2D Arrays

Assignment Solutions

PW SKILLS

Q1. Check if an element x exists in the given matrix or not. If it does not exist, return -1, else return its row and column index.

Input:
x = 12
arr[][] = {{3, 8, 0}, {6, 3, 2}, {12, 9, 10}}

Expected Output:
Row = 2
Column = 0

Explanation:

- Using dual loops, traverse the rows and column of the matrix.
- If element is found, return the row and column in desired format and return from the function(end it there).
- Else print -1 in the end.

Code:

```
import java.util.Arrays;
import java.util.Scanner;
public class Test{
   public static void main(String[] args){
      Scanner scn = new Scanner(System.in);
      System.out.println("Enter the dimensions of the array: ");
      int n = scn.nextInt();
      int m = scn.nextInt();
       System.out.println("Enter the element to be searched: ");
       int x = scn.nextInt();
       int[][] arr = new int[n][m];
       for(int i = 0; i < n; i++){
           for(int j = 0; j < m; j++){
               arr[i][j] = scn.nextInt();
           }
       }
       for(int i = 0; i < n; i++){
           for(int j = 0; j < m; j++){
               if(arr[i][j] == x){
                   System.out.println("Row = " + i);
                   System.out.println("Column = " + j);
                   return;
               }
           }
       }
       System.out.print(-1);
   }
}
```

```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/j
Enter the dimensions of the array:
3
3
Enter the element to be searched:
12
3 8 0
6 3 2
12 9 10
Row = 2
Column = 0

Process finished with exit code 0
```

Q2. Convert a 1D sorted array of length n*m to a 2D array of n rows and m columns. The matrix should also be sorted row and column wise.

Input:
n = 2
m = 2
arr = [1,2,3,4]

Expected Output:
[[1,2],[3,4]]

Explanation:

- Keep a pointer for current index of 1d array
- Traverse the matrix and keep adding element at idx of 1d array and increment idx.

Code:

```
import java.util.Scanner;
public class Test {
   public static void main(String[] args) {
       Scanner scn = new Scanner(System.in);
       System.out.println("Enter the dimensions of 2d array you want to convert to: ");
       int n = scn.nextInt();
       int m = scn.nextInt();
       int[] arr = new int[m*n];
       int[][] mat = new int[n][m];
       System.out.println("Enter the elements of 1D array: ");
       for(int i = 0; i < m*n; i++){
           arr[i] = scn.nextInt();
       }
```

```
    int idx = 0;
        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                mat[i][j] = arr[idx];
                idx++;
            }
        }
        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                System.out.print(mat[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/
Enter the dimensions of 2d array you want to convert to:
2 2
Enter the elements of 1D array:
1 2 3 4
1 2
3 4

Process finished with exit code 0
```

Q3. Given a 2D array of n rows and m columns, return the sum of elements along the range of row and column specified.

Input:
n = 3
m = 3
arr[][] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}
range = [0, 1], [1, 2]

Expected Output:
16

Explanation:

- Traverse from starting row of range given till end row.
- Use a pointer j, to track column for every row.
- Run j from start col to end col.
- Keep adding these elements to sum, and print it in the end.

Code:

```
import java.util.Scanner;
public class Test {
   public static void main(String[] args) {
      Scanner scn = new Scanner(System.in);
      System.out.println("Enter the dimensions of the 2d array: ");
      int n = scn.nextInt();
      int m = scn.nextInt();
      int[][] mat = new int[n][m];
      System.out.println("Enter the elements of the array: ");
      for(int i = 0; i < n; i++){
          for(int j = 0; j < m; j++){
              mat[i][j] = scn.nextInt();
          }
      }
      System.out.println("Enter the range of rows: ");
      int srow = scn.nextInt();
      int erow = scn.nextInt();
      System.out.println("Enter the range of columns: ");
      int scol = scn.nextInt();
      int ecol = scn.nextInt();
      int sum = 0;
      while(srow <= erow){
          int j = scol;
          while(j <= ecol){
              sum += mat[srow][j];
              j++;
          }
          srow++;
      }
      System.out.println(sum);
   }
}
```

```
/Library/Java/JavaVirtualMachines/jdk-19.jdk
Enter the dimensions of the 2d array:
3 3
Enter the elements of the array:
1 2 3
4 5 6
7 8 9
Enter the range of rows:
0 1
Enter the range of columns:
1 2
16

Process finished with exit code 0
```

Q4. Given a 2D array for n rows and m columns, reverse each row.

