22C:44 Homework 1 Solutions

Solution to Problem 1(a)
Rewrite the algorithm in the model of computation defined in class, as follows:

Line No.	Time Units	Label	Code
1.	1		`i ← 1`
2.	4/5	OUTLOOP	`if (i > n-1) goto AFTEROUTLOOP`
3.	2		`j ← n`
4.	4/5	INLOOP	`if (j < i+1) goto AFTERINLOOP`
5.	6/7		`if (A[j-1] <= A[j]) got AFTERIF`
6.	3		`t ← A[j]`
7.	5		`A[j] ← A[j-1]`
8.	4		`A[j-1] ← t`
9.	3	AFTERIF	`j ← j - 1`
10.	1		`goto INLOOP`
11.	3	AFTERINLOOP	`i ← i + 1`
12.	1		`goto OUTLOOP`
13.		AFTEROUTLOOP	

Alongside each statement, the number of time units required to execute the statement is also given. For Lines 2, 4, and 5 there are two entries each in the `Time Units` column. The first entry gives the time taken if the boolean condition in the statement evaluates to `True`. The second entry corresponds to a `False` evaluation of the boolean condition.

From this it follows that the worst case time taken for a sucessful execution of the inner for-loop is

$$4 + 6 + 3 + 5 + 4 + 3 + 1 = 26.$$

An unsuccessful attempt to execute the inner for-loop takes 5 units of time. For any i, $1 \le i \le n-1$, the inner for-loop executes $n - i$ times. Therefore, for any i, $1 \le i \le n-1$, Lines 4-10 take a total of $26(n-i) + 5$ time units.

For any i, $1 \le i \le n-1$, the ith execution of the outer for-loop takes time

$$4 + 2 + 26(n-i) + 5 + 3 + 1 = 26(n-i) + 15.$$

Therefore, *all* the successful executions of the outer for-loop take time

$$\sum_{i=1}^{n-1} [26(n-i) + 15] = 15(n-1) + 26\sum_{j=1}^{n-1} j = 15(n-1) + 13n(n-1) = 13n^2 + 2n - 15.$$

To this quantity we add (i) the time for one unsuccessful execution of the outer for-loop and (ii) time for Statement 1 to get

$$13n^2 + 2n - 15 + 1 + 5 = 13n^2 + 2n - 9$$

as the total number of time units the program takes to run.

Solution to Problem 1(b)
In the new model of computation the running time of Lines 7 and 8 changes to

Line	Time Units
7	$(n+4)$
8	$(n+3)$

From this it follows that the worst case running time of each successful execution of the inner for-loop is, in the worst case,

$$4 + 6 + 3 + (n + 4) + (n + 3) + 3 + 1 = 2n + 24.$$

So for any i, $1 \le i \le n - 1$, Lines 4-10 take a total of $(n - i)(2n + 24) + 5$ time units. For any i, $1 \le i \le n - 1$, the ith execution of the outer for-loop takes

$$(n - i)(2n + 24) + 5 + (4 + 2 + 3 + 1) = 2n^2 - 2in + 24n - 24i + 15.$$

All successful executions of the outer for-loop takes

$$\sum_{i=1}^{n-1} [2n^2 - 2in + 24n - 24i + 15] = n^3 + 11n^2 + 3n - 15.$$

To this quantity we add (i) the time for one unsuccessful execution of the outer for-loop and (ii) time for Statement 1 to get

$$n^3 + 11n^2 + 3n - 9$$

as the total running time of the program.

Solution to Problem 2

The assignment statement inside the inner most for-loop (the k-loop) takes $\Theta(1)$ time. Therefore, the total running time of the function is

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{\min\{i,j\}} \Theta(1) &= \sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{j}\Theta(1) + \sum_{i=1}^{n}\sum_{j=i+1}^{n}\sum_{k=1}^{i}\Theta(1) \\
&= \sum_{i=1}^{n}\sum_{j=1}^{i}\Theta(j) + \sum_{i=1}^{n}\sum_{j=i+1}^{n}\Theta(i) \\
&= \sum_{i=1}^{n}\Theta\left(\frac{i(i+1)}{2}\right) + \sum_{i=1}^{n}\Theta(i(n-i)) \\
&= \Theta\left(\sum_{i=1}^{n}\left[\frac{i}{2} - \frac{i^2}{2} + ni\right]\right) \\
&= \Theta\left(\frac{n(n+1)}{4} - \frac{n(n+1)(2n+1)}{12} + \frac{n^2(n+1)}{2}\right) \\
&= \Theta\left(\frac{n^2}{2} + \frac{n^3}{3} + \frac{n}{6}\right) \\
&= \Theta(n^3).
\end{aligned}$$

