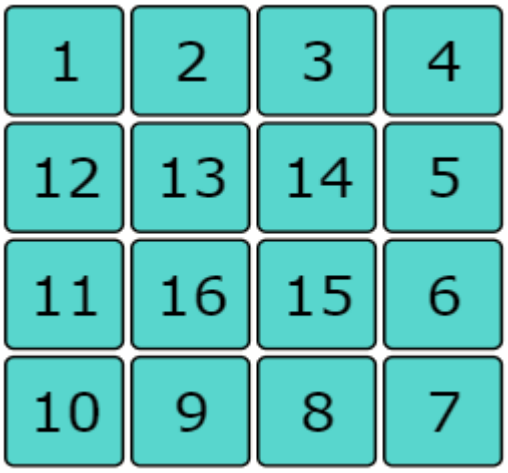
Competitive Programming Lab - 9

Academic year: 2020-2021 **Semester:** Long Sem

Faculty Name: Dr. Ajith Jublison sir **Date:** 15/ 7/ 2022

Student name: Taran Mamidala **Reg. no.:** 19BCE7346

Matrix in the spiral format :



Q1.) Print the matrix in the spiral format. User has to input the direction either it is clockwise or anticlockwise.

If clockwise, print the array starting from 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16.
If anticlockwise print the array as 1,12,11,10,9,8,7,6,6,4,3,2,13,16,15,14.

CODE :

```
import java.util.*;
public class Main {
      static void clockWiseSpiralMatrix(int m, int n, int a[][]) {
      int i, k = 0, l = 0;
     System.out.print("\n");
      while (k < m && l < n) {
            for (i = l; i < n; ++i) {
                  System.out.print(a[k][i] + " ");
            }
```

```
        k++;
        for (i = k; i < m; ++i) {
              System.out.print(a[i][n - 1] + " ");
        }
        n--;

        if (k < m) {

              for (i = n - 1; i >= l; --i) {

              System.out.print(a[m - 1][i] + " ");
              }
              m--;
        }
        if (l < n) {
              for (i = m - 1; i >= k; --i) {
              System.out.print(a[i][l] + " ");
              }
              l++;
        }
}
}
static void antiClockWiseSpiralMatrix(int m, int n, int arr[][])

{
int i, k = 0, l = 0;
int count = 0;
int total = m * n;
System.out.print("\n");
while (k < m && l < n) {
        if (count == total)
              break;

        for (i = k; i < m; ++i) {
              System.out.print(arr[i][l] + " ");
              count++;
        }
        l++;
        if (count == total)
              break;
        for (i = l; i < n; ++i) {
              System.out.print(arr[m - 1][i] + " ");
```

```
            count++;
      }
      m--;
      if (count == total)
            break;
      if (k < m) {
            for (i = m - 1; i >= k; --i) {
            System.out.print(arr[i][n - 1] + " ");
            count++;
            }
            n--;
      }
      if (count == total)
            break;
      if (l < n) {
            for (i = n - 1; i >= l; --i) {
            System.out.print(arr[k][i] + " ");
            count++;
            }
            k++;
      }
}
}
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
System.out.print("Enter number of rows : ");
int R = sc.nextInt();
System.out.print("\nEnter number of columns : ");
int C = sc.nextInt();
int a[][] = new int[R][C];

System.out.print(" \nMatrix = [");

for (int i = 0; i < R; i++) {
      System.out.print("\n[");
      a[i][0] = sc.nextInt();
      for (int j=1; j<C; j++){
      System.out.print(",");
      a[i][j] = sc.nextInt();
}
System.out.print("]");
}
```

```
        System.out.print("]\n");


        System.out.print("\nEnter 1 for Clockwise / Enter 2 for AntiClockwise : ");

        int c = sc.nextInt();

        if (c == 1)

        {
              clockWiseSpiralMatrix(R, C, a);
        } else if (c == 2)

        {
              antiClockWiseSpiralMatrix(R, C, a);
        }
        }
}
```

Output:

Result

compiled and executed in 50.909 sec(s)

```
Enter number of rows : 4
Enter number of columns : 3
Matrix = [
[3 ,2 ,1 ]
[6 ,8 ,9 ]
[2 ,5 ,4 ]
[1 ,0 ,7 ]]

Enter 1 for Clockwise / Enter 2 for AntiClockwise : 2
3 6 2 1 0 7 4 9 1 2 8 5
```

Result

compiled and executed in 68.474 sec(s)

```
Enter number of rows : 4
Enter number of columns : 4
Matrix = [
[1 ,2 ,5 ,9 ]
[7 ,5 ,4 ,2 ]
[3 ,0 ,7 ,1 ]
[8 ,4 ,6 ,5 ]]

Enter 1 for Clockwise / Enter 2 for AntiClockwise : 1
1 2 5 9 2 1 5 6 4 8 3 7 5 4 7 0
```

