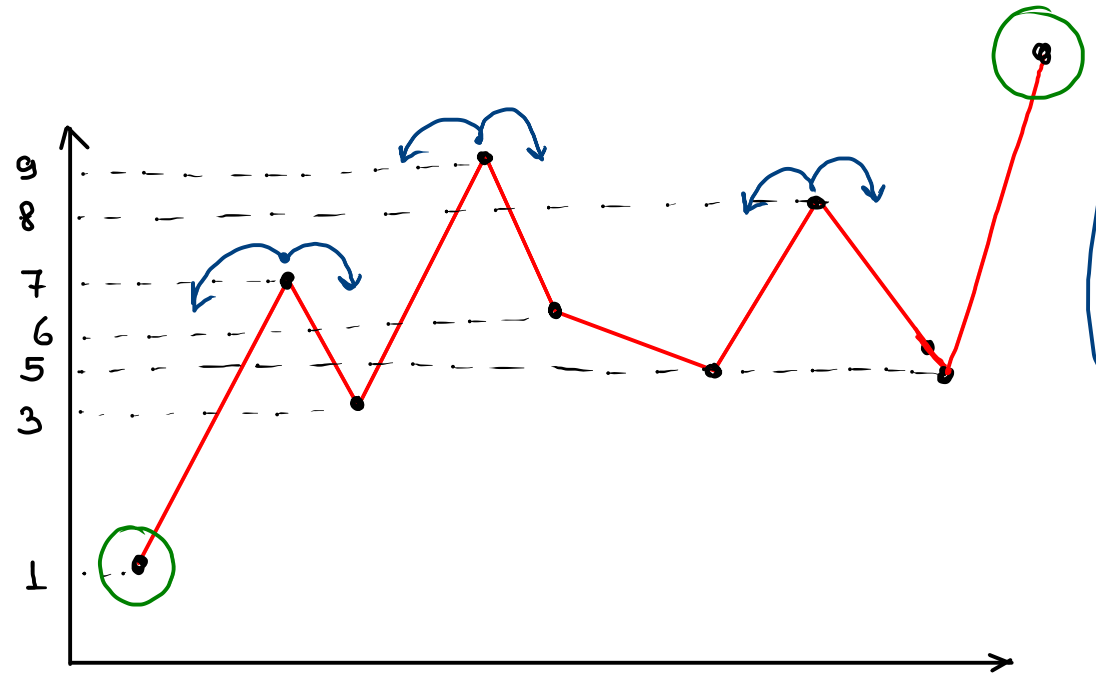
Que) arrange the element where all odd values should appear first followed by even values & odd values are arranged in ↑ing order based on square values & even values are arranged in ↓ing order based on cube values:

	a	b	return
1)	even	even	$b^3 - a^3$
2)	odd	odd	$a^2 - b^2$
3)	even	odd	+1
4)	odd	even	-1

Peak Elements



left < curr

&&

right < curr

peak = [7, 9, 8]

arr [1, 7, 3, 9, 6, 5, 8, 5]

