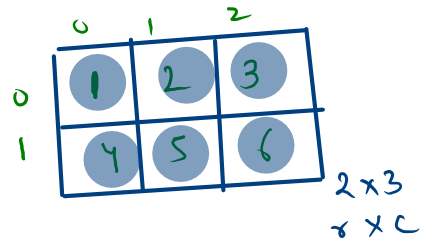
```java
public class Main
{
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);

        int [][] B = new int[2][3];
        for(int i = 0; i < 2; i++){
            for(int j = 0; j < 3; j++){
                B[i][j] = scn.nextInt();
            }
        }
        System.out.println("Now printing output");
        for(int i = 0; i < 2; i++){
            for(int j = 0; j < 3; j++){
                System.out.print(B[i][j] + " ");
            }
            System.out.println();
        }


    }
```



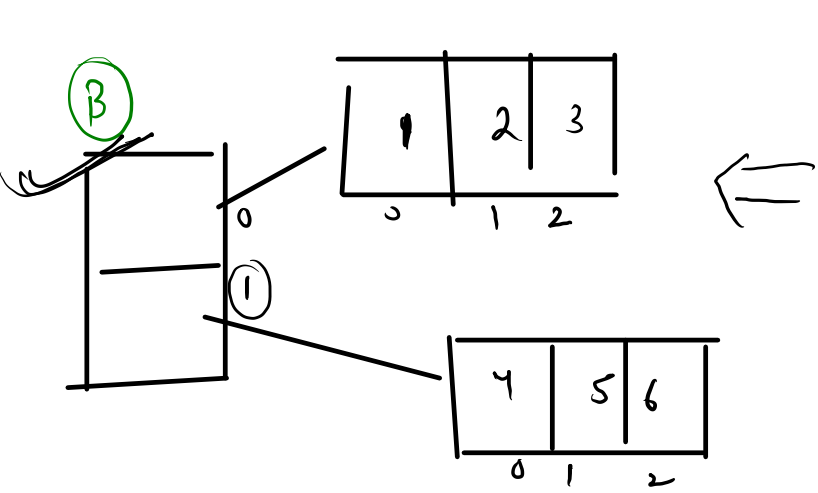
i=0 1 0<2 ✓ 1<2

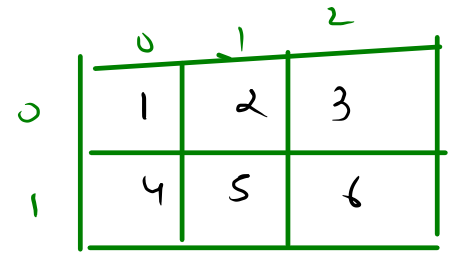
1 2 3
4 5 6

j=0 0<3.

1 2 3
4 5 6

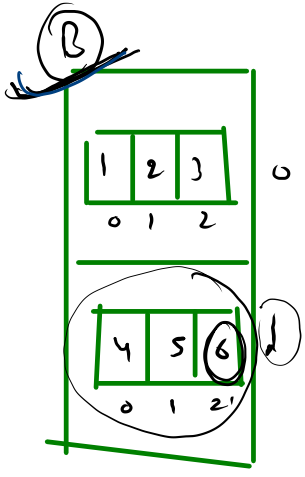
B [1] [0] =





B[i][j]

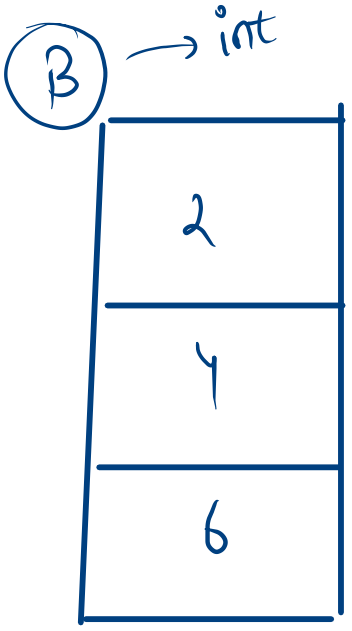
B[1][2].



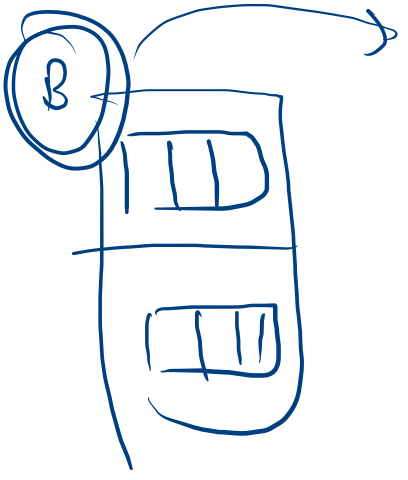
B[1][2]

row = B.length

col = B[0].length
B[1].length



len = 3.



