1. Algorithm A takes time $n^2$ and Algorithm B takes time $3(n^2)$, where $n$ is a positive integer that measures the size of the problem that is being solved. For which values of $n$ is Algorithm A faster, for which values is Algorithm B faster.

2. What is the value of

$$\sum_{i,j,k} \binom{n}{i,j,k} 2^j 3^k.$$ 

3. Simplify

$$\sum_i \frac{(p+i)!}{(2p+2i)![(2n-2p-2i)!]!}.$$ 

4. You have sets $I$ and $J$, and you want to determine whether or not set $I$ is a subset of set $J$. Since $I$ is a set, its elements are distinct; similarly for $J$. You also know that set $I$ has $i$ elements and that set $J$ has $j$ elements. Set $I$ is a subset of set $J$ if and only if each element of $I$ is also an element of $J$. Consider an algorithm that solves this problem by building a hash table containing the elements of $J$ and then looking in the hash for each element of $I$. What will be the average time for this algorithm in the case that $I$ is a subset of $J$, hashing with chaining is used for the hash table, and the hash table size is $N$? Explain which part of the algorithm is associated with each term in your formula for run time.

5. This question is the same as Question 3, except that now we use the following algorithm. We put each element of $I$ into the hash table and then look up each element of $J$. As we look up each element we record in the hash table that the element of $I$ was found. Finally, we look up each element of $I$ to check whether it was found. Again, give a formula for the time and explain which part of the algorithm is associated with each term in your formula for run time.

6. 

a. For the algorithms of the previous two questions, which is faster? (One potential answer is that the times are so close that one cannot tell without programming each algorithm.) Consider both the case where $I$ is a subset of $J$ and the case where $I$ is not a subset of $J$. It is important to justify your answer to this part.

b. In some cases, one can tell that $I$ is not a subset of $J$ just using the value of $i$ and $j$. Explain how this idea can be used. Does adding this idea to each algorithm change the answer to Part a.