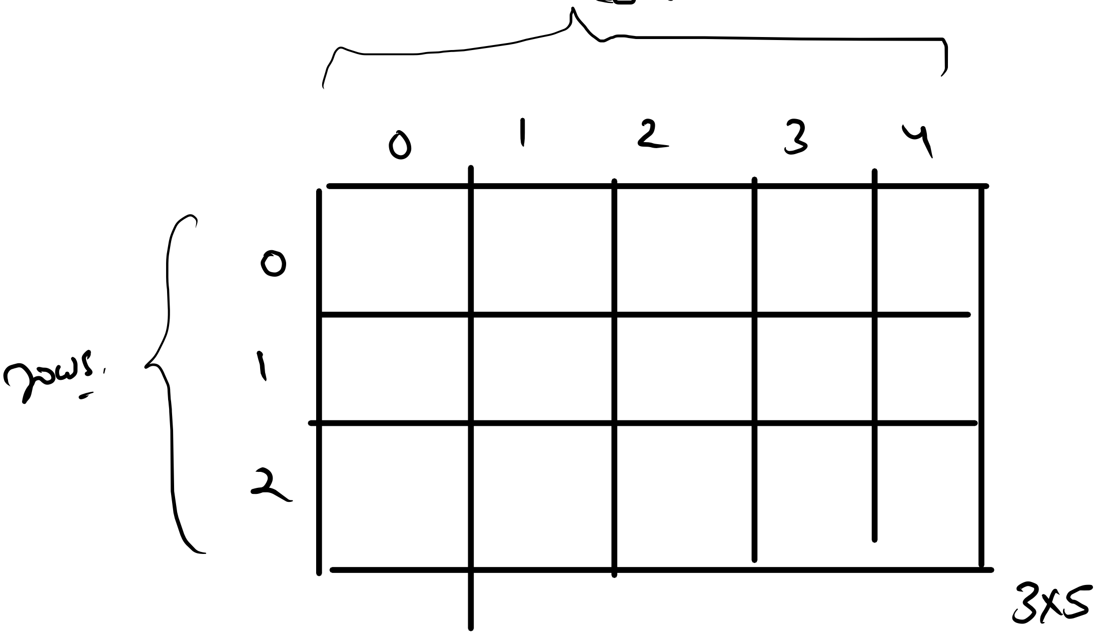
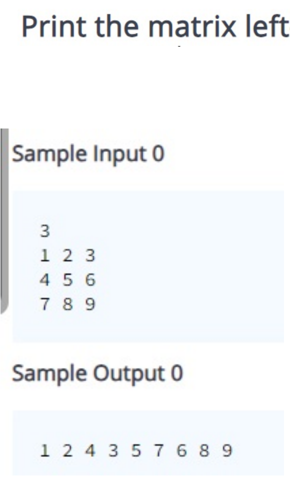
2D - Array.