Print the Matrix Row-wise

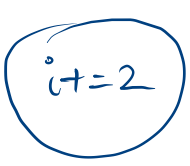
```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int m = scn.nextInt();          //row
        int n = scn.nextInt();          //col

        int [][] A = new int[m][n];     //mxn matrix
        for(int i = 0; i < m; i++){
            for(int j = 0; j < n; j++){
                A[i][j] = scn.nextInt();
            }
        }

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

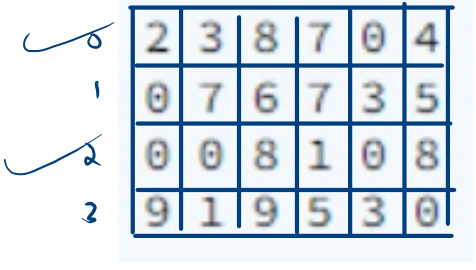
Print Alternate Row

Problem | Submissions | Leaderboard | Discussions

Once upon a time, you found yourself in possession of a **2D matrix** that had some important data. However, you quickly realized that the data was spread out across all the **rows** of the **matrix**, and you needed to organize it. Your mission, should you choose to accept it, was to **print** only the **alternate rows** of the **matrix**, starting from the very **first row**. Are you up for the challenge?

You are given a 2D matrix , your task is print the **alternate rows** of the matrix starting from the **0th** row.



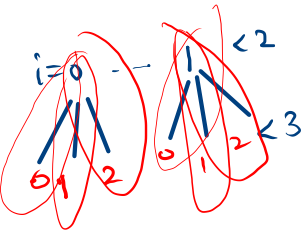


Sample Input 0

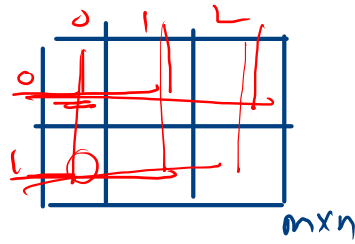
```
4
6
2 3 8 7 0 4
0 7 6 7 3 5
0 0 8 1 0 8
9 1 9 5 3 0
```

Sample Output 0

```
2 3 8 7 0 4
0 0 8 1 0 8
```



```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int m = scn.nextInt();      //2     //row
        int n = scn.nextInt();      //3     //col

        int [][] A = new int[m][n];     //mxn matrix
        for(int i = 0; i < m; i++){
            for(int j = 0; j < n; j++){
                A[i][j] = scn.nextInt();
            }
        }

        //print
        for(int i = 0; i < m; i += 2){          //row
            for(int j = 0; j < n; j++){         //col
                System.out.print(A[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```



Print Upper triangular matrix 1

Problem | Submissions | Leaderboard | Discussions

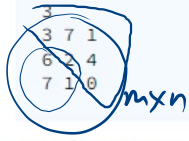
In the world of finance, every second counts. That's why a young financial analyst named Maria is given a complex **matrix** of size **m * n** to analyze, she knows that time is of the essence. The **matrix** contains vital data that could make or break her company's fortunes, and Maria has to act fast to make sense of it all.

As she delves into the **matrix**, Maria realizes that the **upper triangle** holds the key to unlocking the data's secrets.

help Maria and create program that **print** the **upper triangular matrix**.

Sample Input 0

```
3
3
3 7 1
6 2 4
7 1 0
```

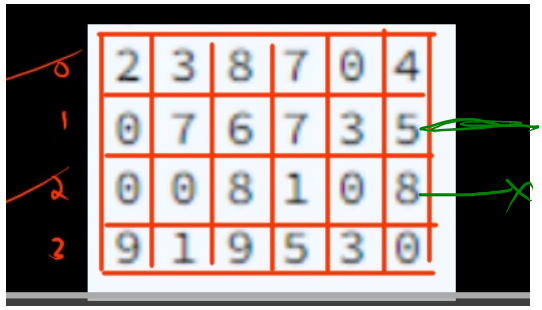


Sample Output 0

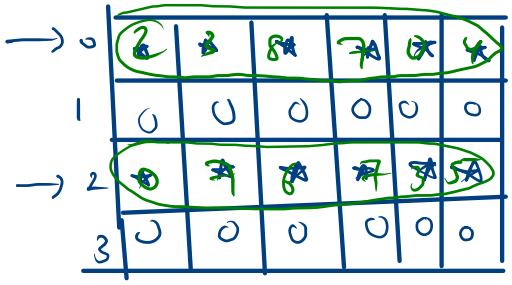
```
3 7 1
0 2 4
0 0 0
```

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner (System.in);
        int m = scn.nextInt();
        int n = scn.nextInt();
        int [][] B = new int [m][n];
        for(int i=0; i<m; i++){          /// ------  why not i+=2 here
            for(int j=0; j<n; j++){
                B[i][j]=scn.nextInt();
            }
        }

        for(int i=0; i<m; i+=2){
            for(int j=0; j<n; j++){
                System.out.print(B[i][j]+" ");
            }
            System.out.println();
        }
    }
}
```

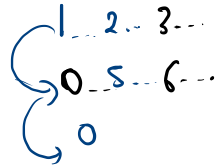
m = 4
n = 6

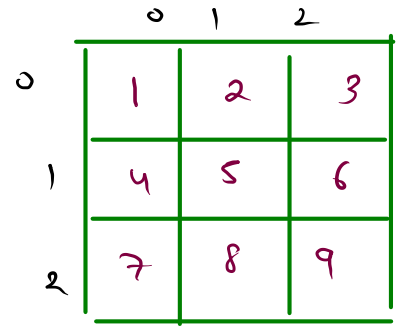


i = 0 ... 2 ...



