Mencari Teman Dekat

Problem | Submissions | Leaderboard | Discussions

Submitted 10 days ago • Score: 100.00

Status: Accepted

Success 0.06s

✓ Test Case #0 ✓ Test Case #1 ✓ Test Case #2
✓ Test Case #3 ✓ Test Case #4 ✓ Test Case #5
✓ Test Case #6 ✓ Test Case #7 ✓ Test Case #8
✓ Test Case #9

Submitted Code

Language: Python 3

Open in editor

```
# Input: n = length of array
n = int(input())

# Input: arr = sorted array
arr = list(map(int,input().split()))

# Input: k = jumlah elemen terdekat, target = target value
k,target = map(int,input().split())

# Fungsi untuk menemukan titik crossover menggunakan pencarian biner
def find_crossover(arr,target):
    # Initialize low and high indices
    low = 0
    high = len(arr) - 1

    # Loop sampai low <= high
    while low <= high:
        # cari mid index
        mid = (low + high) // 2

        # If target is found, return mid index
        if arr[mid] == target:
            return mid

        # If target is smaller than mid element, move high to left
        elif arr[mid] > target:
            high = mid - 1

        # If target is larger than mid element, move low to right
        else:
            low = mid + 1

```

```python
    # Return low index as crossover point
    return low

# Function to find k closest elements using two pointers
def find_k_closest(arr,k,target):
    # Initialize result list
    result = []

    # Find crossover point index
    cross_index = find_crossover(arr,target)

    # Initialize left and right pointers around crossover point
    left = cross_index - 1
    right = cross_index

    # Loop until result list has k elements or pointers are out of bounds
    while len(result) < k and (left >= 0 or right < len(arr)):

        # If left pointer is valid and right pointer is invalid or left element is closer than right element
        if left >= 0 and (right >= len(arr) or abs(arr[left] - target) <= abs(arr[right] - target)):

            # Add left element to result list at front
            result.insert(0,arr[left])

            # Move left pointer to left
            left -= 1

        # Else if right pointer is valid
        elif right < len(arr):

            # Add right element to result list at back
            result.append(arr[right])

            # Move right pointer to right
            right += 1

    # Return result list
    return result

# Print the output by calling the function
print(*find_k_closest(arr,k,target))
```