Input:
n = 3
m = 3
arr[][] = {{1, 2, 3}, {6, 7, 8}, {9, 10, 11}}

Expected Output:
{{3, 2, 1}, {8, 7, 6}, {11, 10, 9}}

Explanation:

- Traverse each row by a for loop and for every row, use 2 pointers for 1st and last column, and reverse the row using 2 pointer approach.

Code:

```
import java.util.Scanner;
public class Test {
   public static void main(String[] args) {
       Scanner scn = new Scanner(System.in);
       System.out.println("Enter the dimensions of the 2d array: ");
       int n = scn.nextInt();
       int m = scn.nextInt();
       int[][] mat = new int[n][m];
       System.out.println("Enter the elements of the array: ");
       for(int i = 0; i < n; i++){
           for(int j = 0; j < m; j++){
               mat[i][j] = scn.nextInt();
           }
       }
```

```
for(int i = 0; i < n; i++){
            int a = 0;
            int b = m-1;
            while(a < b){
                int temp = mat[i][a];
                mat[i][a] = mat[i][b];
                mat[i][b] = temp;
                a++;
                b--;
            }
        }
        for(int i = 0; i < n; i++){
            for(int j = 0; j < m; j++){
                System.out.print(mat[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

```
Test
/Library/Java/JavaVirtualMachines/jdk-19.jd
Enter the dimensions of the 2d array:
3 3
Enter the elements of the array:
1 2 3
6 7 8
9 10 11
3 2 1
8 7 6
11 10 9

Process finished with exit code 0
```

Q5. Check if an element x exists in the given sorted matrix or not. Each row and column is sorted in itself. If it does not exist, return -1, else return its row and column index.

Input:
n = 3
m = 3
arr[][] = {{1,4,7}, {2,5,8}, {3,6,9}}
x = 6

Expected Output:
Row = 2
Column = 1

Explanation:

- Use 2 pointers i and j for row and column respectively.
- Initialize i from first row and j from last column
- Use a while loop and If current element at ith and jth position matches x, print and return
- If current element is greater than x, then x lies in this row but in previous col as they are sorted, so decrement j.
- If current element is less than x, element is not in this row as we are already standing at largest element in this row, so increment x.
- Break out of loop, if any of i or j goes out of bound, in this case we return -1.

Code:

```
import java.util.Arrays;
import java.util.Scanner;
public class Test{
   public static void main(String[] args){
       Scanner scn = new Scanner(System.in);
       System.out.println("Enter the dimensions of the array: ");
       int n = scn.nextInt();
       int m = scn.nextInt();
       int[][] arr = new int[n][m];
       System.out.println("Enter the elements of the array: ");
       for(int i = 0; i < n; i++){
           for(int j = 0; j < m; j++){
               arr[i][j] = scn.nextInt();
           }
       }
       System.out.println("Enter the element to be searched: ");
       int x = scn.nextInt();
       int i = 0, j = m - 1;
       while (i < n && j >= 0) {
           if (arr[i][j] == x) {
               System.out.println("Row = " + i);
               System.out.println("Column = " + j);
               return;
           }
```

```
            if (arr[i][j] > x)
                        j--;
                    else
                        i++;
                }
                System.out.print(-1);
            }
        }
```

```
Test
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Con
Enter the dimensions of the array:
3 3
Enter the elements of the array:
1 4 7
2 5 8
3 6 9
Enter the element to be searched:
6
Row = 2
Column = 1

Process finished with exit code 0
```