CLRS Chapter 2 Edition 4 Solutions

Cameron Beck

Updated June 14, 2023

Section 2.1

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1..j-1].
4      i = j - 1
5      while i > 0 and A[i] > key
6          A[i+1] = A[i]
7          i = i - 1
8      A[i+1] = key
```

- A ***loop invariant*** is a condition that is necessarily true immediately before and immediately after each iteration of a loop. (Note that this says nothing about its truth or falsity part way through an iteration.)
- Given an appropriate invariant, we can help prove the correctness of an algorithm.
- In the example above, we might say our loop invariant is that the subarray $A[1..j-1]$ is **always** sorted.

To use a loop invariant, you must show three things

- **Initialization:** It is true prior to the first iteration of the loop.
- **Maintenance:** If it is true before an iteration of the loop, it remains true before the next iteration.
- **Termination:** The loop terminates, and when it terminates, the invariant, along with the reason that the loop terminated, gives us a useful property that helps show that the algorithm is correct.

Let's apply these principles to INSERTION-SORT

- **Initialization:** We start by considering $j = 2$. The sub-array $A[1\,..\,1]$ consists of only one element, $A[1]$, and therefore must be sorted.
- **Maintenance:** The body of the **for** loop works by moving the values in $A[j-1], A[j-2], A[j-3]$, and so on by one position to the right until it finds the proper position for $A[j]$ (line 8). Thus, the sub-array $A[1\,..\,j-1]$ remains sorted. Incrementing j for the next iteration of the **for** loop preserves the loop invariant.
- **Termination:** Once the value of j exceeds $A.length$, the loop terminates. Substituting $A.length+1$ for j in the wording of the loop variant yields that the sub-array $A[1\,..\,n]$ consists of the elements originally in $A[1\,..\,n]$, but in sorted order. Hence, the algorithm is correct.

A more formal treatment would require that we state and show a loop invariant for the **while** loop on line 5 as well.

Exercise 2.1-1

Exercise 2.1-2

Exercise 2.1-3

Section 2.2

Exercise 2.2-1

Exercise 2.2-2

Exercise 2.2-3

Problems

Problem 2-1