ASSIGNMENT #1

CS 4401 ALGORITHM ANALYSIS & DESIGN

SPRING 2015

Baturay Daylak

bdaylak@students.kennesaw.edu

January 19th , 2015

College of Science and Mathematics

Kennesaw State University

Assignment Description:

2-2 Correctness of bubblesort

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

```
BUBBLESORT($A$)
1  for $i = 1$ to $A.length - 1$
2      for $j = A.length$ downto $i + 1$
3          if $A[j] < A[j - 1]$
4              exchange $A[j]$ with $A[j - 1]$
```

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

$$A'[1] \le A'[2] \le \cdots \le A'[n] , \tag{2.3}$$

where $n = A.length$. In order to show that BUBBLESORT actually sorts, what else do we need to prove?

The next two parts will prove inequality (2.3).

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. Using the termination condition of the loop invariant proved in part (b), state a loop invariant for the **for** loop in lines 1–4 that will allow you to prove inequality (2.3). Your proof should use the structure of the loop invariant proof presented in this chapter.

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

Solution to Problem

a) In addition to the output sequence to be sorted, it is also necessary that the output sequence is a permutation of the original sequence for the algorithm to be correct. (Output must contain the same elements as the input, however in sorted order.) No element of the sequence should be missing.

---

b) Let the input data be a sequence *S* with n elements. Before each iteration of the inner loop in line 2-4, there exists a subset $L=\langle a_j,..,a_n\rangle$ where a_j is smaller than or equal to any element of the *L.* This leads the smaller element of the subset $M=\langle a_i,..,a_n\rangle$ to be in subset $N=\langle a_i,..a_j\rangle$.

Initialization

At the initialization, j is equal to the size of set *S,* which is *n.* Then, subset *L* only has one element, and subsets *M* and *N* are equal. Therefore, subset *N* surely holds the smallest element of subset *M.*

Maintenance

In each iteration of inner loop, the smallest element of subset *L* is swapped with the (j-1)th element of the set *S,* therefore the leftmost element, namely a_j , of the subset *L* is also the smallest of the subset *L* prior to each iteration.

Termination

At the last iteration, j equals to i, which means the leftmost element indexed by a_j is actually a_i . Therefore, the smallest element of subset *M* is placed on a_i

after termination.

---

c) Termination condition of the loop invariant gives a direct insight about the structure of the sequence after each iteration of the outside loop: the subset $R=\langle a_1, .., a_i\rangle$ contains smallest i elements and is sorted.

Initialization:
Before the initial loop, i equals to 1, which results in a subset from a_1 to a_1 , which is trivially sorted.

Maintenance:
In each iteration of outer loop, inner loop swaps the next smallest element of the set *S* into a_i without manipulating $a_{(i-1)}$ upon its termination. Therefore, the subset *R* is sorted before each iteration of outer loop.

Termination:
During the loop before termination, inner loop j replaces the last smallest element on a_n , if any, with $a_{(n-1)}$. At the point, subset *R* is from a_1 to $a_{(n-1)}$, and it is sorted.

---

d) Worst-case running time for bubble sort is $\Theta(n^2)$ In comparison, bubble sort and insertion sort has similar worst-case running times; however, bubble sort performs move swaps then insertion sort.