

Domain Winter Camp DAY-5

Name = DEVANSH TYAGI

Section/Group:-901-Kpit-B

UID = 22BCS10874

Problem 1

```
1  #include <iostream>
2  #include <vector>
3  using namespace std;
4
5  // Function to find the first occurrence of k in the array
6  int findFirstOccurrence(int k, const vector<int>& arr) {
7      for (size_t i = 0; i < arr.size(); ++i) {
8          if (arr[i] == k) {
9              return i + 1; // 1-based indexing
10         }
11     }
12     return -1; // Element not found
13 }
14
15 int main() {
16     int k;
17     vector<int> arr;
18     int n;
19
20     // Input size of array
21     cout << "Enter the number of elements in the array: ";
22     cin >> n;
23
24     arr.resize(n);
25     cout << "Enter the elements of the array: ";
26     for (int i = 0; i < n; ++i) {
27         cin >> arr[i];
28     }
29
30     // Input value to search for
31     cout << "Enter the value to search for: ";
32     cin >> k;
33
34     // Find the first occurrence
35     int result = findFirstOccurrence(k, arr);
36
37     // Output the result
38     cout << "Output: " << result << endl;
39 }
```

input

```
Enter the number of elements in the array: 5
Enter the elements of the array: 1 2 3 4 5
Enter the value to search for: 3
Output: 3
```

Problem 2

```

1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  using namespace std;
5
6  // Function to search for k in a sorted array using binary search
7  bool binarySearch(const vector<int>& arr, int k) {
8      int left = 0, right = arr.size() - 1;
9
10     while (left <= right) {
11         int mid = left + (right - left) / 2;
12
13         if (arr[mid] == k) {
14             return true; // Element found
15         } else if (arr[mid] < k) {
16             left = mid + 1; // Search in the right half
17         } else {
18             right = mid - 1; // Search in the left half
19         }
20     }
21
22     return false; // Element not found
23 }
24
25 int main() {
26     int n, k;
27     vector<int> arr;
28

```

input

```

Enter the number of elements in the array: 3
Enter the elements of the array (sorted in ascending order): 1 4 7
Enter the value to search for: 7
Output: true

```

Problem 3

```
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  using namespace std;
5
6  // Function to find target indices after sorting
7  vector<int> targetIndices(vector<int>& nums, int target) {
8      vector<int> result;
9
10     // Step 1: Sort the array
11     sort(nums.begin(), nums.end());
12
13     // Step 2: Find the target indices
14     for (int i = 0; i < nums.size(); ++i) {
15         if (nums[i] == target) {
16             result.push_back(i);
17         }
18     }
19
20     return result;
21 }
22
23 int main() {
24     int n, target;
25     vector<int> nums;
26
27     // Input size of array
28     cout << "Enter the number of elements in the array: ";
```

▼ ↗ 🖨 ⚙ 📄 input

```
Enter the number of elements in the array: 5
Enter the elements of the array: 1 2 3 4 5
Enter the target value: 4
Output: [3]
```

Problem 4

```
1  #include <iostream>
2  #include <vector>
3  using namespace std;
4
5  // Function to find the insert position or the index of the target
6  int searchInsert(vector<int>& nums, int target) {
7      int left = 0, right = nums.size() - 1;
8
9      while (left <= right) {
10         int mid = left + (right - left) / 2;
11
12         if (nums[mid] == target) {
13             return mid; // Target found
14         } else if (nums[mid] < target) {
15             left = mid + 1; // Search in the right half
16         } else {
17             right = mid - 1; // Search in the left half
18         }
19     }
20
21     // If target is not found, left is the insert position
22     return left;
23 }
24
25 int main() {
26     int n, target;
27     vector<int> nums;
28
```



input

Enter the number of elements in the array: 6

Enter the elements of the array (sorted in ascending order): 1 2 3 4 5 6

Enter the target value: 5

Output: 4

Problem 5

```
11 // Assign rank based on arr2's order
12 for (int i = 0; i < arr2.size(); ++i) {
13     rank[arr2[i]] = i;
14 }
15
16 // Custom comparator for sorting
17 auto comparator = [&rank](int a, int b) {
18     if (rank.count(a) && rank.count(b)) {
19         return rank[a] < rank[b]; // Both in arr2, sort by rank
20     } else if (rank.count(a)) {
21         return true; // a is in arr2 but b is not
22     } else if (rank.count(b)) {
23         return false; // b is in arr2 but a is not
24     } else {
25         return a < b; // Neither in arr2, sort ascending
26     }
27 };
28
29 sort(arr1.begin(), arr1.end(), comparator);
30
31 return arr1;
32 }
33
34 int main() {
35     int n1, n2;
36     vector<int> arr1, arr2;
37
38     // Input size of arr1
39     cout << "Enter the number of elements in arr1: ";
```

input

```
Enter the number of elements in arr1: 5
Enter the elements of arr1: 1 2 3 4 5
Enter the number of elements in arr2: 5
Enter the elements of arr2: 9 8 7 6 5
Output: [5, 1, 2, 3, 4]
```