0 1 2 3 4 5 6 7

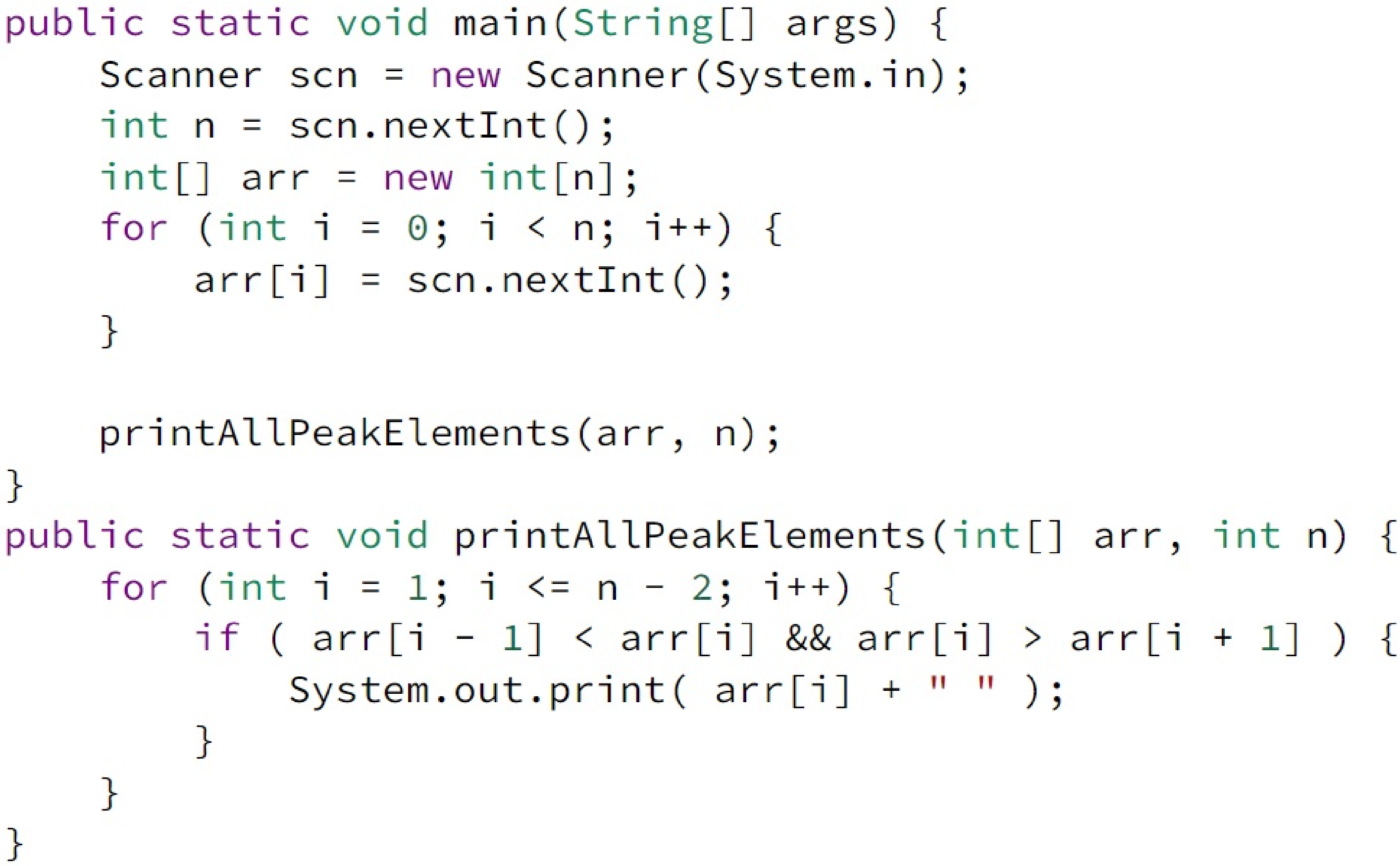
sije = n

opeation = n-2

T.C = O(n)

