ECE374 SP23 HW4

Contributors

Zhirong Chen (zhirong4)

Ziyuan Chen (ziyuanc3)

Problem 4

Suppose we are given an array $A[1:n]$ of n integers, which could be positive, negative, or zero, sorted in increasing order so that $A[1] \leq A[2] \leq \cdots \leq A[n]$.

Suppose we wanted to count the number of times some integer value x occurs in A. Describe an algorithm (as fast as possible) which returns the number of elements containing value x.

Solution

Intuition. We use binary search to locate the leftmost and rightmost occurrences of x in A. The number of occurrences is the difference between bounds.

```
LeftBound(A, n, x)
    left ← 1
    right ← n + 1
    while left < right
        mid ← ⌊(left + right) / 2⌋
        if A[mid] < x
            left ← mid + 1
        else
            right ← mid
    return left
```

```
RightBound(A, n, x)
    left ← 1
    right ← n + 1
    while left < right
        mid ← ⌊(left + right) / 2⌋
        if A[mid] > x
            right ← mid
        else
            left ← mid + 1
    return left − 1    // since mid ← left − 1
```

Search ranges are $[\text{left}, \text{right})$ in both cases.
The algorithms terminate when $\text{left} = \text{right}$, indicating an empty search range.

```
QuickCount(A, n, x)
    left ← LeftBound(A, n, x)
    right ← RightBound(A, n, x)
    return right − left + 1
```