Print the matrix left-diagonal wise

Sample Input 0

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

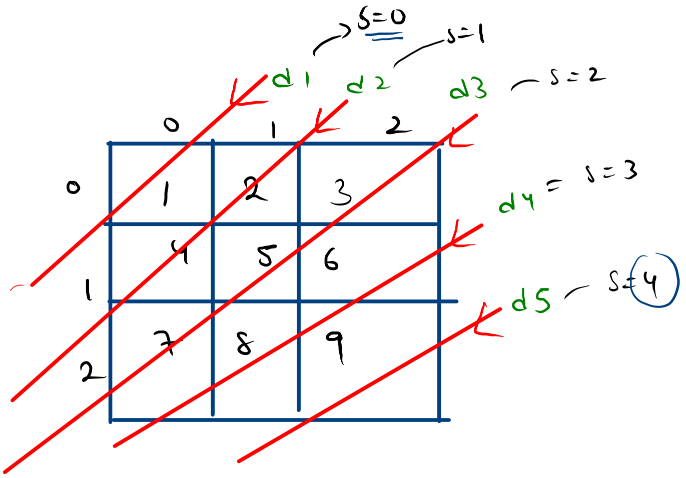
Sample Output 0

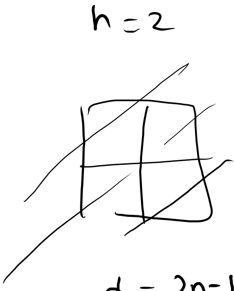
```
1 2 4 3 5 7 6 8 9
```

$n=3$

$d=2n-1$

$s=0 \ldots < d$





$d = 2n-1$

$= 4-1$

$= 3$

✓

0,0
0,1
1,0

0,2
1,1
2,0

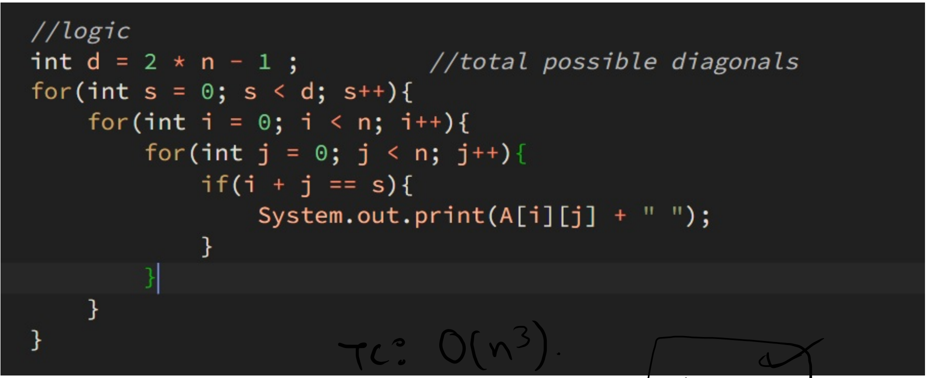
$\left.\begin{array}{l}1,2\\2,1\end{array}\right\} 3$

$2,2 \} 4$

$s=0 \Rightarrow i+j==s.$

$s=1 \Rightarrow i+j==s$

$s=2 \Rightarrow i+j==s$



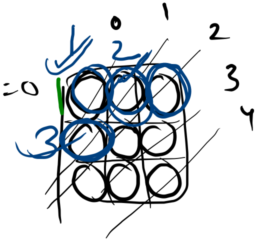
```
//logic
int d = 2 * n - 1 ;            //total possible diagonals
for(int s = 0; s < d; s++){
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            if(i + j == s){
                System.out.print(A[i][j] + " ");
            }
        }
    }
}
```

TC: $O(n^3)$.

