HW 02: Find K-Closest Element (Leetcode-658)

Example 1: Input: arr = [1,2,3,4,5], k = 4, x = 3 Output: [1,2,3,4]	**Example 2:** Input: arr = [1,2,3,4,5], k = 4, x = -1 Output: [1,2,3,4]

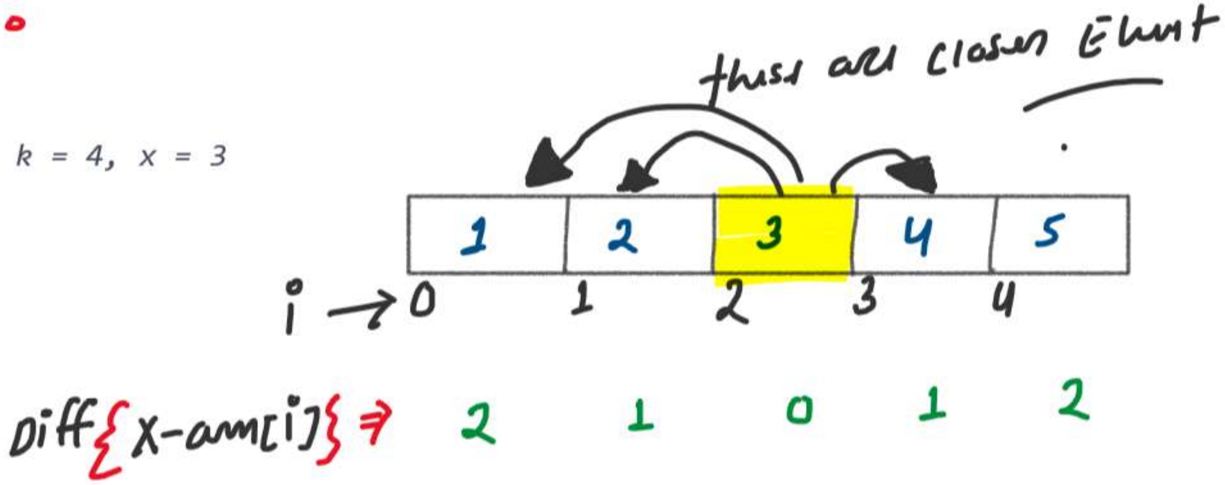
An integer **a** is closer to **x** than an integer **b** if:

- |a - x| < |b - x|, or
- |a - x| == |b - x| and a < b

Note

What is Closer Element to X ?

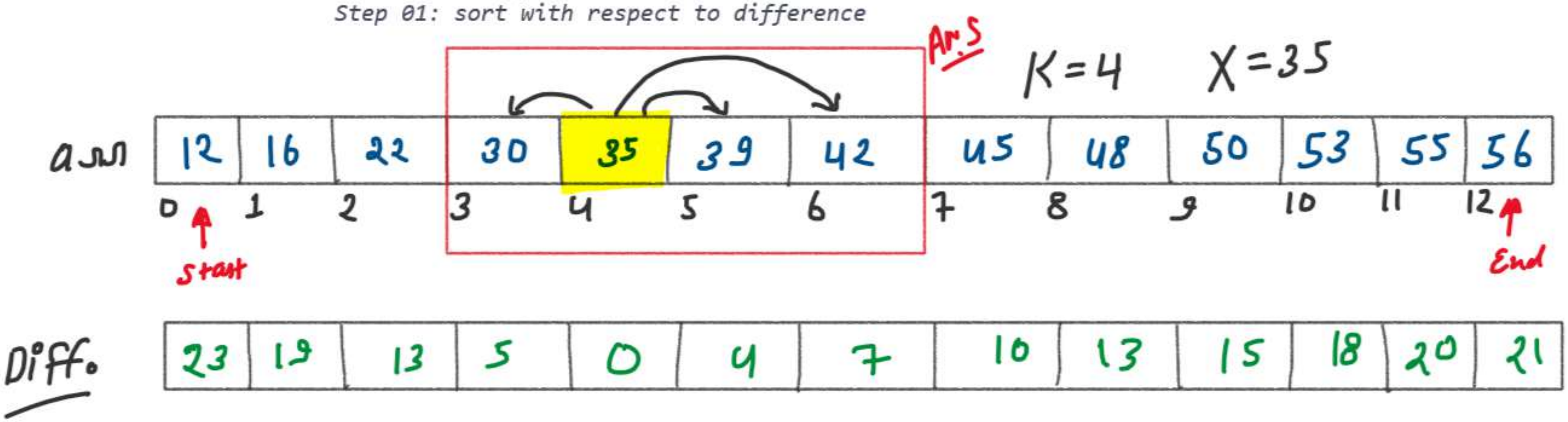
Example 1:
Input: arr = [1,2,3,4,5], k = 4, x = 3
Output: [1,2,3,4]



