Worksheet 14 Solution

March 26, 2020

Question 1

a. **Inner Loop Iterations (upper bound):** n

Inner Loop Step Size: 1

Inner Loop Steps Total: n

Outer Loop Iterations (upper bound): n

Outer Loop Step Size: 1

Outer Loop Steps Total: n

Steps Total: $n \cdot n = n^2$

Correct Solution:

Since the inner loop starts at $i+1$ and ends at $n-1$, where i represents the variable in outer loop, the inner loop has $(n-1-(i+1)+1) = n-i-1$ iterations.

Since each iteration takes 1 step, the total steps taken by inner loop is:

$$(n-i-1) \cdot 1 = (n-i-1) \tag{1}$$

Now, we will evaluate total steps taken by outer loop.

Since the outer loop starts at $i = 0$, and ends at $n - 1$, the loop runs at most n iterations.

Since each iteration takes $(n - i - 1)$ steps, the total steps of outer loop is:

$$
\begin{aligned}
\sum_{i=0}^{n-1}(n-i+1) &= \sum_{i=0}^{n-1}[(n-1)-i] && (2)\\
&= \sum_{i=0}^{n-1}(n-1) - \sum_{i=0}^{n-1} i && (3)\\
&= n(n-1) - \frac{n(n-1)}{2} && (4)\\
&= \frac{n^2-n}{2} && (5)
\end{aligned}
$$

Then, since the last **return** statement takes 1 step, it follows that the total number of steps of this algorithm is at most $\frac{n^2-n}{2} + 1$, or $\mathcal{O}(n^2)$.

---

b. Consider the input family where none of the values in a list are the same (i.e. $[1, 2, 3, 4, 5, 6, 7, 8, 9]$).

Since all values in the input list are not matching, both the inner and the outer loop will run, giving the loops the total number of steps of $\frac{n^2-n}{2}$.

Since the last **return** statement takes 1 step, the total number of steps of this algorithm is $\frac{n^2-n}{2} + 1$, or $\Omega(n^2)$.

<u>Correct Solution:</u>

Let $n \in \mathbb{N}$ and $lst = [1, 2, 3, \ldots, n-1, n-1]$.

Since the inner loop will run without interruptions until the end, the inner loop has

$$n - 1 - (i + 1) + 1 = n - i - 1 \tag{1}$$

iterations.

Then, since the inner loop takes 1 step per iteration, the total steps taken by the inner loop is

$$(n - i - 1) \cdot 1 = (n - i - 1) \tag{2}$$

Since the **if condition** $lst[i] == lst[j]$ and the **return** statement are activated when $i = n - 2$, the outer loop will run until $i = n - 2$, where j is the variable of the inner loop and i is the variable of the outer loop.

Since the outer loop starts at 0 and ends at $n - 2$, it has

$$n - 2 + 1 = n - 1 \tag{3}$$

iterations.

Since each iteration in the outer loop takes $(n - i - 1)$ steps, the outer loop has total cost of

$$\begin{align}
\sum_{i=0}^{n-2}(n - i - 1) &= \sum_{i=0}^{n-2}(n - 1) + \sum_{i=0}^{n-2} i \tag{4}\\
&= (n - 1)(n - 1) - \frac{(n-2)(n-1)}{2} \tag{5}\\
&= \frac{(n-1)n}{2} \tag{6}
\end{align}$$

Since each of the **if condition** and **return** statement has cost of 1, the total cost of algorithm is $\frac{n(n-1)}{2} + 2$, or $\Omega(n^2)$

---

c. Let $n \in \mathbb{N}$, $lst_{upper} = [1, 2, 3, \ldots, n-1, 1]$ and $lst_{lower} = [1, 2, 3, \ldots, 1, n-1]$.

We will prove this statement by finding the upper and lower bound of this input family, and combine them together.

Part1 (Upper bound):

Since the inner loop will run from $j = i + 1$ until the end without interruptions, the loop has

$$(n-1) - (i+1) + 1 = n - i - 1 \tag{1}$$

iterations.

Since the inner loop takes 1 step per iteration, the loop takes total of

$$(n - i - 1) \cdot 1 = (n - i - 1) \tag{2}$$

steps.

Now, because we know that the **if condition** and **return** statement will occur at $i = 0$, the outer loop has at most 1 iteration.

Because we know that the outer loop terminates at $i = 0$, the total cost of inner loop can be simplified to

$$(n - i - 1) = n - 1 \tag{3}$$

Since the outer loop has 1 iteration and takes $n - 1$ steps, the loop has total cost of $n - 1$.

Lastly, since each of the **if condition** and **return** statement has cost of 1, the total cost of the algorithm is

$$n - 1 + 2 = n + 1 \tag{4}$$

steps.

Question 2