$n + n - 1$
$r \quad c \quad -1$

$d = 2n-1 = 5$

0,0
$= 0+0=0$



$\boxed{s=0}$

$s=1$

$0+0=0$

$cs=0$

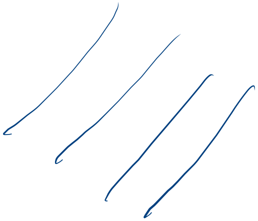
$0+1=1$

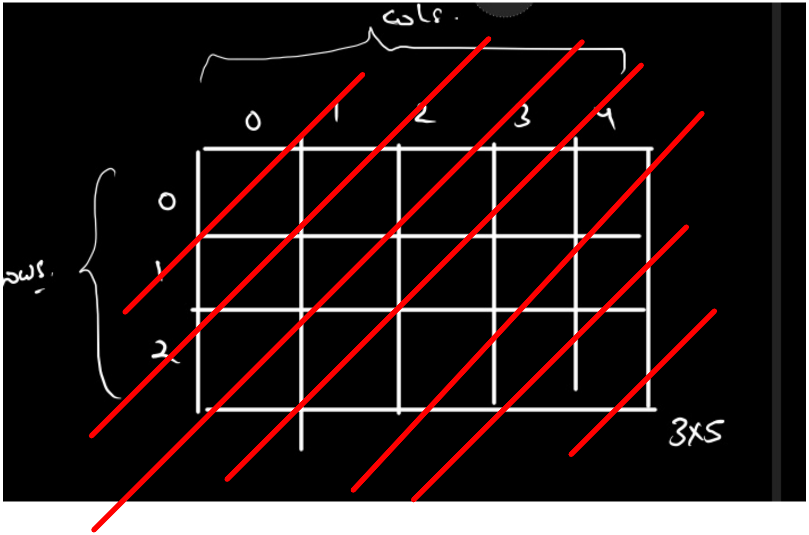
$cs=1$

$0+2=\underline{2}$

$\boxed{0, 1, 2, 3, 4} < 5$

$1+0 = 1$



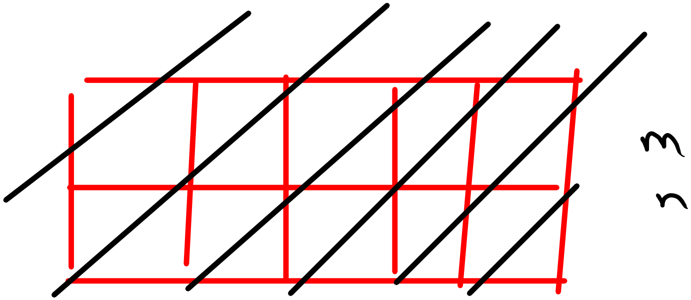


$m = 3$

$n = 5$

$\boxed{m + n - 1}$

$3 + 5 - 1 = 7$

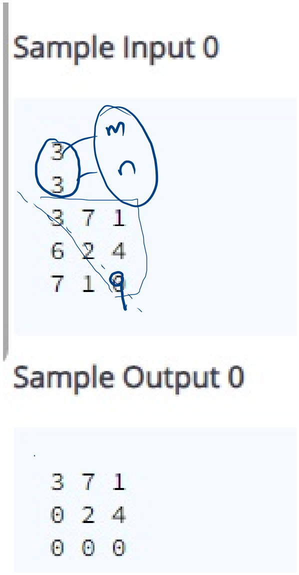


$m = 2$

$n = 5$

$\boxed{m + n - 1}$

Print Upper triangular matrix 1

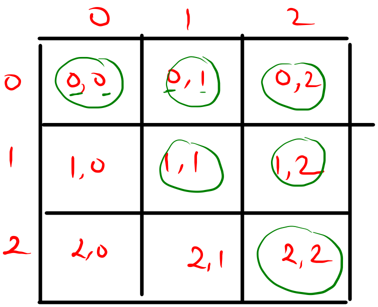


Sample Input 0

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

Sample Output 0

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



	0	1	2
0	0,0	0,1	0,2
1	1,0	1,1	1,2
2	2,0	2,1	2,2

$i \le j$

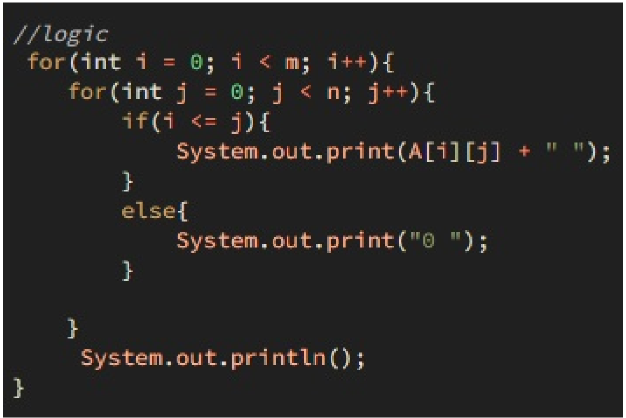
↳ A[i][j]

$i > j$

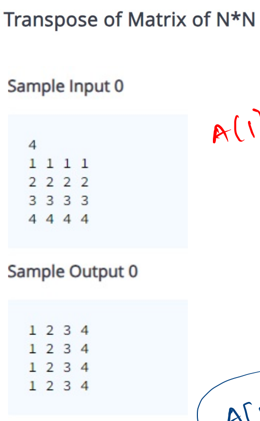
↳ 0

right_now

```
for (                    )
        for(             )
    {
    }
```



```
//logic
 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();
}
```