WHAT IS CLOSER ELEMENTS OF X?

1. X ka sabse jyada closer element konsa hoga jo uske sabse jyada nearly ya uske barabar hoga
2. I need only number of K elements jo X ke nearly hai like: 1 2 3 and 4

Two Pointer Approach
Step 01: sort with respect to difference



	0	1	2	3	4	5	6	7	8	9	10	11	12
arr	12	16	22	30	35	39	42	45	48	50	53	55	56
Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration:1   X=35   Start = 0   End = 12   a = arr[start]   b = arr[End]

= 12   = 56

① End - start >= K

12 - 0 >= 4

TRUE

② (|X - a| > |X - b|)

|35-12| > |35-56|

23 > 21

Start ++

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
		Start											End

Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration:2 X=35 Start =1 End = 12 a = arr[start] = 16 b = arr[End] = 56

① $End - start >= K$

$12 - 1 >= 4$

TRUE

② $(|X-a| < |X-b|)$

$|35-16| < |35-56|$

$19 < 21$

End--

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
		↑ Start										↑ End	

Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration:3   X=35   Start =1   End =11   a = arr[start]
= 16   b = arr[End]
= 55

①

End - Start >= K

11 - 1 >= 4

TRUE

②

$(|X-a| < |X-b|)$

$|35-16| < |35-55|$

$19 < 20$

End--

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
		↑ Start									↑ End		
Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration: 4   X=35   Start =1   End =10   a = arr[start] = 16   b = arr[End] = 53

① End - start $\geq$ K

10 - 1 $\geq$ 4

TRUE

② (|X - a| > |X - b|)

|35-16| > |35-53|

19 > 18

Start ++

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
			↑ Start								↑ End		
Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration: 5 X=35 Start = 2 End = 10 $a = arr[start]$ $= 22$ $b = arr[End]$ $= 53$

① $End - start \geq K$

$10 - 2 \geq 4$

TRUE

② $(|X - a| < |X - b|)$

$|35 - 22| < |35 - 53|$

$13 < 18$

End--

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
			Start							End			
Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration:6 X=35 start = 2 End = 9 a = arr[start] = 22 b = arr[End] = 50

①

End - start >= K

9 - 2 >= 4

TRUE

②

(|X - a| < |X - b|)

|35-22| < |35-50|

13 < 15

End--

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
			↑ Start						↑ End				
Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration: 7 X=35 start = 2 End = 8 a = arr[start] = 22 b = arr[End] = 48

①

End - start >= K

8 - 2 >= 4

TRUE

②

$(|x - a| == |x - b|)$

$|35-22| == |35-48|$

$13 == 13$

End-- → $(a < b)$

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
			↑ Start					↑ End					

Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration: 8   X=35   start = 2   End = 7   a = arr[start] = 22   b = arr[End] = 45

