name ___________________

CS 112 – Data Structures

Summer, 2015

Midterm Exam 1

- You have **90 minutes** for this exam.
- **Do not open this exam** until everyone has one and the instructor says to begin.
- There are 5 pages in this exam, including this one. Make sure you have them all.
- This exam is closed book – closed notes – closed electronics. You must put your cellphone, PDA, ipod, or other electronic devices in a backpack, etc, and leave it out of your reach. The only exception is that you can use a watch that only has time-related functions (e.g. not a "smart" watch or a calculator watch).
- When you turn in your exam, you must show your Rutgers ID card with a legible picture, or else your driver's license.
- Write clearly – if we can't read or can't find your answer, your answer is wrong.
- Make clear what is your answer versus intermediate work.

For graders use only:

1		/20
2		/20
3		/20
4		/20
5		/20
6		/20
7		/20
total		/140

name ___________________

1. For each of the following functions give its simplest big-O equivalent. E.g. for F(n) = 5 the answer would be O(1)
   a. $F(n) = n^3 + 15n$ is O(______________)
   b. F(n) = n*(n+1)*(n+2) / 4 is O(______________)
   c. $F(n) = 2^n + n^2$ is O(______________)
   d. $F(n) = 2^{n+10}$ is O(______________)
   e. $F(n) = 2^{4*n}$ is O(______________)

2. Suppose you had a singly linked list class as below and you want to write a method interleave in this LinkedList class. If list1 held data "A", "B", and "C" and list2 held data "X", "Y", and "Z", list1.interleave(list2) would return a new list containing "A", "X", "B", "Y", "C", and "Z" in that order, and leave list1 and list2 unchanged. Finish interleave. It must **not** be recursive. You may assume list1 and list2 have the same length.

```
public class Node{
    public String data;
    public Node next;
    public Node (String data, Node next){ // constructor for Node
        this.data = data;   this.next = next;
    }
}
public class LinkedList{
    public Node head;
    public LinkedList ( ){  // constructor for LinkedList
        this.head = null;
    }
    public LinkedList interleave(LinkedList list2){
```

3. Suppose you wanted to write interleave as a static **recursive** method in the Node class above (rather than the LinkedList class as for question 2), so that if you had a variable node1 → “A” → “B” → null and variable node2 → “C” → “D” → null then interleave(node1, node2) would return a reference → “A” → “C” → “B” → “D” → null, and leave the original lists unchanged. Finish this version of interleave. It **must** be recursive. You may assume the two lists are the same length.

```
public static Node interleave(Node list1, Node list2){
```

4. Suppose you have generic circular linked list and node classes as below. Finish the method twiddle in the CLL class. Let us say that “twiddling” a list means interchanging its first two elements. E.g., if a list is A, B, C, D then twiddling it makes it be B, A, C, D. Note that your method may not read or write the data field of a Node. Twiddling a list with zero or one node does nothing.

```
public class Node<T>{
     private T data;      // data may only be accessed from the Node class
     public Node<T> next;
}
public class CLL<T>{
    Node<T> last;               // reference to the last Node in the list
    int size;                   // number of nodes in the list

    public void twiddle( ){     // interchanges the first two nodes in the list
       if (size < 2) {
           return;              // if the list is empty or has one element, do nothing
      } else if  (size == 2){   // size == 2 is a special case
                                                          // fill in here

      } else {                  // there are at least 3 nodes
                                                          // and fill in here
```

5. In homework assignment 2, you had classes Person and Friend. Person had three instance variables: Person nextPerson, String name, and Friend firstFriend. A Friend object represented one friendship relation, and had two instance variables: Person who and Friend nextFriend. Suppose the following method was in class Person. p1.sameFriends(p2) return true if person p1 has the same friends as p2, in the same order. Finish sameFriends. You may **NOT** assume the Friend lists are the same length.

   public boolean sameFriends(Person p2){

6.

   a. Suppose you start with an empty stack and do the following operations. For each pop, show the data that would be returned by the pop. If any operation causes an error, say so, and assume the stack is not changed.

      Push 1, push 2, pop: result _____, push 3, pop: result _____, push 4,

      pop: result _____, pop: result _____, pop: result _____.

   b. Do the same as part a above but for a queue.

      Enqueue 7, dequeue: result: ___, enqueue 8, enqueue 9,

      dequeue: result ____ enqueue 10, enqueue 11, dequeue: result: ___ ,

      enqueue 12, dequeue: result: ___ , dequeue: result: ___

name ___________________

7. Suppose you have a class, Queue, with only three public methods: int dequeue( ), void enqueue(int n), and boolean isEmpty( ), and a constructor Queue that returns a new empty queue. Finish the following "client" method (i.e. it is *not* in the Queue class, but in the program that uses a Queue):

```
public int size(Queue q) { // returns the number of ints currently in q
```