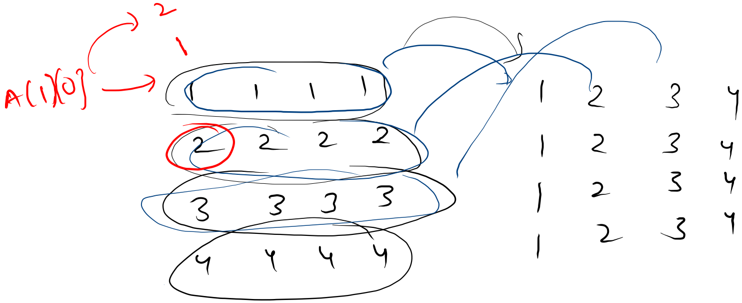
Transpose of Matrix of N*N

Sample Input 0

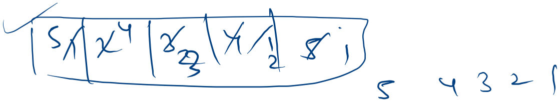
```
4
1 1 1 1
2 2 2 2
3 3 3 3
4 4 4 4
```

Sample Output 0

```
1 2 3 4
1 2 3 4
1 2 3 4
1 2 3 4
```



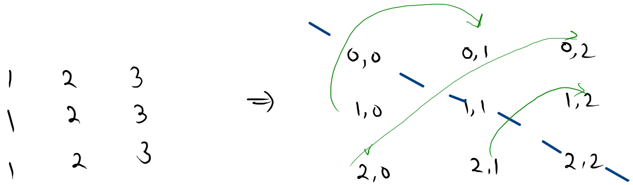
A[3] = 2

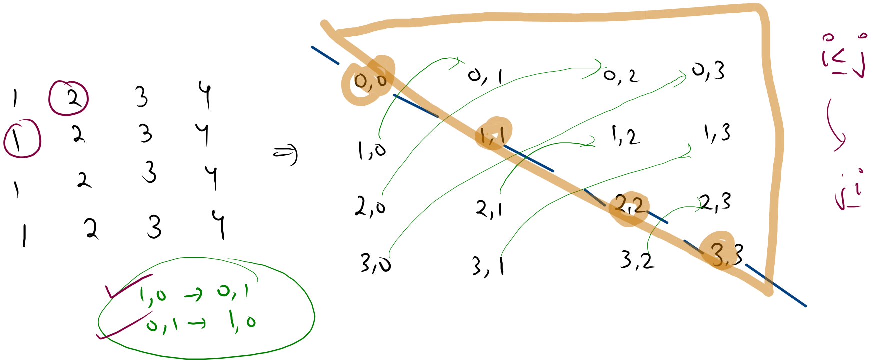


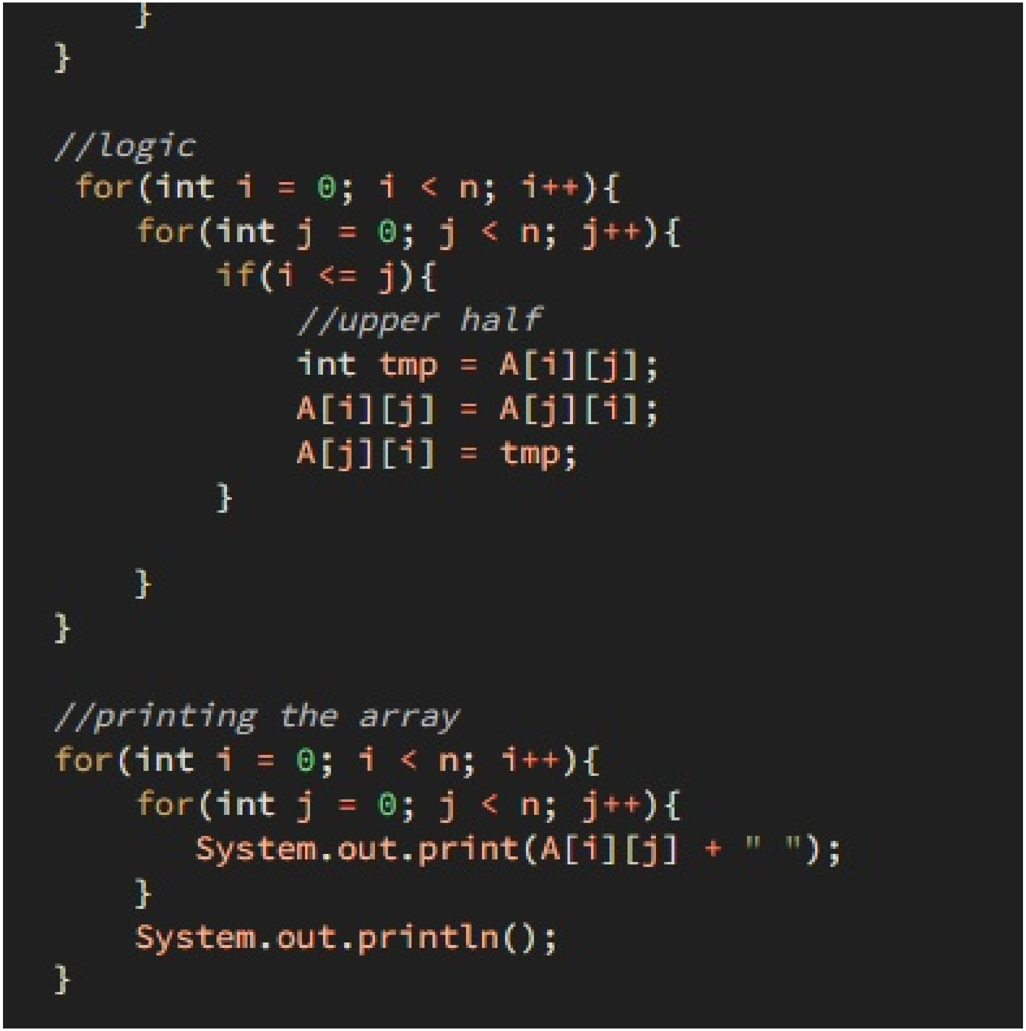
2