① $End - start >= K$

$7 - 2 >= 4$

TRUE

② $(|x-a| > |x-b|)$

$|35-22| > |35-45|$

$13 > 10$

start++

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
				Start				End					

Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration. 9 X=35 start = 3 End = 7 a = arr[start] b = arr[End]

= 30 = 45

① End - start >= K

7 - 3 >= 4

TRUE

② $(|x - a| < |x - b|)$

$|35 - 30| < |35 - 45|$

$5 < 10$

End--

arr	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
				↑ Start			↑ End						

Diff.	23	19	13	5	0	4	7	10	13	15	18	20	21

iteration-10 X=35 start = 3 End = 6 a = arr[start] = 30 b = arr[End] = 42

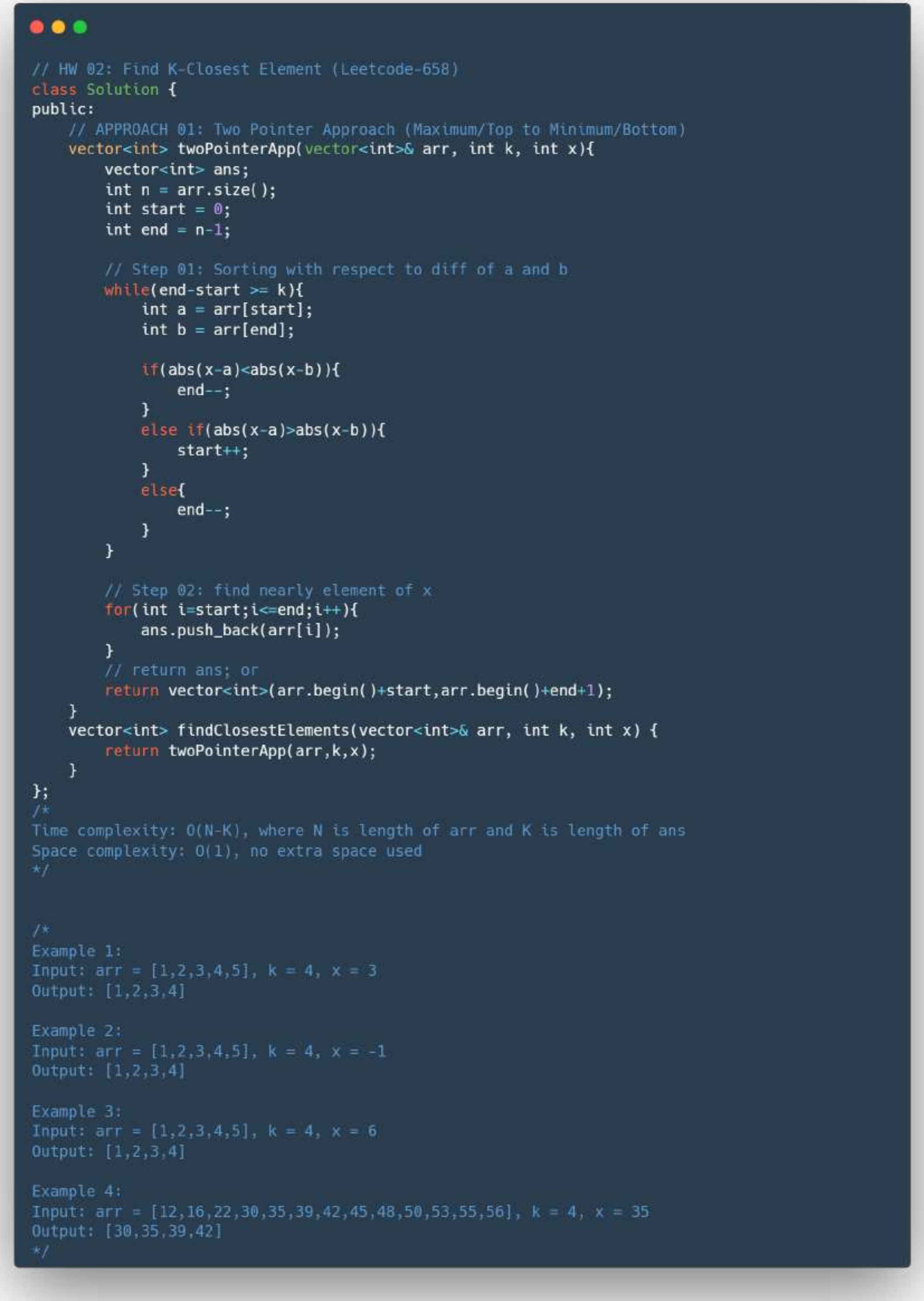
① End - start >= K

6 - 3 >= 4

FALSE ✗

30	35	39	42

Output



```
// HW 02: Find K-Closest Element (Leetcode-658)
class Solution {
public:
    // APPROACH 01: Two Pointer Approach (Maximum/Top to Minimum/Bottom)
    vector<int> twoPointerApp(vector<int>& arr, int k, int x){
        vector<int> ans;
        int n = arr.size();
        int start = 0;
        int end = n-1;

        // Step 01: Sorting with respect to diff of a and b
        while(end-start >= k){
