Design & Analysis of Algorithms I

Mid 1, Spring 2014

Date: 27th Feb. 2014 Time: 90 mins.

Q1. (5+5)

Below are the pseudo codes for insertion sort and bubble sort. It is assumed that data is stored in an array A[1 … n]. Determine the loop invariant for the inner loops of both the sorts and prove their correctness.

Insertion Sort	**Bubble Sort**

for j = 2 **to** A.*length*
    key = A[j]
    i = j -1
    while i > 0 **and** A[i] > *key*
        A[i+1] = A[i]
        i = i - 1
    A[i+1] = *key*
	for i = 1 to A.length-1
	    for j = 1 to A.length - j
	        if (A[j] > A[j + 1])
	            temp = A[j]
	            A[j]=A[j + 1]
	            A[j + 1] = temp

Remarks: Looks good. No changes from me.

Modified Q2 (10)

Below is the pseudo code of count Sort. The indexes are 0-based for the C array, but 1-based for the arrays A and B. The below algorithm is stable.

If we however change the last for loop to go from 1 up to A.length, instead of A.length down to 1, it does not remain stable.

Your task is to change the Count sort code, so that with the new code, if we go from 1 to A.length in the last for loop, it still remains stable.
The modified algorithm must still be stable, and must still run in O(n+k) time.

```
COUNT-SORT (A, B , k)
//Let C[0..k] be a new array
for i = 0 to k
        C[i] = 0
for j = 1 to A.length
        C[A[j]] = C[A[j]]+1
for i = 1 to k
        C[i] = C[i] + C[i-1]
for j = A.length down to 1
        B[C[A[j]]] = A[j]
        C[A[j]] = C[A[j]] – 1
```

Modified Q3 (10)

Let array A be an array consiting of only zeros and ones. (0's and 1's). Suggest an algorithm to sort the records in O(n) time and O(1) additional space.