1 2 3 ~~4~~ 5

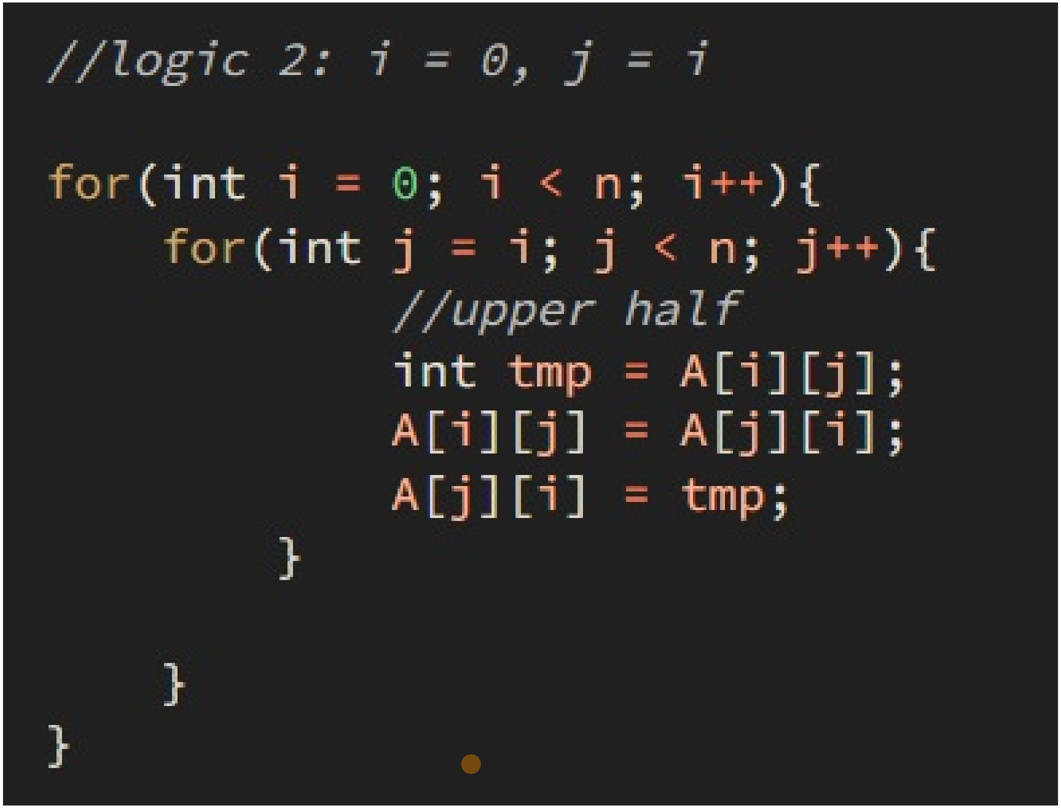
5 4 3 2 1







```
    }
}

//logic
 for(int i = 0; i < n; i++){
    for(int j = 0; j < n; j++){
        if(i <= j){
            //upper half
            int tmp = A[i][j];
            A[i][j] = A[j][i];
            A[j][i] = tmp;
        }

    }
}

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



