CMPT 306 Algorithms & Data Structures

1. Use the Master Theorem to find the order of growth for solutions of the following recurrences.

- $T_n = T_{\frac{n}{3}} + n$
- $T_n = 9T_{\frac{n}{3}} + n^{2.5}$
- $T_n = 8T_{\frac{n}{2}} + n^2$
- $T_n = 8T_{\frac{n}{2}} + n^4 - 2$
- $T_n = 4T_{\frac{n}{4}} + n - 3$

2. This question comes in 2 parts:

- (6 points) Design a recursive divide-and-conquer algorithm for calculating a^n where $a > 0$ and $n > 0$.
- (4 points) Set up and solve a recurrence relation to determine the order of growth of your algorithm.

3. Consider a *ternary* search algorithm which searches for an element x in an ordered list of size n. The algorithm first tests the element at position $n/3$ for equality with x and then possibly checks the element at $2n/3$ either discovering x or reducing the size of the original list by two thirds.

- What algorithmic technique is this algorithm based upon?
- Set up a recurrence relation that models this algorithm.
- What is the worst-case order of growth of the ternary search algorithm?

4. The following recursive algorithm (in pseudocode) determines the height of a binary tree. Assuming a binary tree has N nodes, analyze this algorithm using the big-Theta notation. (Hint - first find a recurrence relation and solve it.)

```
int height(Tree T) {
  if (T == null)
    return 0;
  else {
    if ( height(left subtree of T) > height(right subtree of T) )
      return height(left subtree of T) + 1;
    else
      return height(left subtree of T) + 1;
  }
}
```