where n is sije of array

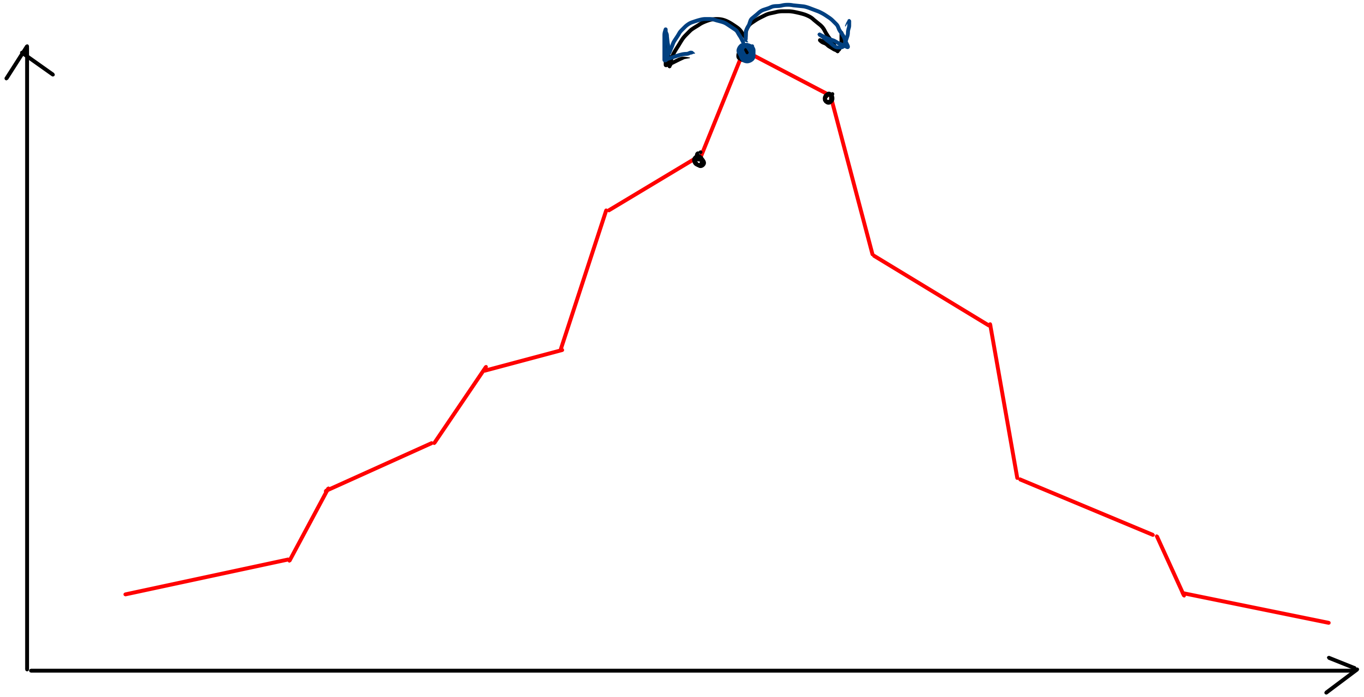
Code



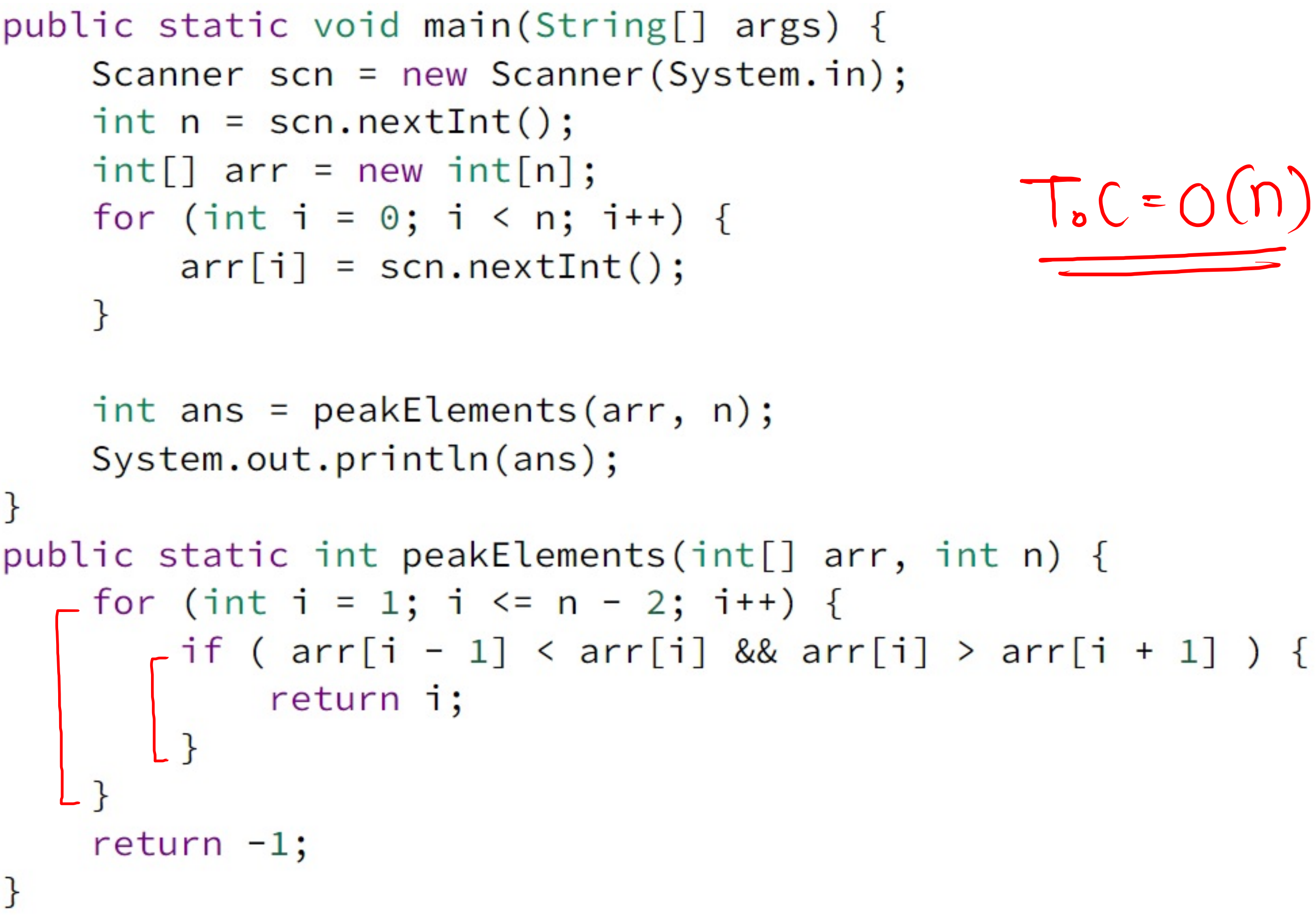
```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    printAllPeakElements(arr, n);
}
public static void printAllPeakElements(int[] arr, int n) {
    for (int i = 1; i <= n - 2; i++) {
        if ( arr[i - 1] < arr[i] && arr[i] > arr[i + 1] ) {
            System.out.print( arr[i] + " " );
        }
    }
}
```