```
//logic 2: i = 0, j = i

for(int i = 0; i < n; i++){
    for(int j = i; j < n; j++){
            //upper half
            int tmp = A[i][j];
            A[i][j] = A[j][i];
            A[j][i] = tmp;
        }


    }
}
```

Rotate The Matrix by 90 Degree

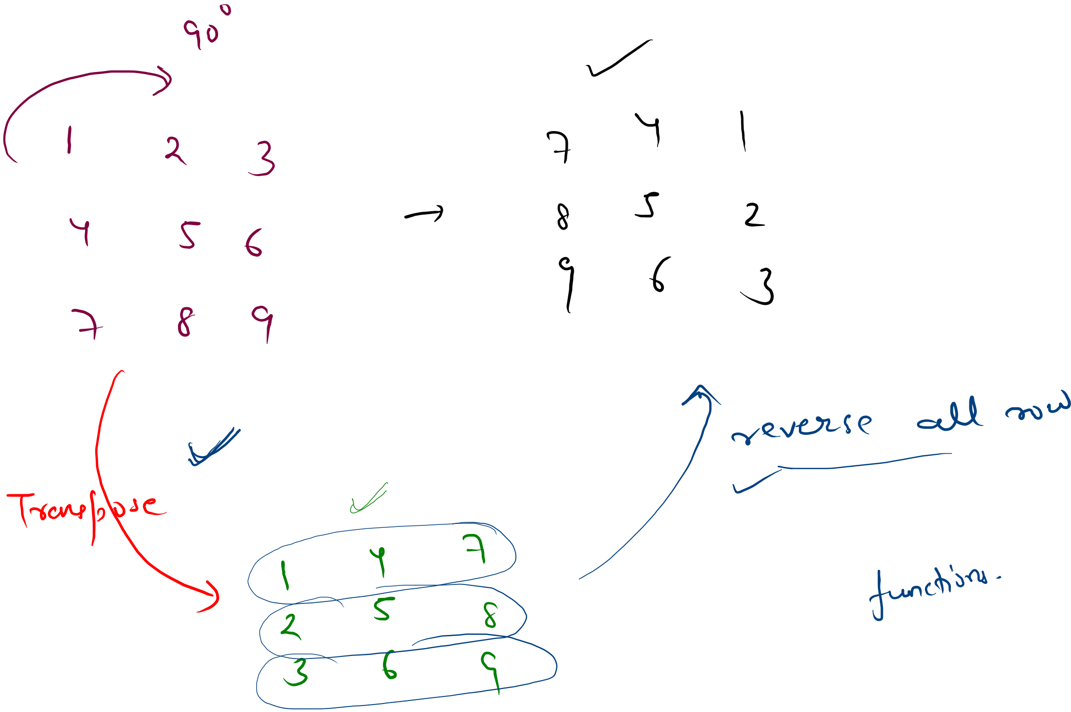
Sample Input 0

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

Sample Output 0

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

✓

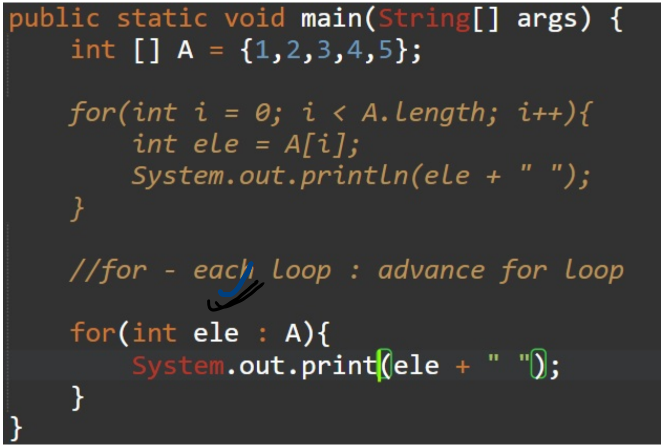


$$\begin{matrix}1&2&3\\4&5&6\\7&8&9\end{matrix} \rightarrow \begin{matrix}7&4&1\\8&5&2\\9&6&3\end{matrix}$$

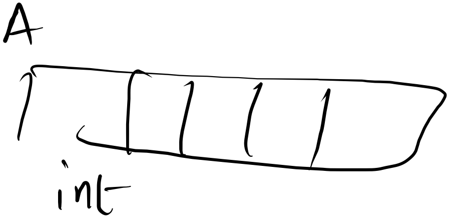
$$\begin{matrix}1&4&7\\2&5&8\\3&6&9\end{matrix}$$

for - each → loop → 1D

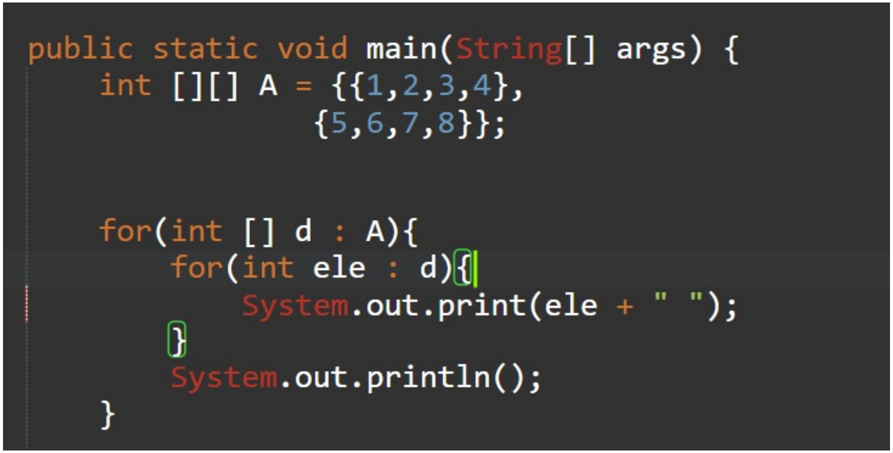
↳ no - control over index



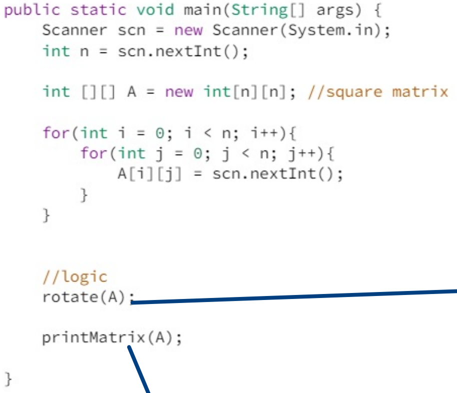
```
public static void main(String[] args) {
    int [] A = {1,2,3,4,5};

