Computer Science 331
Solutions to Selected Tutorial #2 Questions

Questions

1. Give pseudocode for linear seearch.

   Solution:

```
int LinearSearch(T $key$)
  $i = 0$
  while $(i < n)$ and $(A[i] \neq key)$ do
    $i = i + 1$
  end while
  if $i < n$ then
    return $i$
  else
    throw KeyNotFoundException
  end if
```

2. Write a loop invariant for the loop in the algorithm.

   Solution: Loop invariant: the following properties are satisfied at the beginning of each execution of the loop body:

   - i is an integer such that $0 \leq i < n$
   - $A[j] \neq key$ for $0 \leq j \leq i$
   - A and key have not been changed

   Explanation: Recall from the notes that a loop invariant is an assertion that is true *immediately after* the loop termination test and *immediately before* before any execution of the loop body.

   The first part is to give upper and lower bounds on the loop index i. Before the first execution of the loop body we have that $i = 0$, and if we have $i \geq n$ in the termination test the loop terminates. Thus, $0 \leq i < n$.

Finally, we need to provide one or more logical statements succinctly describing the effect of the loop on the loop's variables. The purpose of the example, searching for key in the array, comes from the post-condition provided. The progress made towards this goal after an iteration of the loop comes from the fact that, before an iteration of the loop, we know that key is not equal to any of the array elements with index $j \leq i$.

Putting all this together, we have that the properties defined above represent a loop invariant for **LinearSearch**.

3. Explain how you would prove that this algorithm is correct, using your loop invariant.

   Solution: The first step is to prove that this loop invariant is correct, using mathematical induction on number of executions of the loop body (i) as follows:

   - Prove the base case, i.e. that the loop invariant holds before the first execution of the loop body when $i = 0$.
   - Assume (inductive hypothesis) that the loop body is executed at least $i \geq 0$ times and that the loop invariant is satisfied at the beginning of the ith execution.
   - Show that if there is a $i + 1$st execution of the loop body, then the loop invariant holds immediately before that execution.

   The second step is to prove that when the loop terminates, the truth of the loop invariant implies the post conditions.

   Complete proof: For your reference, a complete proof of partial correctness is given here.

   Theorem 0.1. *The loop invariant is correct.*

   Proof. Proof by induction on i.

   Base Case ($i = 0$):

   - before first execution of loop body we have $i = 0$
   - loop test passes, implying that $A[0] \neq key$
   - A and key have not been changed

   Thus the loop invariant holds.

   Inductive hypothesis: assume that the loop body is executed $i \geq 0$ times and that the loop invariant is satisfied at the beginning of the ith execution.

   By inspecting the code, we see that i has been increased by one, so at the *end* of the ith execution:

   - $0 \leq i \leq n$
   - $A[j] \neq key$ for $0 \leq j < i$

- A and key have not been changed

If there is a $i+1$st execution of the loop body, then the loop test must pass after the end of the ith execution (i.e. $i < n$ and $A[i] \neq key$), implying that immediately before the $i+1$st execution:

- $0 \leq i < n$
- $A[j] \neq key$ for $0 \leq j \leq i$
- A and key have not been changed

Thus, the loop invariant holds. □

Theorem 0.2. ***LinearSearch*** *is partially correct.*

Proof. First, note that the preconditions both imply that the loop invariant will be satisfied for $i = 0$.

Next, we apply the loop invariant to prove that the algorithm's postconditions hold. At the *end* of the loop (loop condition fails), the following properties are satisfied:

- i is an integer such that $0 \leq i \leq n$
- $A[j] \neq key$ for $0 \leq j < i$
- A and key have not been changed
- Either $i = n$ or $i < n$ and $A[i] = key$

The first three come directly from the loop invariant, the last comes from the termination condition. The postconditions are satisfied because

- Case 1 ($i = n$): loop invariant implies that $A[h] \neq key$ for $0 \leq h < n$, so key is not in A and `KeyNotFoundException` is thrown, satisfying Postcondition 2.
- Case 2 ($i < n$ and $A[i] = key$): key is found and i is returned, satisfying Postcondition 1. □

4. Prove that this loop terminates by giving a loop variant for it.

   Solution: We claim that $f(n, i) = n - i$ is a loop variant for this loop. To prove this, we note that $f(n, i)$ is an integer-valued function (because n and i are both integers), and show that $f(n, i)$ satisfies the remaining two properties of a loop variant:

   - $f(n, i) = n - i$ decreases after every iteration of the loop because i increases and n remains constant.
   - $f(n, i) \leq 0$ when $i \geq n$, and the for-loop terminates when $i = n$.

Explanation: To find a correct loop variant, we need to construct a function involving the loop's variables that decreases after each iteration of the loop body and implies the termination of the loop when ≤ 0. In this case, the loop variant will be a function of the length of the array n and the loop index i, as these are the only variables involved in deciding the loop's termination. The second condition ($key = A[i]$) could also cause the loop to terminate but always earlier than the first condition, so it does not need to be included in the loop variant (think of the loop variant as a "worst-case" termination condition). The loop terminates when $i = n$, so the function must be ≤ 0 whenever $i \geq n$ and > 0 whenever $i < n$ (note that i increases after each execution of the loop body). Putting these together, we see that $f(n, i) = n - i$ satisfies the requirements.