Lesson:

2D Array Problems -1

Pre-Requisites

- Basics of Arrays
- Multi-dimensional Arrays

List of Concepts Involved:

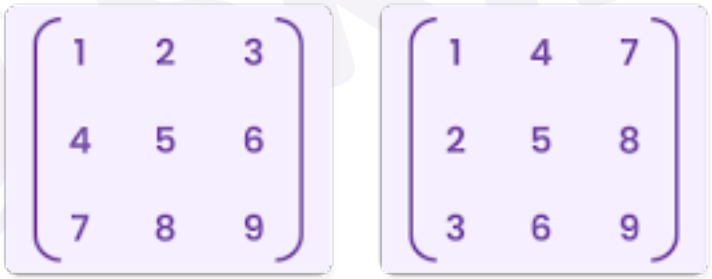
- Transpose of the matrix
- Rotation of matrix
- Pascal's Triangle

Problems

Q1. Write a program to Print the transpose of the matrix entered by the user.

Example 1:
Input :
row=3
col=3
arr[ ] = {{1,2,3}, {4,5,6}, {7,8,9}}

Output : 1 4 7 2 5 8 3 6 9



Input Output

Transpose of a matrix is obtained by changing rows to columns and columns to rows. In other words, the transpose of arr[row][col] is obtained by changing arr[i][j] to arr[j][i].

Approach:

- Take the size of matrices through user input.
- Take the matrice input.
- Create a new Matrix to store the transpose of the matrix.
- Traverse each element of the matrix and. Store this in the new matrix trans[i][j] = arr1[j][i].
- Print the final transposed matrix

Solution:

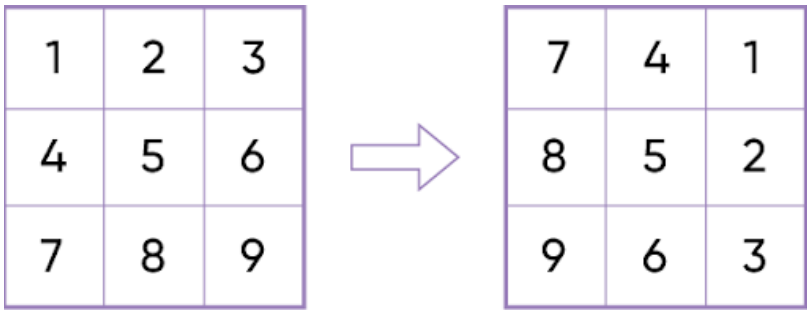
```
import java.io.*;
import java.util.*;
public class Main{
  public static void main(String args[]){
            int m,n;
          Scanner sc=new Scanner(System.in);
          System.out.println("enter the number of rows=");
          m=sc.nextInt();
          System.out.println("enter the number of column=");
          n=sc.nextInt();
          int arr1[][]=new int[m][n];
          int i,j;
          System.out.println("enter the matrix element=\n");
          for(i=0;i<m;i++)
          {
                    for(j=0;j<n;j++)
                    {
                        arr1[i][j]=sc.nextInt();
                    }
          }
          int trans[][]=new int[m][n];
          for(i=0;i<n;i++)
              {
                  for(j=0;j<m;j++)
                  {
                      trans[i][j]=arr1[j][i];
                  }
              }
          //for printing result
          for(i=0;i<m;i++)
          {
                    for(j=0;j<n;j++)
                    {
                        System.out.print(trans[i][j] + " ");
                    }
                    System.out.println("");
          }
} }
```

```
Run:  Main
C:\Users\2018k\.jdks\openjdk-19.0.1\bin\java.e
enter the number of rows=
3
enter the number of column=
3
enter the  matrix element=

1 2 3
4 5 6
7 8 9
1 4 7
2 5 8
3 6 9

Process finished with exit code 0
```

Q2: Given a square matrix, turn it by 90 degrees in a clockwise direction without using any extra space.

