Unit 2 Discussion Question Solutions

1. Suppose that algorithm A takes $1000n^3$ steps and algorithm B takes 2^n steps (Note the carot symbol ^ means raise to the power of which we use here because we cannot create the appropriate mathematical symbol in Moodle) for a problem of size n. For what size of problem is algorithm A faster than B (meaning algorithm A has fewer steps than B)? In your answer describe not only what the answer is but how you arrived at the answer.

This is an interesting problem and one that can easily be solved by putting the two equations into a spreadsheet and incrementing the value of n. What we find is that for all values of n from 1 to 23, the number of steps in the equation $1000n^3$ is actually larger than the equation 2^n. However, at n=24 the equation 2^n becomes larger.

2. Give the upper bound (big O notation) that you can for the following code fragment, as a function of the initial value of n.

```
for(int i = 0; i < n; i++) {
    for(int j = 0; j < i; j++){
        //do swap stuff, constant time
    }
}
```

In this case the outer loop has time of n because the loop will loop between 0 and the value of n. The inner loop will loop between 0 and i. The first time the inner loop is called the value of i will be the same as the value of n, however upon each execution of the loop i will get smaller. This might make us think a little bit, however, Big O notation is about the upper bound, the worst case analysis so we ignore the reduction of I and look at this problem as n * n iterations through the loop which means that it is n^2 or $O(n^2)$.