Name: Ali Albayrak

Student ID: P304320

Date: 22/09/17

Portfolio Activity Task 1.4: Searching and Sorting in 2D Array

Lecturer: Stewart Godwin

Contents

[Methods: 3](#_Toc493842465)

[Fill Method: 3](#_Toc493842466)

[Display Method: 3](#_Toc493842467)

[Sort Method: 4](#_Toc493842468)

[Search Method: 4](#_Toc493842469)

[Test Data Table: 5](#_Toc493842470)

[Results: 5](#_Toc493842471)

# Methods:

## Fill Method:

//Fill method. Filling the matrix with random values between 10 and 100 by an arraylist. Also checks for duplicates and avoid from them.

public static void Fill(ref int[,] x, int size , ArrayList list)

{

Random rnd = new Random();

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

{

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

{

int num = rnd.Next(10, 101);

//duplicate check

if (!list.Contains(num))

{

x[i, j] = num;

list.Add(num);

}

else

{

j--;

}

}

}

}

## Display Method:

//Display method for the matrix. Displays the values of matrix by its row and column number order on the screen

public static void Display(ref int[,] x, int max)

{

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

{

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

{

int mat = x[i, j];

Console.Write("[ " + mat + "] ");

}

Console.WriteLine(" ");

}

}

## Sort Method:

// Sorting Method. Checks the matrix cell values one by one and compares them with their neighbour cell`s value. Sorting matrix by ascending order to create a Young tableau.

public static void Sort(ref int[,] m, int size)

{

int temp = 0;

for (int x = 0; x < size; x++)

{

for (int y = 0; y < size; y++)

{

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

{

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

{

if (m[i, j] > m[x, y])

{

temp = m[x, y];

m[x, y] = m[i, j];

m[i, j] = temp;

}

}

}

}

}

}

## Search Method:

// Step-Wise Linear Search method. Works for sorted matrix. First it checks if target number is in the bounds of sorted matrix.

Then, searches for the user input in the 2D array (starts from top right of the matrix and goes bottom-left like steps) and if it is found, displays the number position in the array and returns true. Returns false if number can`t found.

public static bool stepWise(ref int[,] x, int max, int target)

{

if (target < x[0, 0] || target > x[max - 1, max - 1])

{

return false;

}

else

{

int row = 0;

int col = max - 1;

while (row <= max - 1 && col >= 0)

{

if (x[row, col] < target)

{

row++;

}

else if (x[row, col] > target)

{

col--;

}

else

{

Console.WriteLine(target + " Found at : Column: " + (col+1) + " Row: " + (row+1));

return true;

}

}

return false;

}

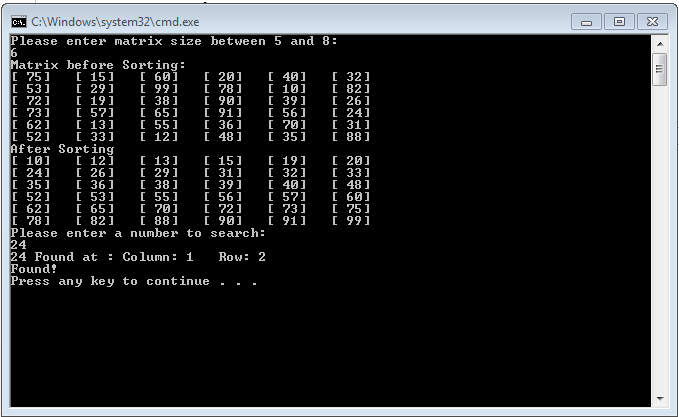
}

# Test Data Table:

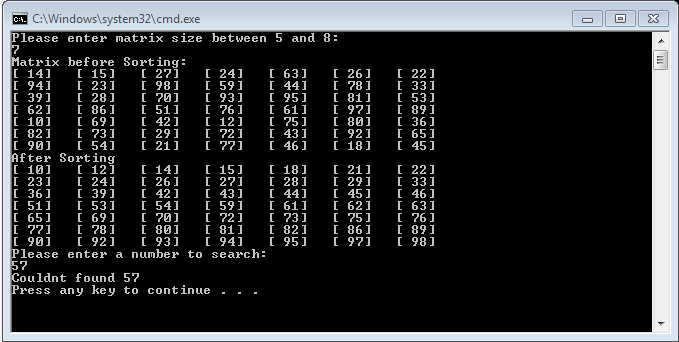
|  |  |  |
| --- | --- | --- |
| Matrix Size | Number to Search | Purpose |
| 6 | 24 | Searching a number(that is in the matrix) |
| 7 | 57 | Searching a number(that is not in the matrix) |
| 9 | - | Testing matrix bounds |
| 4 | - | Testing matrix bounds |
| 6 | 200 | Testing searching method`s bounds |

# Results:

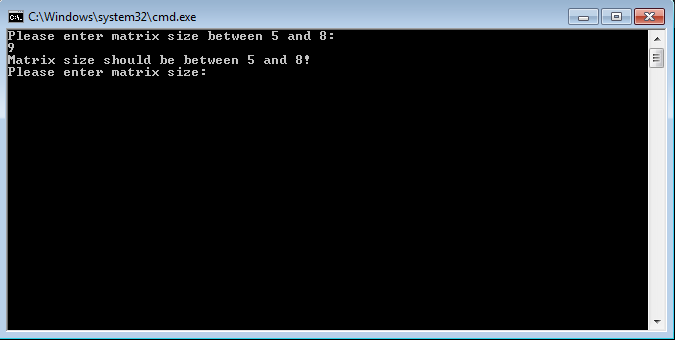
Matrix Size: 6, Number: 24



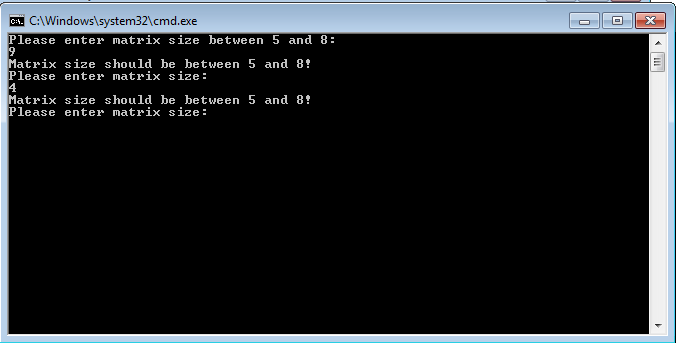
Matrix Size: 7, Number: 57



Matrix Size: 9



Matrix Size: 4



Matrix Size: 6, Number: 200