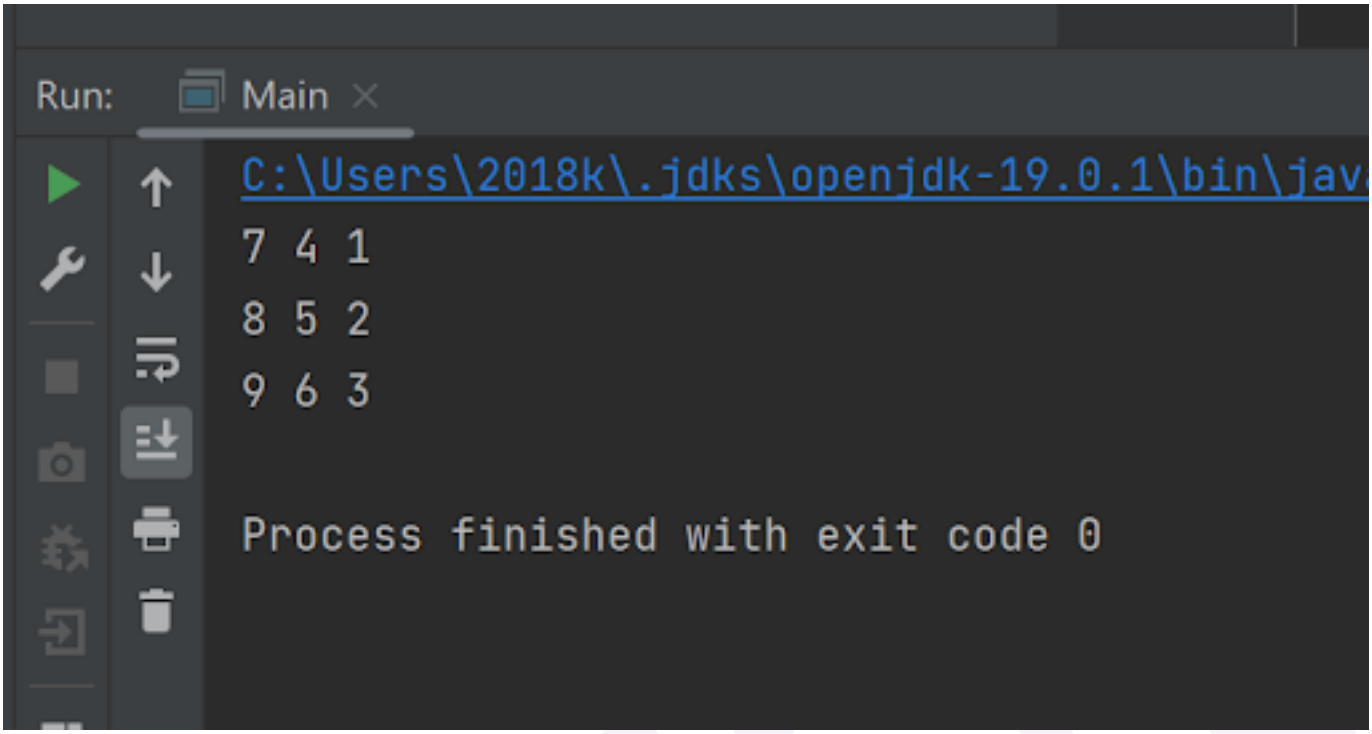


Input: arr = [[1,2,3],[4,5,6],[7,8,9]]
Output: [[7,4,1],[8,5,2],[9,6,3]

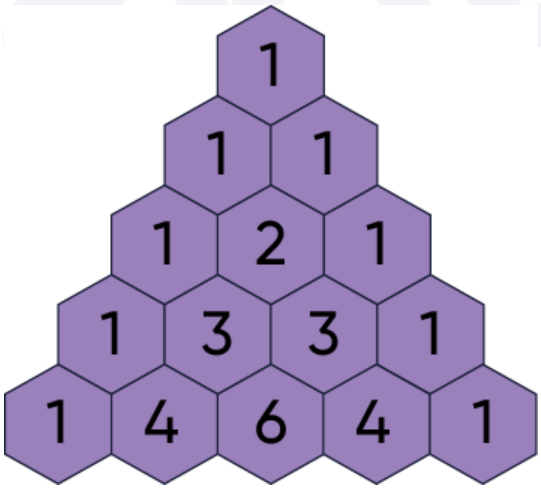
Explanation:
Let size of row and column be 3.
During first iteration –
arr[i][j] = Element at first index (leftmost corner top)= 1.
arr[j][n-1-i]= Rightmost corner top Element = 3.
arr[n-1-i][n-1-j] = Rightmost corner bottom element = 9.
arr[n-1-j][i] = Leftmost corner bottom element = 7.
Move these elements in the clockwise direction.
During second iteration –
arr[i][j] = 2.
arr[j][n-1-i] = 6.
arr[n-1-i][n-1-j] = 8.
arr[n-1-j][i] = 4.
Similarly, move these elements in the clockwise direction

Solution:

```
import java.io.*;
import java.util.*;
public class Main{
  public static void rotate(int arr[][],int N)
{
 // Traverse each cycle
          for (int i = 0; i < N / 2; i++)
          {
                    for (int j = i; j < N - i - 1; j++)
                    {
                              // Swap elements of each cycle
                              // in clockwise direction
                              int temp = arr[i][j];
                              arr[i][j] = arr[N - 1 - j][i];
                              arr[N - 1 - j][i] = arr[N - 1 - i][N - 1 - j];
                              arr[N - 1 - i][N - 1 - j] = arr[j][N - 1 - i];
                              arr[j][N - 1 - i] = temp;
                    }
          }
          }
  public static void main(String[] args){
      int[][] arr1={{1,2,3},{4,5,6},{7,8,9}};
      int n=3;
      rotate(arr1,n);
      for(int i=0;i<n;i++)
      {
          for(int j=0;j<n;j++)
          {
              System.out.print(arr1[i][j]+ " ");
          }
          System.out.println("");
      }
} }
```



Q3. Given an integer n, return the first n rows of Pascal's triangle.
In Pascal's triangle, each number is the sum of the two numbers directly above it as shown:



Example 1:
Input: n = 5
Output: [[1],[1,1],[1,2,1],[1,3,3,1],[1,4,6,4,1]]

Example 2:
Input: n = 1
Output: [[1]]

Approach:

- Take a number of rows to be printed, n.
- Make outer iteration i for n times to print rows.
- We can observe that the first row has 1 column and as we go on the next row col keeps on increasing by 1.
- Make inner iteration j for col times.
- We can observe that the first and last element of every row is 1 .
- If the element is neither in the first nor in the last column,then the element is the sum of the two numbers directly above it .

Solution:

```
import java.io.*;
import java.util.*;
public class Main {
    public static int[][] pascal(int n) {
        int[][] ans=new int[n][];
        int col=1;
        for(int i=0;i<n;i++){
            ans[i]=new int[col];
            for(int j=0;j<col;j++){
                if(j==0){ans[i][j]=1;}
                else if(j==col-1){ans[i][j]=1;}
                else{
                    ans[i][j]=ans[i-1][j-1]+ans[i-1][j];
                }
            }
            col++;
        }
        return ans;
    }
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        int n=sc.nextInt();
        int[][] ans=pascal(n);
        for(int i=0;i<n;i++){
        for(int j=0;j<ans[i].length;j++){
            System.out.print(ans[i][j]+" ");
        }
        System.out.println("");
    }
  }
}
```

Upcoming Class Teasers

- 2D Array problems