    for(int i = 0; i < A.length; i++){
        int ele = A[i];
        System.out.println(ele + " ");
    }

    //for - each loop : advance for loop

    for(int ele : A){
        System.out.print(ele + " ");
    }
}
```



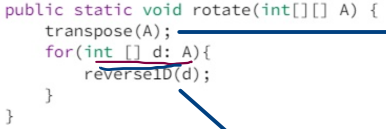
for - each over 2D Array.



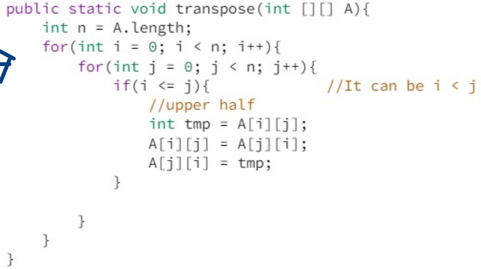
```
public static void main(String[] args) {
    int [][] A = {{1,2,3,4},
                {5,6,7,8}};


    for(int [] d : A){
        for(int ele : d){
            System.out.print(ele + " ");
        }
        System.out.println();
    }
```



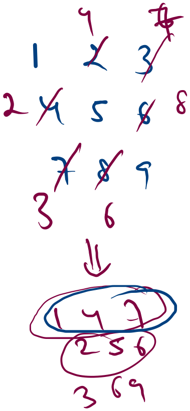
```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();

    int [][] A = new int[n][n]; //square matrix

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


    //logic
    rotate(A);

    printMatrix(A);