Peak Index in a Mountain Array 2



Note:- peak element :- an element which is greater than its left and right elements code



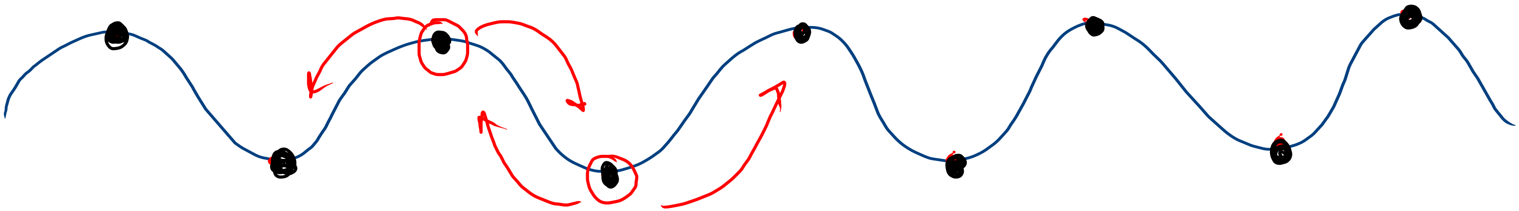
```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }

    int ans = peakElements(arr, n);
    System.out.println(ans);
}
public static int peakElements(int[] arr, int n) {
    for (int i = 1; i <= n - 2; i++) {
        if ( arr[i - 1] < arr[i] && arr[i] > arr[i + 1] ) {
            return i;
        }
    }
    return -1;
}
```

T.C = O(n)

Sort an array in wave form 1 Imp

$arr = [\underset{0}{8}, \underset{1}{3}, \underset{2}{5}, \underset{3}{2}, \underset{4}{1}, \underset{5}{4}, \underset{6}{5}, \underset{7}{6}]$



