CS1332, Fall 2011

Homework #1 Big-O and complexity

1. Find n0 such that the running time of an algorithm which has n2 operations is greater than one which has 500n operations.
   1. **N = 501**
2. Bob and Sally are arguing. Bob says his algorithm is O(n) and must therefore always be faster than Sally’s algorithm which is O(n3). Sally says that is wrong, and in fact her algorithm is sometimes faster. Who is right and why?
   1. **Sally is correct. The O(n) and O(n3) does not take into account constants and other lower order terms that may alter each function. For example, Bob’s algorithm’s may be actually 5000n and Sally’s may be 2n3. So for some range of small n, Sally’s algorithm may be faster up to some no.**
3. Solve problem 10 on page 77 of the textbook (Code analysis).
   1. **O(n2)**
   2. **O(n2)**
   3. **O(nlogn)**
   4. **O(nlogn)**
4. What is the Big-O of the following runtimes and note whether they are constant, logarithmic, linear, n log n, quadratic, polynomial, or exponential:
   1. 4nlog n + 2n – **O(nlogn) NlogN**
   2. 210 **- O(1) Constant**
   3. 3n + 100log n **– O(n) Linear**
   4. 4n **– O(n) Linear**
   5. 2n **- O(2**n**) Exponential**
   6. n2 + 10 n **– O(n2) Quadratic**
   7. n3**- O(n3) Cubic**
   8. log n + 5000 **– O(logn) Logarithmic**
   9. 2048 **– O(1) Constant**
   10. n! ­**– O(n!) Factorial**