↳ **Questions**

- ↳ Tic Tac Toe ✓
- ↳ Reverse by words ✓
- ↳ **Spiral matrix**

Sort Array By Parity

Given an integer array `nums[]` , move all the **even** integers at the beginning of the array followed by all the **odd** integers in non- decreasing order.

arr $\boxed{1}\ \boxed{5}\ \boxed{2}\ \boxed{3}\ \boxed{4}\ \boxed{2}\ \boxed{10}\ \boxed{15}\ \boxed{16}\ \boxed{10}\ \boxed{3}\ \boxed{7}\ \boxed{3}$

Step1

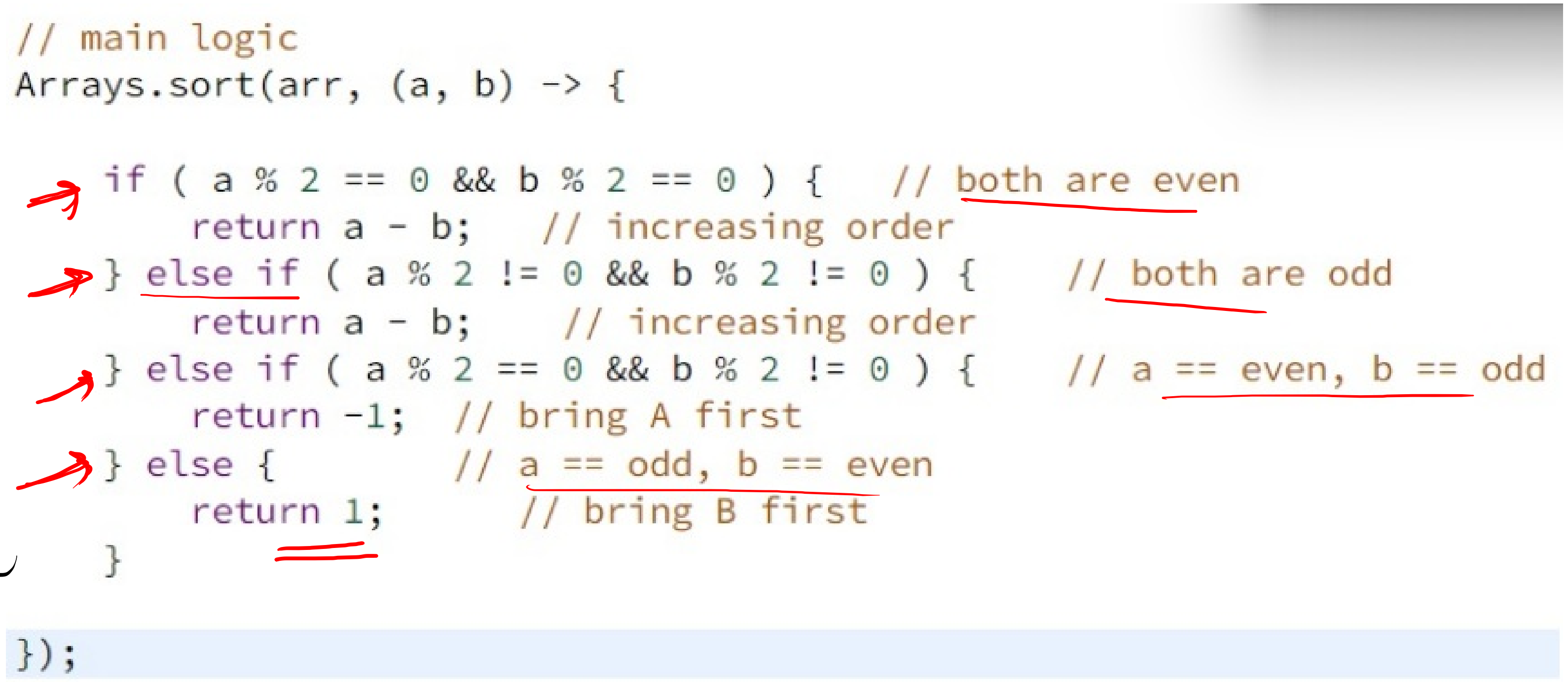
$\underbrace{2\quad 4\quad 2\quad 10\quad 16\quad 10}_{\text{even}}\quad \underbrace{1\quad 5\quad 3\quad 15\quad 3\quad 7\quad 3}_{\text{odd}}$

