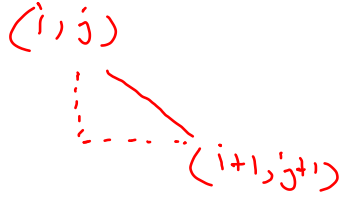
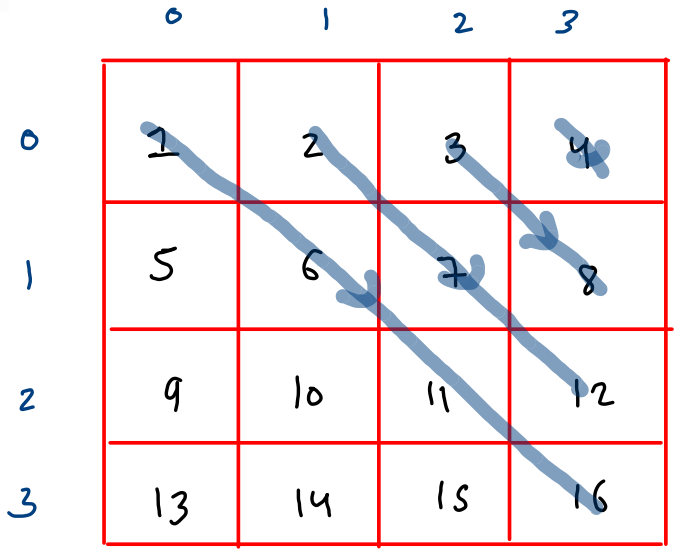
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16



1->6->11->16->2->7->12->3->8->4



d	st(i)	st(j)
0	0	0
1	0	1
2	0	2
3	0	3

d^{th}

$st(i) = 0$

$st(j) = d$

```java
public static void diagonalTraversal(int[][]mat) {

    //to select diagonal
    for(int d=0;d < mat.length;d++) {
        System.out.print(d + " -> ");
        for(int i = 0,j = d; j < mat.length ; i++,j++) {
            System.out.print(mat[i][j] + " ");
        }
        System.out.println();
    }
}
```

0 -> 11 22 33 44

1 -> 12 23 34

2 -> 13 24

3 -> 14

0 -> 11 22 33 44

1 -> 12 23 34

2 -> 13 24

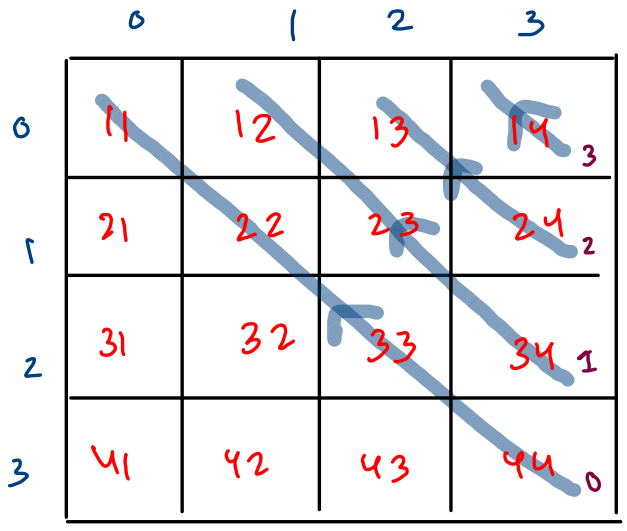
3 -> 14

d = 4 (0 to 3)

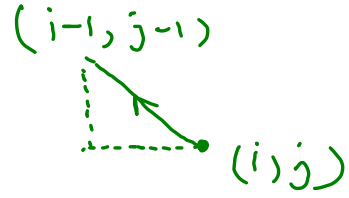
i = ~~0~~ 1

j = ~~3~~ 4

	0	1	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34
3	41	42	43	44




```
for( int d = 0 ; d < n ; d++) {
    for (i = n-d-1 , j = n-1 ; i >= 0 ; i--, j--) {
        Syso (mat [i] [j]) ;
    }
}
```



d	st(i)	st(j)
0	3	3
1	2	3
2	1	3
3	0	3

d^{th}

$st(i) = n-d-1$

$st(j) = n-1$

1. You are given a square matrix of size 'n'. You are given n*n elements of the square matrix.
2. You are required to find the saddle price of the given matrix and print the saddle price.
3. The saddle price is defined as the least price in the row but the maximum price in the column of the matrix.

