**CSCI 132**: Basic Data Structures and Algorithms

Midterm Study Guide

**Logistics**

* Wednesday, March 8th @ 3:10 PM in Norm Asbjornson Hall 165
* Time length: 50 minutes (but I will allow you to work for an extra 20-25 minutes since nobody uses the classroom after us). This exam is designed to be completed in 35-40 minutes.
* Open notes. You are allowed to use your laptop, your IDE, any notes, slides, lecture examples. This exam can be completed without a laptop.
* You are NOT allowed to use the internet to access external resources (Google, Stack Overflow, W3 Schools, etc)
* The midterm exam will consist of different types of question, such as:
  + Multiple choice questions
  + True/False
  + Short answer
  + What is wrong with this code?
  + What is the difference between X and Y
  + Complete the line of code so that X happens.

**Content**

The following topics are all fair game for the midterm exam.

* Basic Java Classes, OOP
* Operations and variables
* Methods
* If statements
* Loops
* Arrays
* Inheritance
* References
* Static Methods
* Abstract Classes, Interfaces
* Polymorphism
* ArrayLists
* LinkedLists (Singly, Doubly, Circular)
* Growth Rates
* Big-O Notation, How to determine theoretical running time of an algorithm

Sample Exam Questions

(Coming Soon)

1. Consider the following Java Class

**public** **class** Duck {

**private** **String** name;

**public** Duck() {

**this**.name = n;

}

}

1. What does the **private** keyword mean?
2. Consider the following line of code in a demo class:

Duck don = **new** Duck("Donald");

This line of code results in an error. Why is this error occurring?

1. Rewrite the constructor below so that the line of code from part B work correctly
2. What is an **interface** in Java?
3. What will the following code print out?

**int** myArray[] = {1, 2, 3, 4, 5};

**for**(**int** i = 0; i < myArray.length - 1; i++) {

System.***out***.println(myArray[i+1] \* 2 );

}

1. True/False. To add a node to the very end of a doubly linked list, we must traverse the entire linked list first.
2. The code below prints out an N x N multiplication table.

**public** **void** print\_table(**int** n) {

**for** (**int** i =1; i <= n; i++){

**for** (**int** b=1; b <= n; b++){

System.***out***.print(i\*b + " ");

}

System.***out***.println();

}

}

1. For each instruction in the function, clearly mark/label the running time of that operation.
2. What is the total running time (in Big-O) of this function?
3. What is the running time of printing out a Circular Linked List?
   1. O(1)
   2. O(N)
   3. O(N2)
   4. O(XN)
4. What is the difference between an Array, and an ArrayList?

1. Consider this basic Node class that is used in a circular linked list.

**public** **class** Node {

**private** String name;

**private** Node next;

**private** Node prev;

**public** Node(String c) {

**this**.name = c;

**this**.next = **null**;

**this**.prev = **null**;

}

**public** Node getNext() {

**return** **this**.next;

}

**public** Node getPrev() {

**return** **this**.prev;

}

**public** **void** setNext(Node newNode) {

**this**.next = newNode;

}

**public** **void** setPrev(Node newNode) {

**this**.prev = newNode;

}

1. A new node needs to be inserted into the circular linked list before the head. Supply the code below so that a new node (newNode) will get added. You can assume that the circular linked list has at least 1 node already in it. The linked list class does keep track of the head node, and tail node.

**public** **void** addBeforeHead(Node newNode){

}

1. What is the running time of your algorithm from part A (You must state your answer in Big-O notation)