**Data Structures Exam 1 Review**

**Big O Notation**

* Super Factorial
* Factorial
* Exponential
* Polynomial
* n log(n)
* n
* log(n)
* Constant

**Data Structures**

Heap:

Insert and remove: O(log(n))

Singly LinkedList:

Any Operation: O(n)

Stack (LIFO):

Push and Pop: Constant

Queue (FIFO):

Push and Pop without Tail: O(n)

Push and Pop with Tail: Constant

BST:

Worst Case: O(n)

Balanced BST:

O(log(n))

**Pseudocode**

LinkedList Get and Set

Stack or Queue Pop and Push

Two Line Pseudocode for Heaps:

Insert:

1. Insert at next position
2. Upheap

Remove:

1. Replace root with tail
2. Downheap

Heapify: Making a Heap tree out of a list of numbers

**Amortized Complexity**

AKA “Average” Complexity

If you have a list that doubles every time is fills up, what is the amortized complexity?

O(2) – Constant Time

**NP Complete**

Non-deterministic Polynomials

Any NP Complete problem can be transformed into any other NP Complete problem in Polynomial Time (called reduction)

Find Examples of NP Complete Problems (Ex. Traveling Salesman, \_)