16

	0	1	2	3
0	11	13	4	19
1	16	14	15	28
2	19	20	16	18
3	24	13	6	29

pot SP (min)	col
4	2
14	1
16	2

No saddle point

more than one SP

why only 1 sp ?

	0	1	2	3	4
0	a	b	c	d	e
1	f	g	h	i	j
2	k	l	m	n	o
3	p	q	r	s	t
4	u	v	w	x	y

n -> saddle point

$i < n < l$ — (1)

$\Rightarrow i < l$

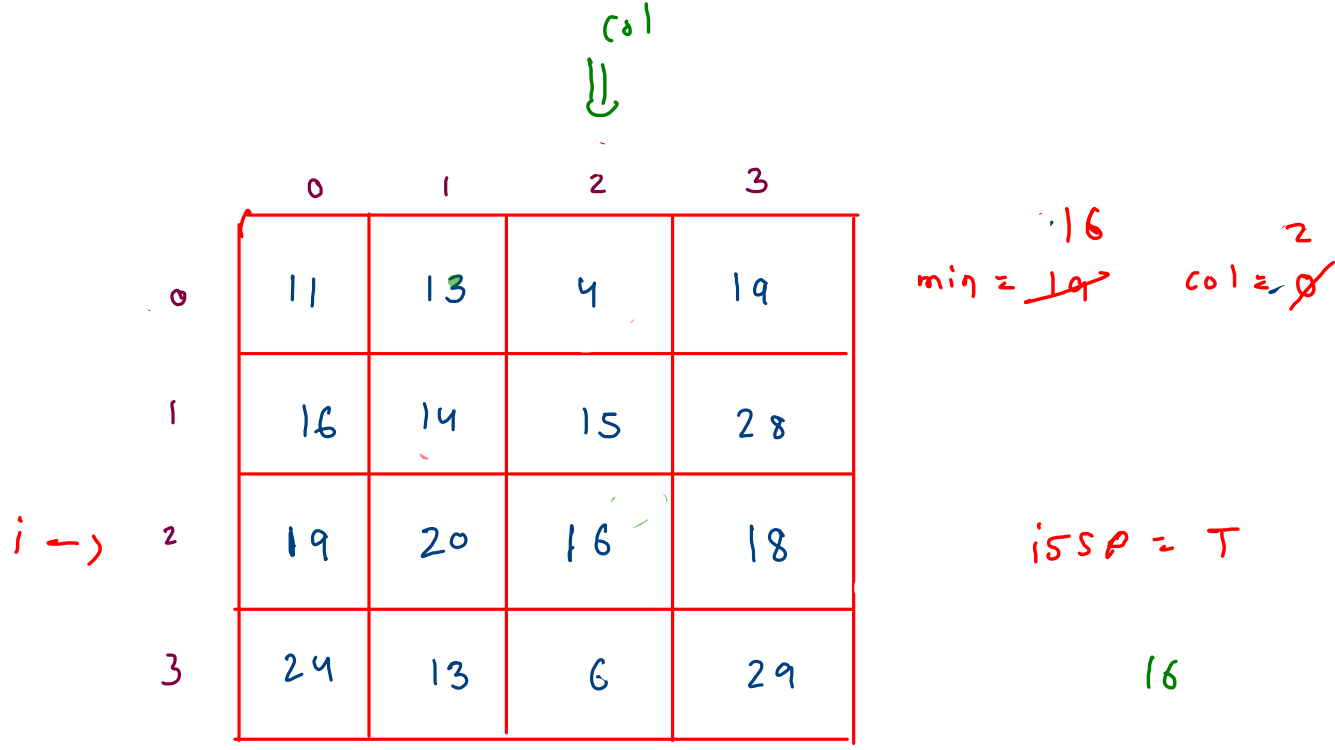
$u < g < i$ — (2)

