ADS Homework 7

Sanjay Timilsena

March 2020

1 Problem 7.1

1.1 a)

Please see the attached file counting_sort.cpp

1.2 b)

Please see the attached file bucket_sort.cpp

1.3 c)

The required pseudo-code is as follows:

```
type algorithm(arr[], a, b){
    m = max(arr);
    count = [0]*(m+1);
    for(i:range of arr[]){
        count[arr[i]]+=1;
    }
    for(i:range of arr[]){
        count[i] += count[i-1];
    }
    return count[b]-count[a-1];
}
```

1.4 d)

Please see the attached file trei_sort.cpp.

1.5 e)

When all the entries go in a single bucket then, the time complexity for bucket sort is wort. In this condition the time complexity of the algorithm depends upon the sub-sorting algorithm used, which in our case is insertion sort. The worst-case complexity of insertion-sort is $O(n^2)$, so the worst-case complexity for bucket-sort is also $O(n^2)$.

Example:
Array = (0.9, 0.5, 0.1, 0.2, 0.6)
The elements are divide into following buckets:
Bucket[0] = Null
Bucket[1] = Null
Bucket[2] = Null
Bucket[3] = Null
Bucket[4] = (0.9, 0.5, 0.1, 0.2, 0.6)
Bucket[5] = Null
Bucket[6] = Null
Bucket[7] = Null
Bucket[8] = Null
Bucket[9] = Null

Here, the time-complexity depends upon the sorting algorithm for Bucket[4] which is insertion sort in our case.

1.6 f)

The required pseudo-code is as follows:

```

type distance(arr[])
    return square root of(arr[x]^2 + arr[y]^2)

type algorithm(arr[][], size){
    for(i: 0 to size){
        flag=0;
        for(j: 0 to size-i-1){
            if(distance(A[j]) > distance(A[j+1]){
                swap(A[j], A[j+1]);
                flag=1;
            }
        }
        if(flag == 0)
            break;
    }
}
```