In evaluating the above sum we use the fact that

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}.$$

Solution to Problem 3

$$C_1 = \left\{\frac{1}{n}\right\}$$

$$
\begin{aligned}
C_2 &= \left\{\frac{8n}{n!} + 20,\ 20\right\} \\
C_3 &= \{\lg\lg n\} \\
C_4 &= \{\lg^2 n\} \\
C_5 &= \left\{\frac{n}{\lg n}\right\} \\
C_6 &= \{\lg(n!)\} \\
C_7 &= \left\{((n+16)(8n^{0.5} + \lg n)),\ n^{3/2}\right\} \\
C_8 &= \left\{\frac{n^2}{\ln^2 n},\ \frac{8n^2}{\lg^2 n} + n\lg n\right\} \\
C_9 &= \left\{(n+10)^5,\ 7n^5 - 30n + 2\right\} \\
C_{10} &= \left\{2^{\lg^2 n},\ n^{\lg n} + 80n^5\right\}
\end{aligned}
$$

Solution to Problem 4

(a) $f(n) + g(n) = \Theta(\max(f(n), g(n)))$. **True**
Let $c_1 = 1$, $c_2 = 2$, and n_0 be such that for all $n \geq n_0$ $g(n), f(n) \geq 0$. We know that n_0 exists because the functions are asymptotically non-negative. We now show that for all $n \geq n_0$

$$\max\{f(n), g(n)\} \leq f(n) + g(n) \leq 2\max\{f(n), g(n)\}.$$

This is sufficient to show that $f(n) + g(n) = \Theta(\max\{f(n), g(n)\})$.

For all $n \geq n_0$, $f(n), g(n) \geq 0$. This implies that for all $n \geq n_0$, $f(n) + g(n) \geq f(n)$ and $f(n) + g(n) \geq g(n)$. This further implies that for all $n \geq n_0$, $f(n) + g(n) \geq \max\{f(n), g(n)\}$.

For all n, $f(n) \leq \max\{f(n), g(n)\}$ and $g(n) \leq \max\{f(n), g(n)\}$. Hence, we have that $f(n) + g(n) \leq 2\max\{f(n), g(n)\}$.

(b) $f(n) + o(f(n)) = \Theta(f(n))$. **True**.
$f(n) + o(f(n)) = f(n) + g(n)$ where $g(n)$ is some function such that for every constant $c > 0$, there is a constant $n_0 > 0$ such that $0 \leq g(n) < cf(n)$ for all $n \geq n_0$. In particular, letting $c = 1$, we have that there is a constant n_0 such that $0 \leq g(n) < f(n)$ for all $n \geq n_0$. Therefore, $f(n) \leq f(n) + g(n) < 2f(n)$ for all $n \geq n_0$. By the definition of the "Big-Theta" notation it follows that $f(n) + o(f(n)) = \Theta(f(n))$.

(c) $f(n) = O(f(n)^2)$. **False**.
Let $f(n) = 1/n$. Clearly, $1/n \neq O(1/n^2)$.

(d) $(n+1)^2 = O(n^2)$. **True.**
To show this we need to show that there exist constants $c, n_0 > 0$ such that for all $n \geq n_0$, $0 \leq (n+1)^2 \leq cn^2$. The inequality $(n+1)^2 \geq 0$ is true for all $n \geq 0$. Now we consider the inequality $(n+1)^2 \leq cn^2$:

$$
\begin{aligned}
(n+1)^2 \leq cn^2 \quad &\equiv \quad cn^2 - (n+1)^2 \geq 0 \\
&\equiv \quad (\sqrt{c}n - (n+1))(\sqrt{c}n + (n+1)) \geq 0
\end{aligned}
$$

So we can ensure that $(n+1)^2 \leq cn^2$ by making sure that

$$\sqrt{c}n - (n+1) \geq 0 \quad \text{and} \qquad \sqrt{c}n + (n+1) \geq 0.$$

Now if we let $c = 4$, we see that both inequalities are true for all $n \geq 1$. Hence, we have shown that for all $n \geq 1$, $0 \leq (n+1)^2 \leq 4n^2$. By the definition of the "Big-Oh" notation we have that $(n+1) = O(n^2)$.