CISC 233 Essential Algorithms — Problem Set # 2

Mina Gabriel.

Problem 1. *Bubblesort is a popular, but inefficient, sorting algorithm. It works by repeatedly swapping adjacent elements that are out of order.*

Algorithm 1 BUBBLESORT(A)

```
1: for $i = 1$ to $A.length - 1$ do
2:     for $j = A.length$ down to $i + 1$ do
3:         if if $A[j] < A[j-1]$ then
4:             exchange $A[j]$ with $A[j-1]$
5:         end if
6:     end for
7: end for=0
```

A. *Let A' denote the output of BUBBLESORT(A) To prove that BUBBLESORT is correct, we need to prove that it terminates and that:*

$$A'[1] \leq A'[2] \leq \cdots \leq A'[n]$$

What is meant by the previous statement?

B. *State precisely a loop invariant for the for loop in lines $2-4$, and prove that this loop invariant holds. Your proof should use the structure of the loop invariant proof presented in this chapter.*

C. *What is the worst-case running time of bubblesort? why? How does it compare to the running time of insertion sort?*

Problem 2. *Consider the **searching problem**:*

Input: *A sequence of n numbers $A = \langle a_1, a_2, \ldots, a_n \rangle$ and a value v.*

Output: *An index i such that $v = A[i]$ or -1 if v does not appear in A.*

Write python program for linear search, which scans through the sequence, looking for v. Using a loop invariant, prove that your algorithm is correct. Make sure that your loop invariant fulfills the three necessary properties.

Problem 3. *Consider linear search again (Problem 2). How many elements of the input sequence need to be checked on the average, assuming that the element being searched for is equally likely to be any element in the array? How about in the worst case? What are the average-case and worst-case running times of linear search in Θ-notation? Justify your answers.*

Problem 4. *What is the difference between **INSERTION SORT** and **MERGE SORT** in-terms of algorithm and complexity (best case, worst case, average, item exists and item doesn't exists)? Which algorithm do you prefer when the list is too small? why?*

Problem 5. *use **Algorithm 2** and the following given list to fill in the trace table until the list is completely sorted- it is up to you but you don't have to fill any repeated cell-, Remember: index starts at 1 to n*

Algorithm 2 INSERTION-SORT(A)

```
1: for $j = 2$ to $A.length$ do
2:     $key = A[j]$
3:     $i = j - 1$
4:     while $i > 0$ and $A[i] > key$ do
5:         $A[i + 1] = A[i]$
6:         $i = i - 1$
7:     end while
8:     $A[i + 1] = key$
9: end for=0
```

List:

$$[2, 5, 4, 6, 3, 1]$$

Trace Table:

line	***j***	***key***	***i***	***list***
1	***2***			***[2, 5, 4, 6, 3, 1]***
2		***5***		
...	...	...	...	...

Problem 6. *Show that for any real constants a and b, where $a > 0$,*

$$(n + a)^b = \Theta\left(n^b\right)$$

Problem 7. *Determine whether each of the following is $O(x)$*

A. $f(x) = 10$

B. $f(x) = 5\ logx$

C. $f(x) = x^2 + x + 1$

Problem 8. *Use the definition of $f(x)$ is $O(g(n))$ if c and $n_0 \ni f(n) \leq O(g(x))\ \forall\ n > n_0$ to show that:*

$$x^4 + 9x^3 + 4x + 7 \text{ is } O(x^4)$$

Problem 9. *Show that*

A. 2^n *is* $O(3^n)$ *but* 3^n *is not* $O(2^n)$

B. $n\ logn$ *is* $\theta(\log n!)$