}
```

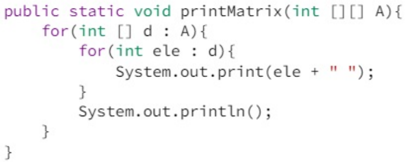


```java
public static void rotate(int[][] A) {
    transpose(A);
    for(int [] d: A){
        reverse1D(d);
    }
}
```

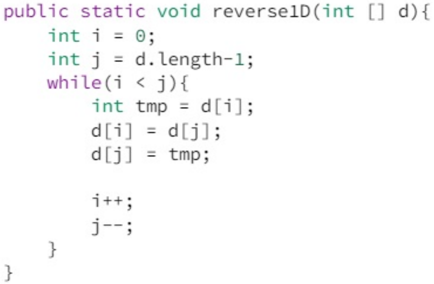


```java
public static void transpose(int [][] A){
    int n = A.length;
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            if(i <= j){                    //It can be i < j
                //upper half
                int tmp = A[i][j];
                A[i][j] = A[j][i];
                A[j][i] = tmp;
            }

        }
    }
}
```



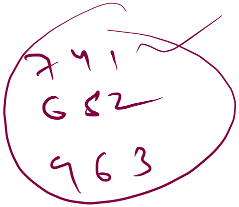


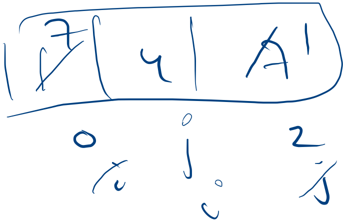
```java
public static void printMatrix(int [][] A){
    for(int [] d : A){
        for(int ele : d){
            System.out.print(ele + " ");
        }
        System.out.println();
    }
}
```

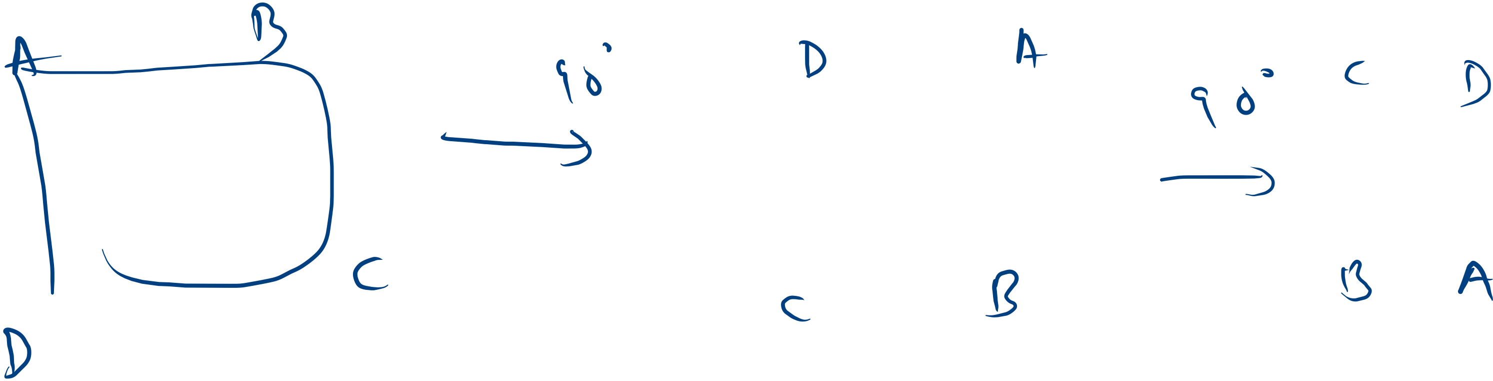


```java
public static void reverse1D(int [] d){
    int i = 0;
    int j = d.length-1;
    while(i < j){
        int tmp = d[i];
        d[i] = d[j];
        d[j] = tmp;

        i++;
        j--;
    }
}
```

7 4 1
8 5 2
9 6 3






A
B
C
D
90°
D
A
C
B
90°
C
D
B
A