ESO207: Assignment 1

Due on 10 August,2015

Q1.1 Show using the definition of $O()$ as given in the class

(a)[5] that the time complexity of an algorithm is $O(n^2)$ if and only if it is $O(n^2 + 1000000000n)$.

(b)[5] that the time complexity of an algorithm is $O(n^2 + n^{1.8})$ if and only if it is $O(n^2)$

Q1.2[15] For each pair (i, j) determine whether $g_i(n)$ is $O(g_j(n))$ or not

(i)	$g_1(n)$	=	n^2	for even n
			n^3	for odd n
(ii)	$g_2(n)$	=	n	for $0 \leq n \leq 100$
			n^3	for $n > 100$
(iii)	$g_3(n)$	=	$n^{2.5}$	

Q2.1[5] If $f(n)$ is $O(g(n))$, then does it imply that $g(n)$ is $\Omega(f(n))$? Justify.

Q2.2 Let $T_1(n)$ be $\Omega(f(n))$ and $T_2(n)$ be $\Omega(g(n))$. For each of the following statements determine whether it is true or false and justify.

(i)[5] $T_1(n) + T_2(n)$ is $\Omega(\max\{f(n), g(n)\})$.

(ii)[5] $T_1(n).T_2(n)$ is $\Omega(f(n).g(n))$.