(1) Suppose you are comparing implementations of insertion sort and selection sort on the same machine. For inputs of size $n$, insertion sort runs in $8n^2 + 20n$ instructions, while selection sort runs in $10n^2$. For which values of $n$ does insertion sort beat selection sort?

(2) Assume you have an algorithm that runs in $n$ time. What is the largest size $n$ that can be solved by this algorithm in 1 day? Assume a 1GHz processor (therefore one instruction takes $10^{-9}$ seconds).

(3) Same as above, when the algorithm runs in $n^2$ instructions.

(4) Same as above, when the algorithm runs in $2^n$ instructions.

(5) Order the following expressions by their asymptotic growth (fastest to slowest):

$$2^n, n^3, 3^n, n\lg n, n^2, n^2\lg n$$

Justify your answer. That is, if your final order is $a, b, c, d$ you need to show that $a = O(b), b = O(c)$ and $c = O(d)$. You can use either the definition of $O()$ or limits.

(6) Find an asymptotic tight bound for the following expressions:
   (a) $3n + \lg n + n\lg n + 27$
   (b) $10000n + 3n^2$
   (c) $n^3 + 2^n$
   (d) $n^2 + 3n^3 + n\lg n$

(7) What is order of growth of the running time of the following function, which reverses a string $s$ of length $n$? You may want to think whether concatenating two strings $s_1 + s_2$ is a constant time operation (or not).

```java
public static String reverse(String s) {
    int N = s.length();
    String reverse = "";
    for (int i = 0; i < N; i++)
        reverse = s.charAt(i) + reverse;
    return reverse;
}
```

(8) Give a $\Theta(n)$ algorithm for reversing a string.