计算最接近的数题目描述：

给定一个数组 X 和正整数 K，请找出使表达式 X[i] - X[i + 1] - … - X[i + K - 1] 结果最接近于数组中位数的下标 i，如果有多个 i 满足条件，请返回最大的 i。

其中，数组中位数：长度为 N 的数组，按照元素的值大小升序排列后，下标为 N/2 元素的值

补充说明：

1. 数组 X 的元素均为正整数；

2. X 的长度 n 取值范围：2<= n <= 1000；

3. K 大于 0 且小于数组的大小；

4. i 的取值范围：0 <= i < 1000；

5.题目的排序数组 X[N]的中位数是 X[N/2]。

示例 1

输入：

```
[50,50,2,3],2
```

输出：

```
1
```

说明：

1、中位数为 50：[50,50,2,3]升序排序后变成[2,3,50,50]，中位数为下标 4/2=2 的元素 50；

2、计算结果为 1：X[50,50,2,3]根据题目计算 X[i] - … - X[i + K - 1]得出三个数 0（X[0]-X[1] = 50 - 50）、48（X[1]-X[2 = 50 -2）和 -1（X[2]-X[3] = 2 - 3），其中 48 最接近 50，因此返回下标 1。

```
import java.util.*;


public class Solution {
    /**
```

```
     * 语句转换
     * @param scores int 整型一维数组 分数
     * @param K int 整型
     * @return int 整型
     */
    public int findTheStartPosition (int[] scores, int K) {
        // write code here


        int[] copy = new int[scores.length];
        for(int i = 0; i < copy.length; i ++){
            copy[i] = scores[i];
        }

         // length - 1
        // x[i+ k - 2] -

        Arrays.sort(copy);
        int midVal = copy[copy.length/2];
        int targetIdx = 0;
        int targetVal = Math.abs(scores[0] - scores[1] - midVal);
        for(int i = 1 ; i < scores.length + K - 2 ; i++){
            if(Math.abs(scores[i] - scores[i+1] - midVal) <= targetVal){
                targetIdx = i;
                targetVal = Math.abs(scores[i] - scores[i+1] - midVal);
            }
        }
        return targetIdx;

    }
}
```