1 Solutions

Question 1 ():

Solution: Coding question, see submitted code files

Question 2 ():

Solution:

(a) $O(2^n)$. *The last term, which is exponential, is the fastest growing term.*

(b) $O(n^4)$. *The last term simplifies to just n, so the n^4 term grows fastest. Note: $n^2 \cdot \log n^2 = 2n^2 \log n < n^2 \cdot n = n^3$, so the middle term does not grow as fast as n^3.*

(c) $O(n \log_2 n)$. *The middle term $29n \log_2 n$ is $O(n \log_2 n)$ which grows more quickly than $O(n)$ which grows more quickly than $n^{0.7}$.*

Question 3 ():

Solution: Answers for each part:

a) Answer: Statements 3 and 4 are true. *Explanation: Statement 3 is true because n^3 is the tightest upper bound ($19n^3$) is the fastest growing term, and stripping the constants leaves n^3. But 2^n grows even more quickly than n^3 so if n^3 is an upper bound, then 2^n must also be an upper bound.*

c) Answer: $\Theta(n^3)$. *Explanation: Since the third term in the expression is inarguably $\Theta(n^3)$ and it is the most quickly growing thrm, then $T_A(n) \in \Theta(n^3)$. This is the only possible answer because big-Θ implies that the given function is not only an upper bound, but also a lower bound. Anything that grows more quickly that n^3 would not be a lower bound.*

Question 4 ():

Solution:

(a) $O(n^2)$. *By first rule in lecture 4, slide 11.*

(b) $O(n^2 2^n)$, or equivalently $O(2^n n^2)$. *By second rule in lecture 4, slide 11.*

(c) $O(n^3)$. *By first applying rule 3 to each term, then applying rule 1.*

(d) $O(n^2 \log_2 n^2 + m)$ *By rule #1 in lecture 4, slide 11. This is as much as we can simplify because we can't possibly know which term grows faster because m and n are independent..* $O(n^2 \log_2 n + m)$ is also acceptable since $\log_2 n^2 = 2 \log_2 n$.

Question 5 ():

Solution:

(a) This is a *dependent quadratic loop* where the number of iterations of the inner loop depends on the value of i in the outer loop.

The inner loop executes two statements per loop iteration (the print statement and the loop condition). The inner loop condition is true $n-i-1$ times, depending on the value of i. So the total number of statements executed by the inner loop is $2(n-i-1)+1$

(since this is pseudocode, there is no explicit loop condition to count, so if the loop condition was not included, that solution is also acceptable, in which case the number of statements executed by the inner loop is $(n-i-1)+1$ and the factor of 2 disappears from the simplification below; it's also not a big deal if they miss the $+1$ for the loop condition being false, in which case the trailing n term disappears from the simplification below).

The outer loop executes n times, for values of i from 0 to $n-1$. Thus the total number of statements executed is the sum of the number of statements executed by the inner loop for each of these values of i:

$$\begin{aligned}
\sum_{i=0}^{n-1}[2(n-i-1)+1] &= 2\times((n-1)+(n-2)+(n-3)+\cdots+2+1+0)+n \\
&= 2\times(0+1+2+\cdots+(n-3)+(n-2)+(n-1))+n \\
&= 2\sum_{i=0}^{n-1} i+n \quad \textit{(a familiar sum for which we know a closed form!)} \\
&= 2(n-1)(n)/2+n \\
&= n^2
\end{aligned}$$

Note: it is also acceptable to count the lines where the "loop condition" would be false. So there could be minor variants with an extra $+1$ or $+2$, and that's fine.

(b) $\Theta(n^2)$

Question 6 ():

Solution: The active operation is the inner-loop condition. It is executed one more time than the print statement and all individual statements are $O(1)$.

The cost of the active operation is $O(1)$. The active operation is executed $(n-i-1)+1$ times by the inner loop. The outer loop executes n times, once each for the values of i from 0 to $n-1$. Thus the total cost of all active operation invocations is:

$$\begin{aligned}
&\sum_{i=0}^{n-1}[O(1)\cdot((n-i-1)+1)] \\
&\sum_{i=0}^{n-1}[(n-i-1)+1] \\
&= \sum_{i=0}^{n-1}\cdot(n-i-1)+n \\
&= \sum_{i=0}^{n-1} i+n \quad \textit{(By same reasoning as previous question)} \\
&= (n)(n-1)/2+n \\
&= n^2/2+n/2
\end{aligned}$$

Question 7 ():

Solution: Coding question, see submitted code files

Question 8 ():

Solution: Coding question, see submitted code files