ICT1008: Data Structures and Algorithms

Tutorial 3: Analysis of Algorithms

Q1. A sorting method with "Big-Oh" complexity O($nlog_{10}n$) spends exactly 1 millisecond to sort 1,000 data items. Assuming that time T(n) of sorting *n* items is directly proportional to $nlog_{10}n$, that is, T(n) = $cnlog_{10}n$, where *c* is a constant. Derive a formula for T(n), given the time T(*N*) for sorting *N* items, and estimate how long this method will sort 1,000,000 items.

Q2. A quadratic algorithm with processing time T(n) = cn^2 spends T(*N*) seconds for processing *N* data items. How much time will be spent for processing *n* = 5000 data items, assuming that *N* = 100 and T(*N*) = 1ms?

Q3. Assume that each of the expressions below gives the processing time $T(n)$ spent by an algorithm for solving a problem of size *n*. Select the dominant term(s) having the steepest increase in *n* and specify the lowest Big-Oh complexity of each algorithm.

Expression	Dominant term(s)	Big-Oh complexity
$5 + 0.0001n^3 + 0.025n$		
$500n + 100n^{1.5} + 50nlog_{10}n$		
$0.3n + 5n^{1.5} + 2.5n^{1.75}$		
$n^2 \log_2 n + n(\log_2 n)^2$		
$nlog_3n + nlog_2n$		
$100nlog_3n + n^3 + 100n$		

Q4. The statements below show some features of "Big-Oh" notation for the functions f(n) and g(n). Determine whether each statement is TRUE or FALSE and correct the formula in the latter case.

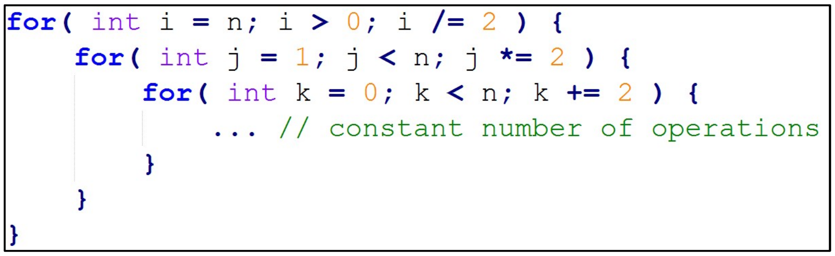
Statement	Is it TRUE or FALSE	If it is FALSE, write the correct formula
Rule of sums: $O(f + g) = O(f) + O(g)$		
$O(f \cdot g) = O(f) \cdot O(g)$		
$5n + 8n^2 + 100n^3 = O(n^4)$		
$5n + 8n^2 + 100n^3 = O(n^2logn)$		

Q5. Prove that $T(n) = 5 + 7n + 2n^2 + 9n^3$ is $O(n^3)$ using the formal definition of the Big-Oh notation. Hint: Find a constant c and threshold n_0 such that $cn^3 \geq T(n)$ for $n \geq n_0$.

Q6. Algorithms A and B spend exactly $T_A(n) = c_A n \log_2 n$ and $T_B(n) = c_B n^2$ microseconds, respectively, for a problem of size n. Find the best algorithm for processing $n = 2^{20}$ data items if the algoritm A spends 10 microseconds to process 1024 items and the algorithm B spends only 1 microsecond to process 1024 items.

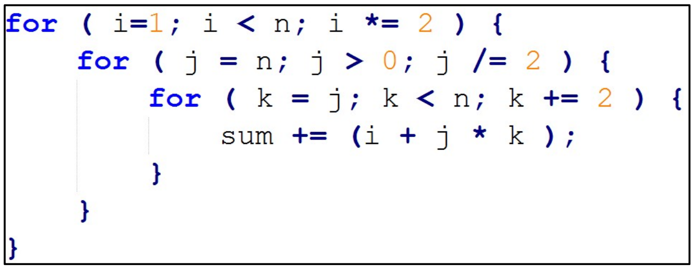
Q7. Find the computational complexity of the following piece of code using Big-oh notation:

a)



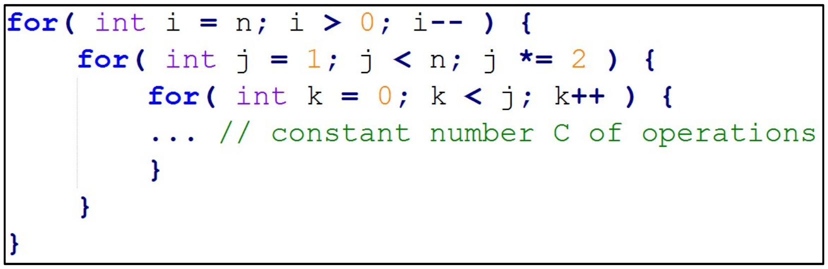
```
for( int i = n; i > 0; i /= 2 ) {
    for( int j = 1; j < n; j *= 2 ) {
        for( int k = 0; k < n; k += 2 ) {
            ... // constant number of operations
        }
    }
}
```

b)



```
for ( i=1; i < n; i *= 2 ) {
    for ( j = n; j > 0; j /= 2 ) {
        for ( k = j; k < n; k += 2 ) {
            sum += (i + j * k );
        }
    }
}
```

c) Find the computational complexity of the following piece of code assuming that $n = 2^m$:



```
for( int i = n; i > 0; i-- ) {
    for( int j = 1; j < n; j *= 2 ) {
        for( int k = 0; k < j; k++ ) {
        ... // constant number C of operations
        }
    }
}
```