Step2

$\underbrace{2\quad 2\quad 4\quad 10\quad 16\quad 16}_{\text{non-decreasing order}}\quad \underbrace{1\quad 3\quad 3\quad 3\quad 5\quad 7\quad 15}_{\text{non-decreasing order}}$

Code

4 case :- both even, both odd, one even one odd } 4



```
// main logic
Arrays.sort(arr, (a, b) -> {

    if ( a % 2 == 0 && b % 2 == 0 ) {    // both are even
        return a - b;    // increasing order
    } else if ( a % 2 != 0 && b % 2 != 0 ) {     // both are odd
        return a - b;     // increasing order
    } else if ( a % 2 == 0 && b % 2 != 0 ) {     // a == even, b == odd
        return -1;   // bring A first
    } else {           // a == odd, b == even
        return 1;        // bring B first
    }

});
```

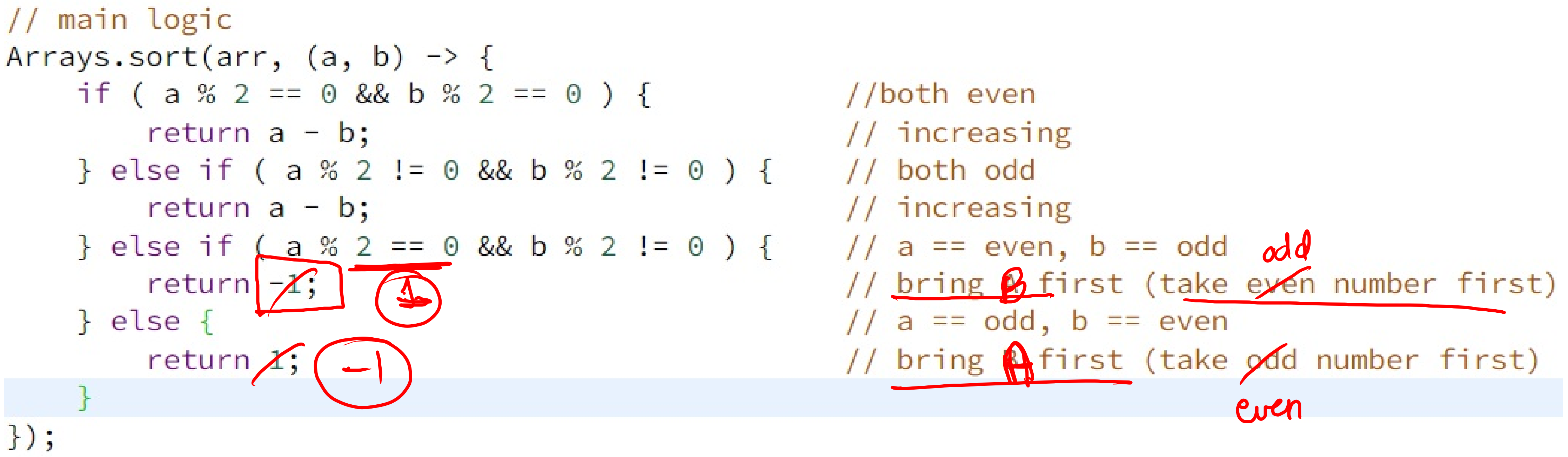
Case III & IV

1 3 5 3 4 2 4 6 8

Case II

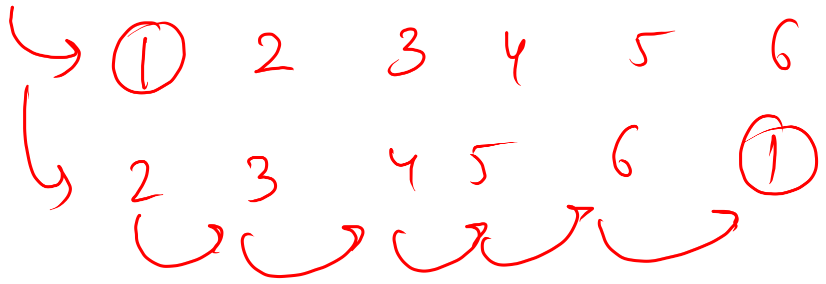
Case I

Ques If we want to sort array in increasing order again but take odd values first and the even values



```
// main logic
Arrays.sort(arr, (a, b) -> {
    if ( a % 2 == 0 && b % 2 == 0 ) {          //both even
        return a - b;                           // increasing
    } else if ( a % 2 != 0 && b % 2 != 0 ) {   // both odd
        return a - b;                           // increasing
    } else if ( a % 2 == 0 && b % 2 != 0 ) {   // a == even, b == odd
        return ~~-1~~; 1                        // bring ~~A~~ B first (take ~~even~~ odd number first)
    } else {                                    // a == odd, b == even
        return ~~1~~; -1                        // bring ~~B~~ A first (take ~~odd~~ even number first)
    }
});
```

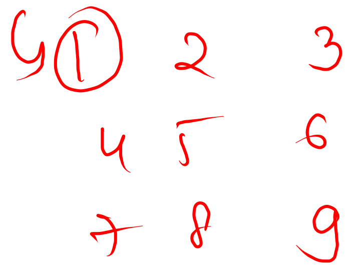
Ques find 1 in 1D array



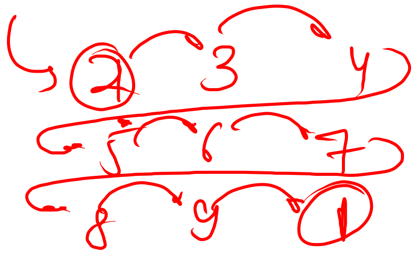
best $O(1) \Leftarrow$

worst $O(N) \Leftarrow$

Ques find 1 in 2D array



best $O(1) \Leftarrow$



worst case $O(N^2) \Leftarrow$

```
int n= scn.nextInt(); //5
                      //5000
                      //5M

int[] arr = new int[n];
        SC. // O(n)

int[][] arr1 = new int[n][n];
        SC. // O(n^2)
```

Spiral Matrix 44

matrix [sr] [i]

