsoleCursorPosition(hout).X, y};
  SetConsoleCursorPosition(hout, pos);
  }

  void drawMatrix(int matrix[col][raw]){
  printf("\n\n");
  for(int i = 0; i < col; i++){
    for(int j = 0; j < raw; j++){
        gotoX(2);
        printf("%3d", matrix[i][j]);
    }
    printf("\n\n");
```

```
  }
  printf("\n");
  }
int sortMatrix(int matrix[col][raw],int direction){
 int hoar(int m[col][raw], int L,int R,int i,int dir){
  int K = L, M = R;

  int T = m[L][i];
  while(L < R){
    while(m[R][i]*direction < T*direction && L < R){
        R--;
    }
    if(L != R){
        m[L][i] = m[R][i];
        L++;
    }
    while (m[L][i]*direction > T*direction && L < R){
        L++;
    }
    if(L != R){
        m[R][i] = m[L][i];
        R--;
    }
  }
    m[L][i] = T;
    int P = L;
    L = K, R = P - 1;

    if(P != L && P != R){
        m = hoar(m, L, R, i, dir);
    }
    L = P + 1;
    R = M;
    if(P != L && P != R){
        m = hoar(m, L, R, i, dir);
    }
    return m;
 }
    for(int i = 0; i < raw; i++){
        matrix = hoar(matrix, 0, col - 1, i, direction);
    }
    return matrix;
  }
  int x = 0;
  while(x != 1 && x !=2 && x !=3){
    printf("Hello!\nChoose the matrix for testing: \n1)original\n2)sorted\n3)reversed-sorted\n");
    scanf("%d", &x);
  }

  switch(x){

  case 1://сортування початкового масиву
      printf("Original matrix:\n");
      drawMatrix(matrix);

     printf("Sorted matrix: \n");
     drawMatrix(sortMatrix(matrix, decrease));
     break;

  case 2://сортування вже відсортованого масиву
      printf("Originally sorted: \n");
      drawMatrix(sortMatrix(matrix, decrease));

      printf("Resorted matrix: \n");
      drawMatrix(sortMatrix(sortMatrix(matrix, decrease), decrease));
      break;

  case 3://сортування оберненого відсортованого масиву
      printf("Reversed-sorted matrix: \n");
      drawMatrix(sortMatrix(matrix, increase));
      printf("Resorted reversed-sorted matrix: \n");
      drawMatrix(sortMatrix(sortMatrix(matrix, increase), decrease));
      break;
  }
    return 0;
}
```

Результат тестування програми:

```
Hello!
Choose the matrix for testing:
1)original
2)sorted
3)reversed-sorted
1
Original matrix:


    9    2    3    8    7    6    4    7

   10    8    0    3    2    7    2    1

    6    9    8    2    3   10    9    2

    5    3    1   10    9    5    3    0

    2    1    2    1    4    4    1    4

    8    7    5    6    6    8    8    5

    1   10    6    7    1    1    5    8


Sorted matrix:


   10   10    8   10    9   10    9    8

    9    9    6    8    7    8    8    7

    8    8    5    7    6    7    5    5

    6    7    3    6    4    6    4    4

    5    3    2    3    3    5    3    2

    2    2    1    2    2    4    2    1

    1    1    0    1    1    1    1    0



Process returned 0 (0x0)   execution time : 1.984 s
Press any key to continue.
```

```
Hello!
Choose the matrix for testing:
1)original
2)sorted
3)reversed-sorted
2
Originally sorted:


   10   10    8   10    9   10    9    8

    9    9    6    8    7    8    8    7

    8    8    5    7    6    7    5    5

    6    7    3    6    4    6    4    4

    5    3    2    3    3    5    3    2

    2    2    1    2    2    4    2    1

    1    1    0    1    1    1    1    0


Resorted matrix:


   10   10    8   10    9   10    9    8

    9    9    6    8    7    8    8    7

    8    8    5    7    6    7    5    5

    6    7    3    6    4    6    4    4

    5    3    2    3    3    5    3    2

    2    2    1    2    2    4    2    1

    1    1    0    1    1    1    1    0



Process returned 0 (0x0)   execution time : 1.394 s
Press any key to continue.
```

```
Hello!
Choose the matrix for testing:
1)original
2)sorted
3)reversed-sorted
3
Reversed-sorted matrix:


    1    1    0    1    1    1    1    0

    2    2    1    2    2    4    2    1

    5    3    2    3    3    5    3    2

    6    7    3    6    4    6    4    4

    8    8    5    7    6    7    5    5

    9    9    6    8    7    8    8    7

   10   10    8   10    9   10    9    8


Resorted reversed-sorted matrix:


   10   10    8   10    9   10    9    8

    9    9    6    8    7    8    8    7

    8    8    5    7    6    7    5    5

    6    7    3    6    4    6    4    4

    5    3    2    3    3    5    3    2

    2    2    1    2    2    4    2    1

    1    1    0    1    1    1    1    0



Process returned 0 (0x0)   execution time : 1.245 s
Press any key to continue.
```