```
18        //upper half triangle
19        for(int i = 0; i < m; i++){
20            for(int j = 0; j < n; j++){
21                if(i <= j){
22                    System.out.print(A[i][j] + " ");
23                }
24                else{
25                    System.out.print("0 ");
26                }
27            }
28            System.out.println();
29        }
30
31
```





i=2 2<3 j=0 0<3.

2≤0 ✗

i=0 0<3

j=0 0<3 0≤0 ✓

j=1 1<3 0≤1 ✓

j=2 2<3 0≤2 ✓

j=3 3<3 ✗

i=1 1<3

j=0 0<3 1≤0 ✗

j=1 1<3 1≤1 ✓

j=2 2<3 1≤2 ✓

j=3 3<3 ✗

i=2 2<3 ✓

i=0

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int m = scn.nextInt();          //row
        int n = scn.nextInt();          //col

        int [][] A = new int[m][n];     //mxn matrix
        for(int i = 0; i < m; i++){
            for(int j = 0; j < n; j++){
                A[i][j] = scn.nextInt();
            }
        }

        //upper half triangle
        for(int i = 0; i < m; i++){
            for(int j = 0; j < n; j++){
                if(i <= j){
                    System.out.print(A[i][j] + " ");
                }
                else{
                    System.out.print("0 ");
                }
            }
            System.out.println();
        }


    }
}
```

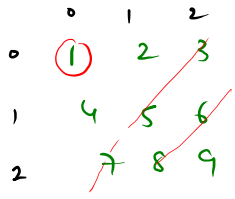
Print the matrix left-diagonal wise

Problem | Submissions | Leaderboard | Discussions

Once upon a time, you discovered a mysterious **matrix** that contained vital information. However, this **matrix** was a bit of a mess - the data was scattered all over the place, and you needed to figure out how to read it. Your task was to **print** the **matrix left-diagonal wise**, starting from the very **first upper left-diagonal**.

It was no easy feat, but with some clever problem-solving, you knew you could crack the code and uncover the secrets hidden within the matrix. Are you ready to take on the challenge?

Take a matrix of size **n * n** as input and Print the matrix **left-diagonal wise starting from the first upper left-diagonal**.



0,0 } = 0

0,1 / 1,0 } = 1

0,2 / 1,1 / 2,0 } = 2

1,2 / 2,1 } = 3

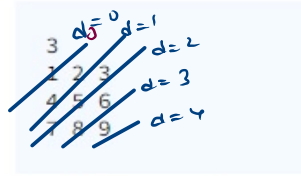
2,2 } = 4

i+j = 0 → 1st

1 → ✓

2 → ✓

Sample Input 0



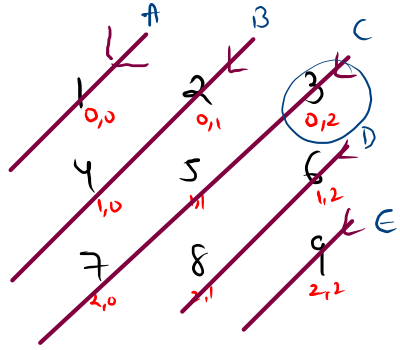
d = 2*n - 1

Sample Output 0

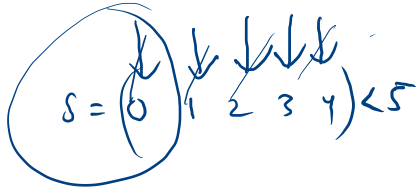
1 2 4 3 5 7 6 8 9

i+j = 0 ... 1 ... 2

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [][] A = new int[n][n];
        for(int i = 0; i < n; i++){
            for(int j = 0; j < n; j++){
                A[i][j] = scn.nextInt();
            }
        }

        int d = 2*n-1;          //total diagonal
        for(int s = 0; s < d; s++){

            for(int i = 0; i < n; i++){
                for(int j = 0; j < n; j++){
                    if(i + j == s){
                        System.out.print(A[i][j] + " ");
                    }
                }
            }

        }

    }
}
```



0
1
2
3
4



n=3 d=5

s=0

1 2 4 3

$1 \longrightarrow n^2$

$n \longrightarrow n \cdot n^2 = n^3$