            int a = arr[start];
            int b = arr[end];

            if(abs(x-a)<abs(x-b)){
                end--;
            }
            else if(abs(x-a)>abs(x-b)){
                start++;
            }
            else{
                end--;
            }
        }

        // Step 02: find nearly element of x
        for(int i=start;i<=end;i++){
            ans.push_back(arr[i]);
        }
        // return ans; or
        return vector<int>(arr.begin()+start,arr.begin()+end+1);
    }
    vector<int> findClosestElements(vector<int>& arr, int k, int x) {
        return twoPointerApp(arr,k,x);
    }
};
/*
Time complexity: O(N-K), where N is length of arr and K is length of ans
Space complexity: O(1), no extra space used
*/


/*
Example 1:
Input: arr = [1,2,3,4,5], k = 4, x = 3
Output: [1,2,3,4]

Example 2:
Input: arr = [1,2,3,4,5], k = 4, x = -1
Output: [1,2,3,4]

Example 3:
Input: arr = [1,2,3,4,5], k = 4, x = 6
Output: [1,2,3,4]

Example 4:
Input: arr = [12,16,22,30,35,39,42,45,48,50,53,55,56], k = 4, x = 35
Output: [30,35,39,42]
*/
```

HW 02: Find K-Closest Element (Leetcode-658)

