Student Number: 1226909816
Name: Wei Hng Yeo
CSE450 Assignment 1

1) In ascending order, f1(n), f6(n), f4(n), f3(n), f2(n), f7(n), f5(n), f8(n)

2) (a) $O(3n^2 + 5) = O(3n^2) = O(n^2)$ as constant multiples or additions can be omitted in the order of growth computation. Thus since $O(n^2) \notin O(n)$ as $O(n^2)$ exceeds the upper bound $O(n)$, so $O(3n^2 + 5) \notin O(n)$.

   (b) Since $O(n^{1.1})$ cannot be simplified such that it becomes $O(kn)$ where k is a constant multiple, so $O(n^{1.1})$ exceeds the upper bound $O(n)$, $O(n^{1.1}) \notin O(n)$.

   (c) Since the $\Omega(n^2\log(n))$ tells us the lower bound runtime of $n^2\log(n)$ while $O(n^2\log(n))$ tells us the upper bound runtime of $n^2\log(n)$. But in this case, the big theta must be $\Theta(n^2\log(n))$ as big theta definition is to find the tightest bound between big ohmega and big O. Thus $\Theta(n^2\log(n)) \notin \Theta(n^2)$ as the $n^2\log(n)$ exceeds the tightest bound of n^2.

   (d) Since 2^n does not exceed the upper bound of n! except only at small n values such as when $n = 2$, thus $2^n \in O(n!)$.

   (e) $O(n^4) + \Theta(n^3)$ can be simplified to $O(n^4)$, and since $O(n^4)$ gives the upper bound, while $\Theta(n^4)$ gives the tight bound. $O(n^4) + \Theta(n^3) \in \Theta(n^4)$.

3) (a) $f(n) = O(f(n)^2)$ is false for $0 < f(n) < 1$, since big O describes the upper bound for a certain function. If for a particular n, $0 < f(n) < 1$, $f(n)^2$ would be smaller than f(n) which contradicts the definition of big O describing the upper bound.

   (b) $f(n) = \Theta(\max(f(n), g(n))$ is not true when $g(n) > f(n)$. Since the definition of big theta is that it gives the tightest bound of function f(n), the tightest bound of function f(n) should be itself. But when g(n) is greater than f(n), big theta would be g(n) which is incorrect unless $g(n)= f(n)$ for all values of n.

   (c) "If $f(n) = \Omega(g(n))$, then $f(n) = O(g(n))$" is not true. Say if $g(n) = n$ and $f(n) = n^2$, g(n) is indeed the lower bound of f(n) but it is not the upper bound of f(n) since $n < n^2$ for $n < 0$ and $n > 1$.

   (d) If $f(n) = O(g(n))$, then $2^{f(n)} = O(2^{g(n)})$ is true. Since $f(n) \leqslant g(n)$, so $2^{f(n)} \leqslant 2^{g(n)}$.

4) Operation per second = 10^{10} operations/second
   Total operations in 1 hours = $10^{10} \times (60 \times 60) = 3.6 \times 10^{13}$ operations

   (a) Largest n = 6000000
   (b) Largest n = 33019
   (c) Largest n = 600000

(d) Largest n = 1.29095×10^{12}
(e) Largest n = 45