$\Rightarrow u < l$

	0	1	2	3
0	11	13	~~4~~	19
1	20	~~14~~	15	28
2	15	20	~~11~~	18
3	24	13	~~6~~	29

no saddle point

```java
public static void saddlePoint(int[][]mat) {


    for(int i=0; i < mat.length;i++) {
        //find min elements and its col in ith row
        int min = mat[i][0] ;
        int col = 0;

        for(int j=1; j < mat[0].length;j++) {
            if(mat[i][j] < min) {
                min = mat[i][j];
                col = j;
            }
        }

        boolean isSp = true;

        //if min is the maximum value of col
        for(int r = 0; r < mat.length;r++) {
            if(mat[r][col] > min) {
                isSp = false;
                break;
            }
        }

        if(isSp == true) {
            System.out.println(min);
            return;
        }

    }

    System.out.println("Invalid input");

}
```



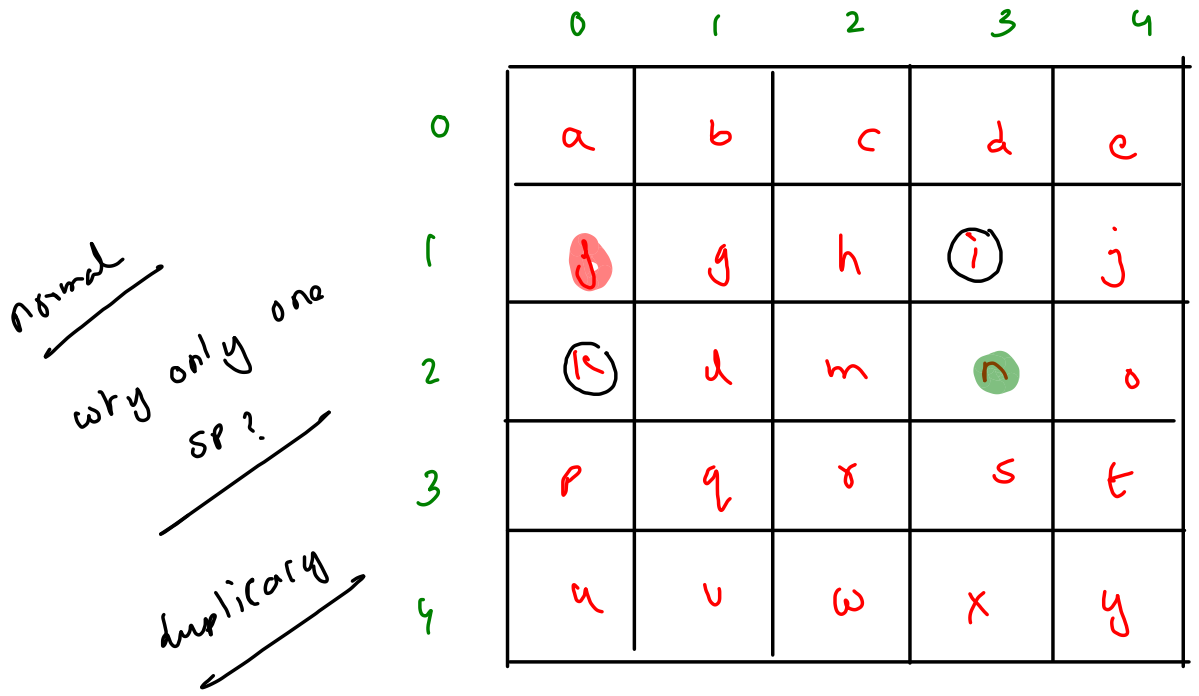
duplicates

	0	1	2	3
0	11	13	4	19
1	15	14	15	28
2	16	20	16	18
3	24	13	6	29

min = 16, c = 2,

col

0	2



$i < n < k$

$\Rightarrow \quad i < k$

$k < j < i$

$\Rightarrow \quad k < i$