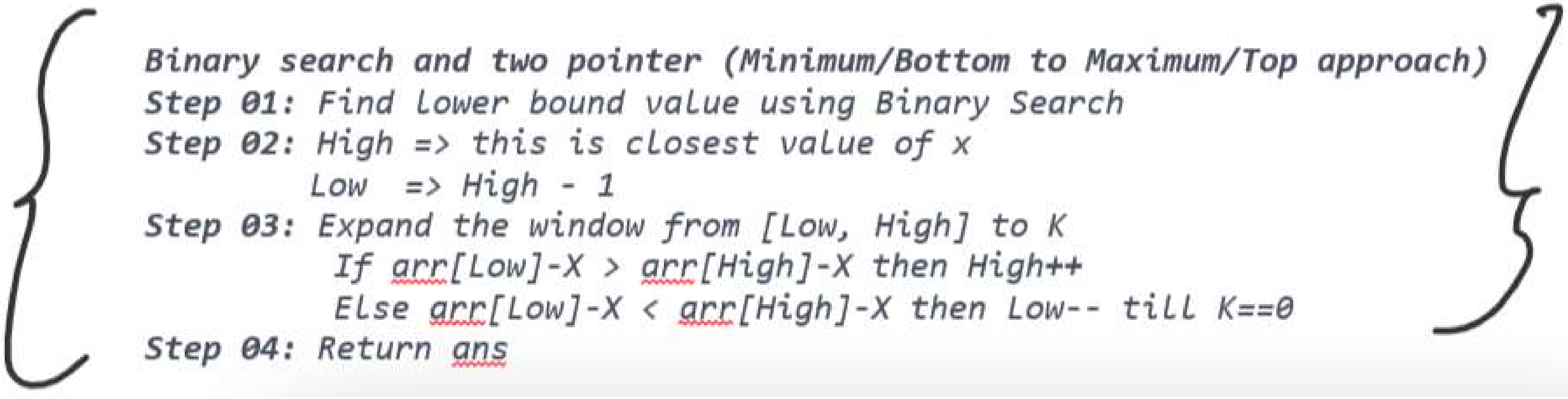
Ex:01

K = 4   X = 35

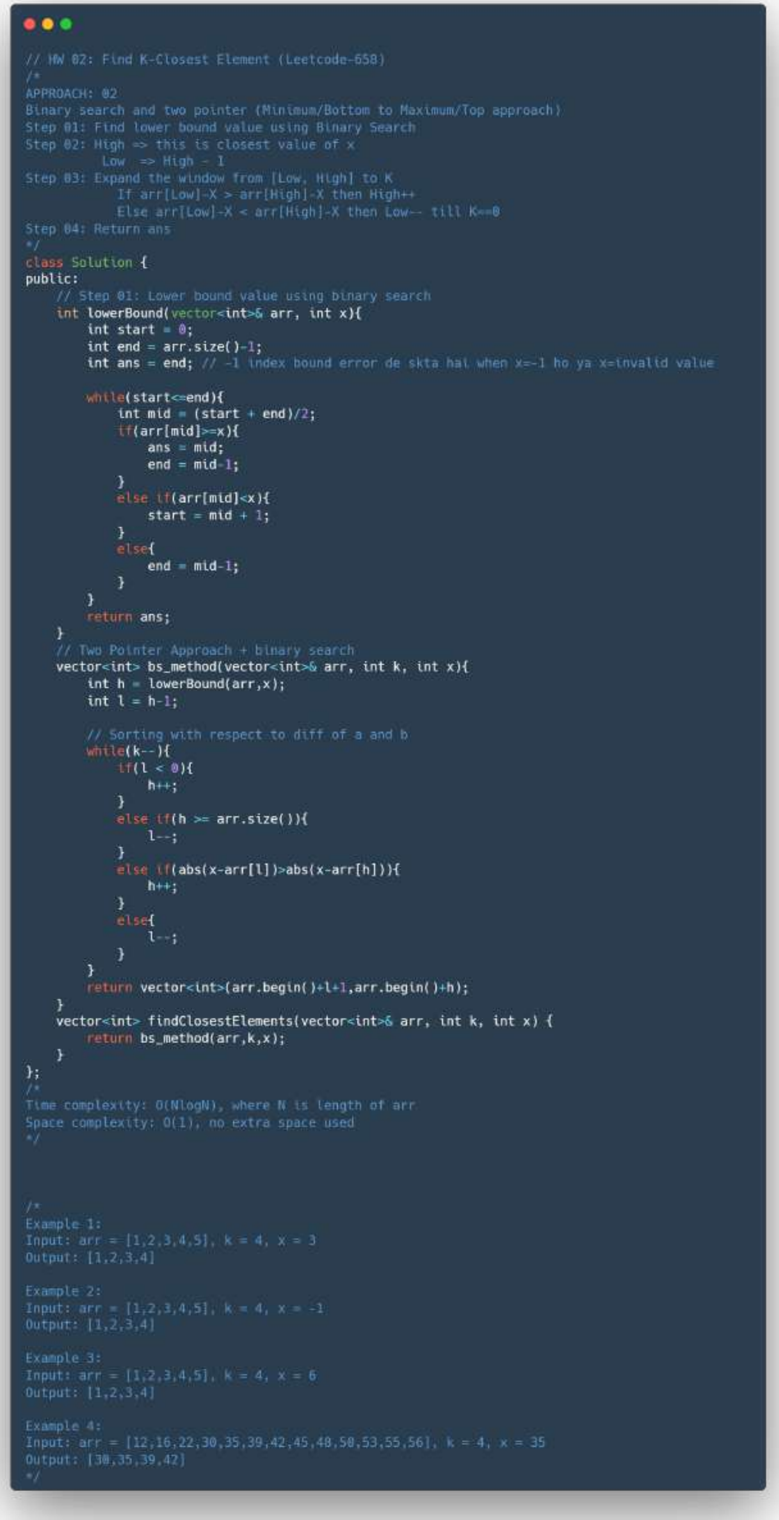
12	16	22	30	35	39	42	45	48	50	53	55	56

Output

	12	16	22	30	35	39	42	45	48	50	53	55	56
	0	1	2	3	4	5	6	7	8	9	10	11	12
|X- arr[i]| →	23	19	13	5	0	4	7	10	13	15	18	20	21



```
Binary search and two pointer (Minimum/Bottom to Maximum/Top approach)
Step 01: Find lower bound value using Binary Search
Step 02: High => this is closest value of x
         Low  => High - 1
Step 03: Expand the window from [Low, High] to K
          If arr[Low]-X > arr[High]-X then High++
          Else arr[Low]-X < arr[High]-X then Low-- till K==0
Step 04: Return ans
```



```
// HW 02: Find K-Closest Element (Leetcode-658)
/*
APPROACH: 02
Binary search and two pointer (Minimum/Bottom to Maximum/Top approach)
Step 01: Find lower bound value using Binary Search
Step 02: High => this is closest value of x
          Low  => High - 1
Step 03: Expand the window from [Low, High] to K
            If arr[Low]-X > arr[High]-X then High++
            Else arr[Low]-X < arr[High]-X then Low-- till K==0
Step 04: Return ans
*/
class Solution {
public:
    // Step 01: Lower bound value using binary search
    int lowerBound(vector<int>& arr, int x){
        int start = 0;
        int end = arr.size()-1;
        int ans = end; // -1 index bound error de skta hai when x=-1 ho ya x=invalid value

        while(start<=end){