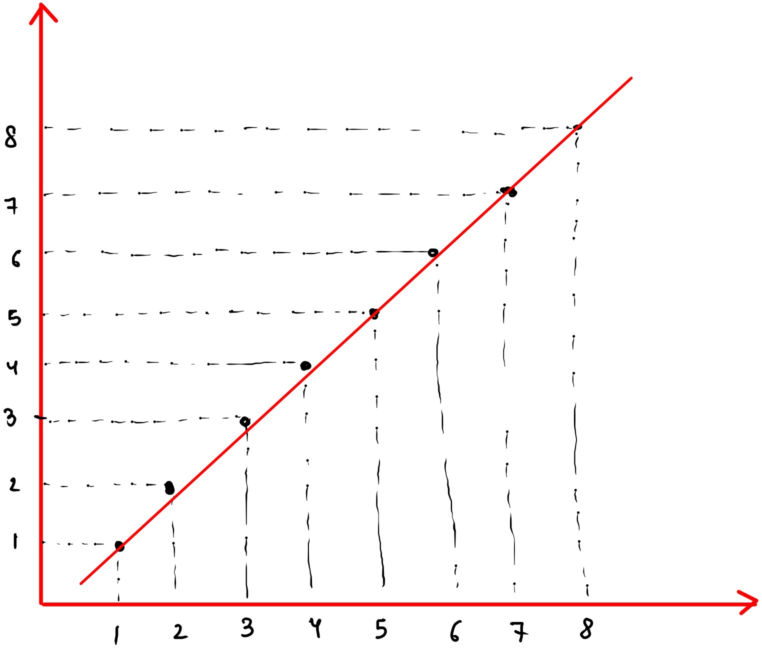
ans = 5 >= 3 <= 4 >= 2 <= 8 >= 1 <= 6 >= 5

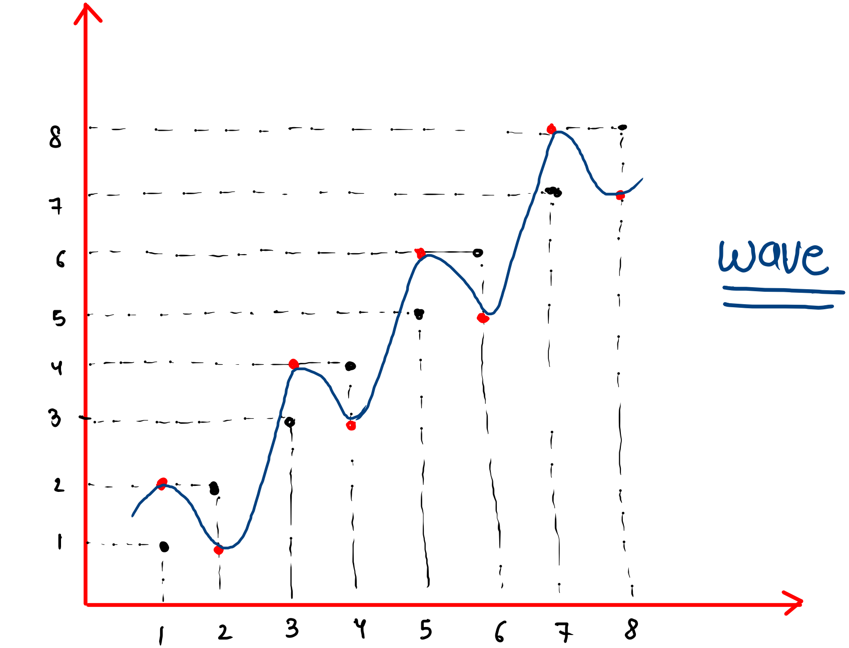
ans = $\underbrace{1 \quad 2} \quad \underbrace{3 \quad 4} \quad \underbrace{5 \quad 5} \quad \underbrace{6 \quad 8}$

arr = [5, 3, 1, 7, 6, 8, 2, 4]

