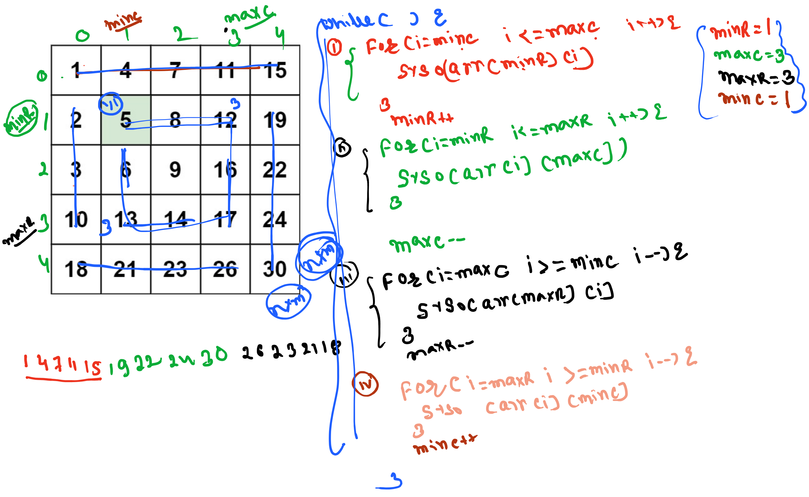
Spiral Print

07/05/25 7:05 PM

1	4	7	11	15
2	5	8	12	19
3	6	9	16	22
10	13	14	17	24
18	21	23	26	30

minR = 0
maxR = arr.length-1
minC = 0
maxC = arr[0].length-1

1 4 7 11 15 19 22 24 30 26 23 21 18 10 3 2 5 8 12 16 17 14 13 6 9




```
public static void Print(int[][] arr) {
    int minr = 0, minc = 0, maxr = arr.length - 1, maxc = arr[0].length - 1;
int te = arr.length * arr[0].length;// total_element
int c = 0;
while (c < te) {
    for (int i = minc; i <= maxc; i++) {
        System.out.print(arr[minr][i] + " ");
        c++;
    }
    minr++;
    for (int i = minr; i <= maxr; i++) {
        System.out.print(arr[i][maxc] + " ");
        c++;
    }
    maxc--;
    for (int i = maxc; i >= minc; i--) {
        System.out.print(arr[maxr][i] + " ");
        c++;
    }
    maxr--;
    for (int i = maxr; i >= minr; i--) {
        System.out.print(arr[i][minc] + " ");
        c++;
    }
    minc++;
}
}
```

1 2 3 4 8 12 11 10 9 5 6 7

Boyer voting algorithm

[0 1 2 3 4 5 6 7 8]
[2 7 3 7 3 7 7 5 7]

Given an array `nums` of size `n`, return *the majority element*.

The majority element is the element that appears more than `⌊n / 2⌋` times. You may assume that the majority element always exists in the array.

```
for(i=1 i<arr.len i++){
    if(arr[i]==ele){
        vote++
    }
    else{
        vote--
        if(vote==0){
            ele=arr[i]
            vote=1
        }
    }
}
```

7 7 7 7 3 3 3

7 2 7 7 3 7 7