            int mid = (start + end)/2;
            if(arr[mid]>=x){
                ans = mid;
                end = mid-1;
            }
            else if(arr[mid]<x){
                start = mid + 1;
            }
            else{
                end = mid-1;
            }
        }
        return ans;
    }
    // Two Pointer Approach + binary search
    vector<int> bs_method(vector<int>& arr, int k, int x){
        int h = lowerBound(arr,x);
        int l = h-1;

        // Sorting with respect to diff of a and b
        while(k--){
            if(l < 0){
                h++;
            }
            else if(h >= arr.size()){
                l--;
            }
            else if(abs(x-arr[l])>abs(x-arr[h])){
                h++;
            }
            else{
                l--;
            }
        }
        return vector<int>(arr.begin()+l+1,arr.begin()+h);
    }
    vector<int> findClosestElements(vector<int>& arr, int k, int x) {
        return bs_method(arr,k,x);
    }
};
/*
Time complexity: O(NlogN), where N is length of arr
Space complexity: O(1), no extra space used
*/



/*
Example 1:
Input: arr = [1,2,3,4,5], k = 4, x = 3
Output: [1,2,3,4]

Example 2:
Input: arr = [1,2,3,4,5], k = 4, x = -1
Output: [1,2,3,4]

Example 3:
Input: arr = [1,2,3,4,5], k = 4, x = 6
Output: [1,2,3,4]

Example 4:
Input: arr = [12,16,22,30,35,39,42,45,48,50,53,55,56], k = 4, x = 35
Output: [30,35,39,42]
*/
```