Short Answer: Analysis of Algorithms [8 points]

1. Consider the following growth function: [4 points]

   $f(n) = 12n^2 + 42log(n) + 30n^3$

   (a) What is the Big-Oh order of this function? You should provide a relatively tight upper bound (e.g., not just 2^n). [1 point]

   (b) Prove that your bound holds. (Hint: the definition for Big-Oh is $f(n) = O(g(n))$ iff $|f(n)| \leq c|g(n)|$ (for $n > x_0$, where x_0 is a constant, and c is also a constant). [3 points]

2. What is the Big-Oh order of the following code fragment? The fragment is parameterized on the variable n. Assume that you are measuring the number of println calls. You should provide a relatively tight upper bound. [2 points]

```java
for (int i = 1; i <= n; i++)
  for (int j = n-100; j >= 0; j--)
    for (int k = 1; k <= c; k++)
      System.out.println("Nested loops!"); //count these
```

3. Consider the following algorithm that implements matrix addition:

   ```
   public Matrix plus(Matrix other) {
     if(!sameDimensions(other)) throw new RuntimeException("Incompatible");

     int[][] result = new int[data.length][data[0].length];

     for(int y = 0; y < data.length; y ++)
       for (int x = 0; x < data.length; x++)
         result[y][x] = data[y][x] + other.getElement(y, x);
     return new SolnMatrix(result);
   }
   ```

   If you were choosing a cost metric (i.e., what operation should be counted), in order to determine this algorithm's Big-Oh order, what operation would be the best choice? Explain. [2 points]

4. If the induction proof for the towers of Hanoi problem used a base case of n=5 (instead of n=1), would it still be a valid proof? How would the truth that it aims assert be different than the n=1 proof? [2 points]