Quiz Questions: Complexity of Algorithms

1. If for an algorithm time complexity is given by $O((3/2)^n)$ then complexity is:
   a. Quadratic
   b. Cubic
   c. Exponential
   d. Rational

   Explanation: the algorithm is $O(b^n)$ for $b > 1$, so it is exponential.

----

2. What is the least integer n such that $f(x) = 3x^3 + (log\ x)^4\ is\ O(x^n)$:
   a. 1
   b. 2
   c. 3
   d. 4

   Explanation: $f(x)$ is $O(x^3)$. The term $3x^3$ dominates the term $(log\ x)^4$

----

3. Let k be a positive integer, then $1^k + 2^k + \cdots + n^k$ is:
   a. $O(k^{n+1})$
   b. $O(n^{k+1})$
   c. $O(\log_k n)$
   d. $O(kn^2)$

   Explanation: $1^k + 2^k + \cdots + n^k \leq n * n^k = n^{k+1}$, therefore, it is $O\left(n^{(k+1)}\right)$ (for all positive k)

   $1^k + 2^k + \cdots + n^k \geq n^k$. This function grows faster (in n) than $\log_k n$ and kn^2
   For $k = 1$ the formula reduces to $n * (n+1)/2$ which grows faster than $O(1 = 1^{(n+1)})$, therefore $O(k^{n+1})$ is also not correct for all positive k.

----

4. The big-O estimate for $\sum_{j=1}^{n} j(j+1)\ \ is$:
   a. $O(2^n)$
   b. $O(n^2)$
   c. $O(\log n)$
   d. $O(n^3)$

   Explanation: Unfortunately the question should have asked for the big-Theta estimate to be correct (or the best big-O estimate)
   $\sum_{j=1}^{n} j(j+1) \leq \sum_{j=1}^{n} n(n+1) \leq n^2(n+1)$ which is $O(n^3)$
   $\sum_{j=1}^{n} j(j+1) \geq \sum_{j=ceil\left(\frac{n}{2}\right)}^{n} j(j+1) \geq \sum_{j=ceil\left(\frac{n}{2}\right)}^{n} j(j+1)\, {}^{n}/_{2}\left({}^{n}/_{2}+1\right) \geq {}^{n}/_{2} * {}^{n}/_{2} * \left({}^{n}/_{2}+1\right)$
   which is $\Omega(n^3)$.
   So the function is in fact $\Theta(n^3)$. Therefore it is also $O(2^n)$.

----

5. Algorithm **A** and **B** have a worst-case running time of $O(n)$ and $O(\log n)$, respectively.

a. For all A, B, I A runs faster than B for input I
b. For all A, B, I B runs faster than A for input I
c. For all B exists A for all I B runs faster than A for input I
d. None of the possibilities is correct

Explanation: answers a and b are wrong, since independent of the complexity of the algorithm it is always possible that on a specific input the algorithm A succeeds e-g- in the first step and the other has to execute multiple steps (example: linear search vs. Binary search).

Answer c is correct, since whatever algorithm B we choose with complexity O(log n), we can choose an arbitrary algorithm A' with complexity O(n), and write a new algorithm A = A';B that executes first A and then B, and is of complexity O(n), and that always performs more steps than B alone.

----

6. Which of the following are ordered by increasing complexity:
   a. $n \log n^2 < n(\log n)^2 < n^2$
   b. $n^2 < n \log n^2 < n(\log n)^2$
   c. $n(\log n)^2 < n \log n^2 < n^2$
   d. $n \log n^2 < n^2 < n(\log n)^2$

Explanation: when we write $f(n) < g(n)$ we mean that an algorithm that is $O(f(n))$ is also $O(g(n))$.

Since $log\, n^2 = 2\, log\, n$, $log\, n^2 < (log\, n)^2$, and therefore $n\, log\, n^2 < n\, (log\, n)^2$
Since for all k, $(log\, n)^k < n$, we have $(log\, n)^2 < n$ and therefore $n\, (log\, n)^2 < n^2$

----

7. Which of the following is correct?
   a. $x^3 is\ o(x^2)$
   b. $x^3 is\ o(x^{\frac{3}{2}})$
   c. $x^3 is\ o(x^3)$
   d. $x^3 is\ o(x^4)$

Explanation: $\lim x^3 / x^k$ is not 0 for $k = 2, \frac{3}{2}, 3$
More precisely it is infinity for $k = 2, \frac{3}{2}$ and 1 for $k = 3$

----

8. The complexity of adding two $n \times n$ matrices is:
   a. $O(n)$
   b. $O(n^2)$
   c. $O(n^3)$
   d. $O(2^n)$

Explanation: for adding two matrices we have to add all the entries of the two matrices separately. Since there are n^2 entries in an $n \times n$ matrix, the complexity is $O(n^2)$.