Result

compiled and executed in 27.344 sec(s)

```
Enter number of rows : 2
Enter number of columns : 2
Matrix = [
[3 ,6 ]
[8 ,2 ]]

Enter 1 for Clockwise / Enter 2 for AntiClockwise : 2
3 8 6 2
```

```
Enter number of rows : 3
Enter number of columns : 2
Matrix = [
[4 ,6 ]
[8 ,1 ]
[7 ,5 ]]

Enter 1 for Clockwise / Enter 2 for AntiClockwise : 2
4 8 7 5 1 6
```

```
Enter number of rows : 2
Enter number of columns : 4
Matrix = [
[3 ,1 ,2 ,4 ]
[9 ,7 ,6 ,5 ]]

Enter 1 for Clockwise / Enter 2 for AntiClockwise : 1
3 1 2 4 5 6 7 9
```

```
Enter number of rows : 3
Enter number of columns : 3
Matrix = [
[2 ,5 ,4 ]
[7 ,9 ,8 ]
[3 ,0 ,1 ]]

Enter 1 for Clockwise / Enter 2 for AntiClockwise : 1
2 5 4 8 1 0 3 7 9
```

Computing matrix

Q2.) Consider a n* matrix and compute the diagonal difference , sum of all corner elements and average of all elements in the matrix.

Sample Input

12	3	15
4	7	8
5	6	2

Sample Output

```
Diagonal difference : 4
Sum of all corner elements : 24
Average of all elements : 5.7777777
```

CODE:

```
import java.util.*;

public class Main {

	public static void main(String[] args) {
	int n;
```

```
      Scanner in=new Scanner(System.in);
      System.out.print("Enter length of an array : ");
      n=in.nextInt();
      int a[][]=new int[n][n];

      System.out.print("\nMatrix = [");

      for (int i = 0; i < n; i++) {
            System.out.print("\n[");
            a[i][0] = in.nextInt();
            for (int j=1; j<n; j++){
            System.out.print(",");
            a[i][j] = in.nextInt();
      }
      System.out.print("]");
      }
      System.out.print("]\n");

      int pd=0,npd=0,s=0;
      for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                  s=s+a[i][j];
                  if(j==i)
                  pd=pd+a[i][j];
                        if(i==n-j-1){
                  npd=npd+a[i][j];
                  }
            }
      }
      int cornerSum=a[0][0]+a[0][n-1]+a[n-1][0]+a[n-1][n-1];
      float avgElements=s/(float)(n*n);
      int differ = npd-pd;
      differ = Math.abs(differ);
      double f=3/2;
  System.out.println("\n////Obtained Results////\nDiagonal difference : "+differ);
      System.out.println("Sum of all corner elements : "+cornerSum);
            System.out.println("Average of all elements : "+avgElements);
      }
}
```

Output:

Result

compiled and executed in 28.437 sec(s)

```
 Enter length of an array : 3
 Matrix = [
[4 ,6 ,1 ]
[3 ,5 ,2 ]
[7 ,8 ,9 ]]

////Obtained Results////
 Diagonal difference : 5
Sum of all corner elements : 21
Average of all elements : 5.0
```

compiled and executed in 64.373 sec(s)

```
Enter length of an array : 4
Matrix = [
[3 ,6 ,1 ,8 ]
[2 ,5 ,9 ,7 ]
[8 ,4 ,3 ,6 ]
[5 ,1 ,4 ,9 ]]

////Obtained Results////
Diagonal difference : 14
Sum of all corner elements : 25
Average of all elements : 6.3125
```

compiled and executed in 19.358 sec(s)

```
Enter length of an array : 2
Matrix = [
[4 ,1 ]
[5 ,8 ]]

////Obtained Results////
Diagonal difference : 6
Sum of all corner elements : 18
Average of all elements : 4.5
```

Run-length Encoding

Q3.) You have been given a text message. You have to return the Run-length Encoding of the given message.

Run-length encoding is a fast and simple method of encoding strings.

The basic idea is to represent repeated successive characters as the character and a single count. For example, the string `` gggaarrrrdddenn`` is g3a2r4d4e1n2.

CODE:

```java
import java.util.*;

public class RunlengthEncoding{

 public static void main(String[] args) {
  Scanner sc = new Scanner(System.in);
  System.out.print("Enter your String : ");
      String encodedString = "";
      String givenString = sc.nextLine();

  int count = 1;
```

```
  for (int i = 0; i < givenString.length()-1; i++) {

   count = 1;
   while (i < givenString.length() - 1 && givenString.charAt(i) ==
givenString.charAt(i + 1)) {
      count++;
      i++;
   }
   encodedString = encodedString + givenString.charAt(i) + count;
  }
   System.out.println("\nString after Run-length Encoding : " + encodedString);

 }

}
```

Output:

Result

compiled and executed in 9.464 sec(s)

```
Enter your String : rrrreeeewwsdt

String after Run-length Encoding : r4e4w2s1d1
```

Result

compiled and executed in 11.859 sec(s)

```
Enter your String : eeeewwwqqqqaasf

String after Run-length Encoding : e4w3q4a2s1
```

compiled and executed in 12.874 sec(s)

```
Enter your String : wwwwwwwddddaaaaqqrryyyyyyuu

String after Run-length Encoding : w7d5a4q2r2y5u2
```

compiled and executed in 19.88 sec(s)

```
Enter your String : gggaaeeeewwwwwwqqqqquuuurrttd

String after Run-length Encoding : g3a2e4w6q5u4r2t2
```