ECE374 Assignment 4

Due 03/06/2023

Group & netid

Chen Si *chensi3*

Jie Wang *jiew5*

Shitian Yang *sy39*

Problem 4

4. Suppose we are given an array $A[1..n]$ of n *distinct 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: as the array is sorted, all occurrences of a number x must be consecutive. Therefore, we could solve this problem by using a binary search for the left bound of the consecutive xs and then use another binary search for he right bound of the consecutive xs. A subtraction of the indices would give us the number of xs.

Algorithm:

```
NUMBER_OF_X (A, x):
  # binary search for the left most x
  l = 0
  r = length(A) – 1
  left_most = None
  while (l <= r):
    m = floor ((l + r) / 2)      # use floor to avoid the case of
                                 # can’t reach 0 (l=0, r=1, m=0.5→1)
    if (A[m]>=x):
        left_most=m
        r = m - 1
    else:
        l = m + 1

  # binary search for the left most x
```

```
l = 0
r = length(A) – 1
right_most = None
while (l <= r):
  m = ceil ((l + r) / 2)        # use ceil to avoid the case of can’t
                                # reach n-1 (l=n-2,r=n-1,m=n-1.5→n-2)
  if (A[m] <= x):
      right_most = m
      l = m + 1
  else:
      r = m + 1

if (left_most != None and right_most != None)
    return right_most – left_most + 1
else:
  return 0
```

With two binary searches, the running time of this algorithm is O(log n)+O(log n)+O(1)=O(log n).