sc

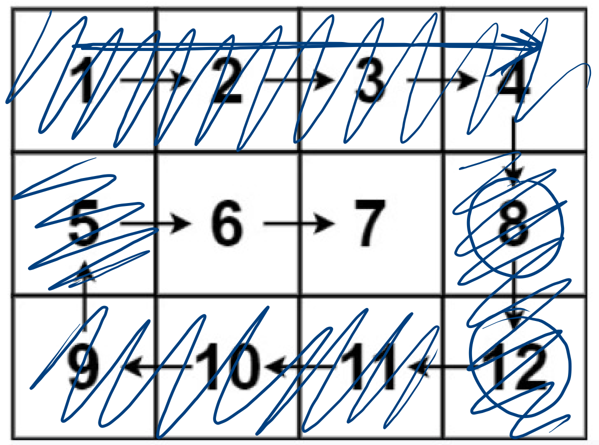
cc

matrix

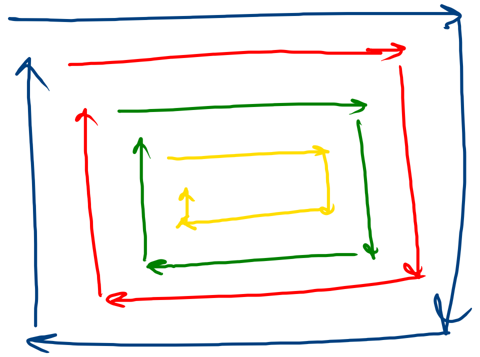
sr

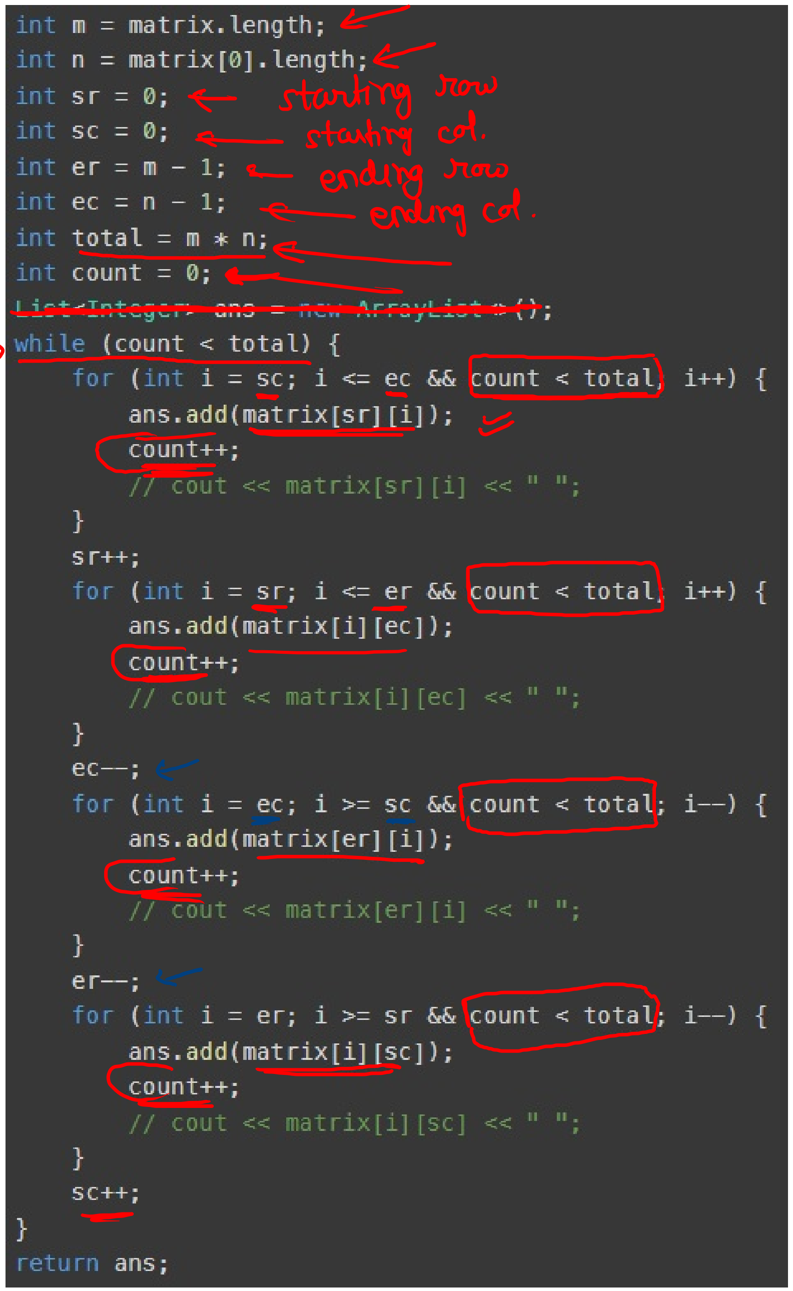
er

Example 2:



```
Input: matrix = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]
Output: [1,2,3,4,8,12,11,10,9,5,6,7]
```





```
int m = matrix.length;
int n = matrix[0].length;
int sr = 0;
int sc = 0;
int er = m - 1;
int ec = n - 1;
int total = m * n;
int count = 0;
~~List<Integer> ans = new ArrayList<>();~~
while (count < total) {
    for (int i = sc; i <= ec && count < total; i++) {
        ans.add(matrix[sr][i]);
        count++;
        // cout << matrix[sr][i] << " ";
    }
    sr++;
    for (int i = sr; i <= er && count < total; i++) {
        ans.add(matrix[i][ec]);
        count++;
        // cout << matrix[i][ec] << " ";
    }
    ec--;
    for (int i = ec; i >= sc && count < total; i--) {
        ans.add(matrix[er][i]);
        count++;
        // cout << matrix[er][i] << " ";
    }
    er--;
    for (int i = er; i >= sr && count < total; i--) {
        ans.add(matrix[i][sc]);
        count++;
        // cout << matrix[i][sc] << " ";
    }
    sc++;
}
return ans;
```

starting row

starting col.

ending row

ending col.

```
Public static int[] fun(______){

        return ans;

    }

main() {
    int[] ans = fun(______);
    syso(ans[0]);        syso(ans[1]);

}
```