1) Sort array

2) swap every alternate pair

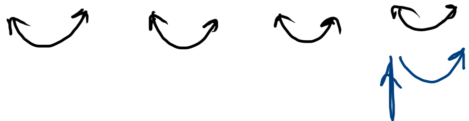




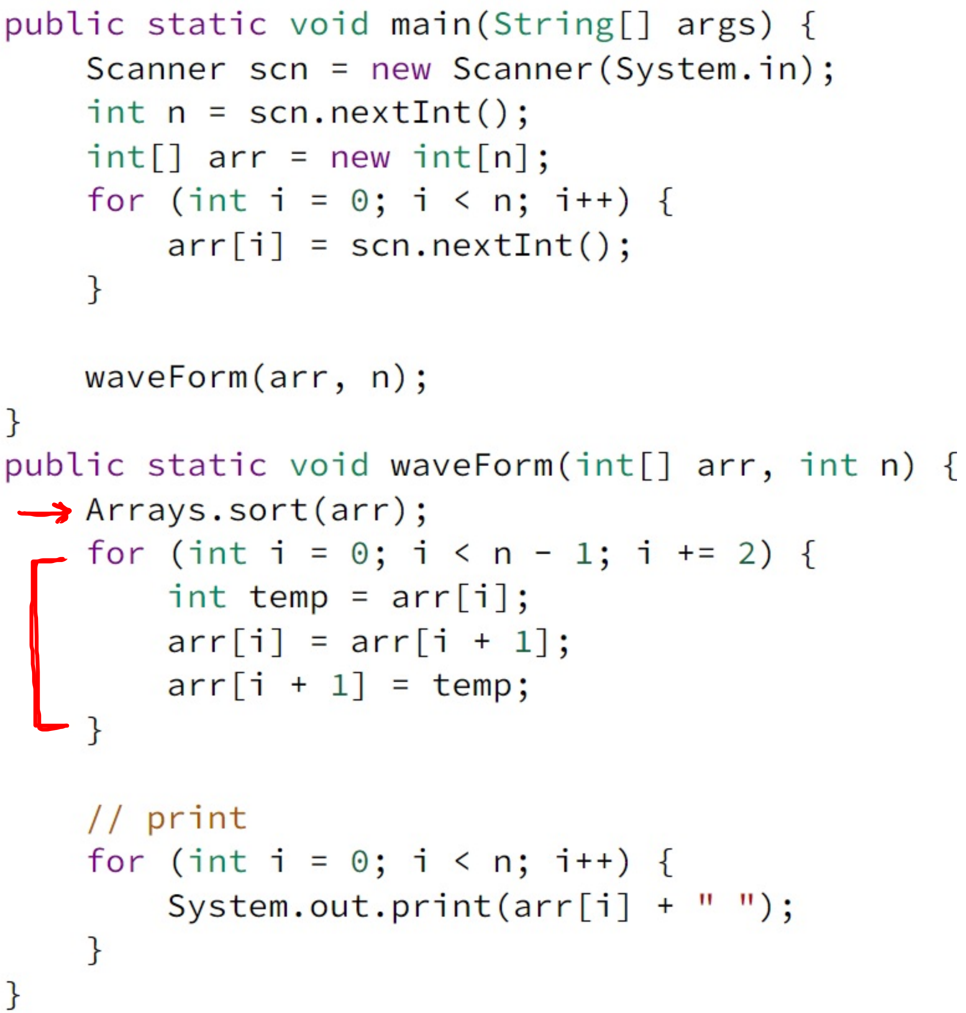
arr [1 2 3 4 5 6 7 8]

$\overset{0}{2}$ $\overset{1}{1}$ $\overset{2}{4}$ $\overset{3}{3}$

arr [2 1 4 3 6 5 8 7]



Code



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

    waveForm(arr, n);
}
public static void waveForm(int[] arr, int n) {
    Arrays.sort(arr);
    for (int i = 0; i < n - 1; i += 2) {
        int temp = arr[i];
        arr[i] = arr[i + 1];
        arr[i + 1] = temp;
    }

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

sort = $n\log(n)$

loop = $\frac{n}{2}$

oper = $\frac{n}{2} + n\log(n)$

T.C = $O(n + n\log(n))$

$\simeq O(n\log(n))$

Minimum difference 7

arr =

0	1	2	3
9	4	1	7

k=2

pairs		diff
(9, 4)	=	5
(9, 1)	=	8
(9, 7)	=	2
(4, 1)	=	3
(4, 7)	=	3
(1, 7)	=	6

ans = 2

k=3

pairs		diff
(9, 4, 7)	=	5
(9, 4, 1)	=	8
(9, 1, 7)	=	8
(4, 1, 7)	=	6

ans = 5

$arr = [1, \underbrace{\underset{\uparrow}{4}, 7, \underset{\uparrow}{9}}]$ $\quad$ $\underline{\underline{K = 3}}$

$\underline{\underline{diff = \cancel{6}\ 5}}$

Ex:-

arr = [5, 3, 7, -2, -8, 19, 10, 4, 9, 12, 15, 13]

sort

$$arr \Rightarrow [-8, -2, 3, 4, 5, 7, 9, \underbrace{10, 12, 13, 15, 19}]$$

↑ smallest ↑ largest

K = 4

ans = ~~12~~ ~~7~~ 4