CS 307 – Midterm 2 – Spring 2010

Name__________________________________________

UTEID login name _______________________________

TA's Name:       Guillermo          Reza  Shyamala  (Circle One)

Instructions:
1. Please turn off your cell phones and other electronic devices.
2. There are 5 questions on this test.
3. You have 2 hours to complete the test.
4. You may not use a calculator on the test.
5. When code is required, write Java code.
6. When writing methods, assume the preconditions of the method are met. Do not write code to check the preconditions.
7. On coding question you may add helper methods if you wish.
8. After completing the test please turn it in to one of the test proctors and show them your UTID.

1. (2 points each, 30 points total) Short answer. Place you answers on the attached answer sheet.
   • If the code contains a syntax error or other compile error, answer “compile error”.
   • If the code would result in a runtime error / exception answer “Runtime error”.
   • If the code results in an infinite loop answer “Infinite loop”.

Recall that when asked for Big O your answer should be the most restrictive correct Big O function. For example Selection Sort has an average case Big O of O(N^2), but per the formal definition of Big O it is correct to say Selection Sort also has a Big O of O(N^3) or O(N^4). I want the most restrictive correct Big O function. (Closest without going under.)

A. Allen Rabinovich of Yahoo was a guest speaker in our class. What did Allen recommend about commenting code and programs you write for your own use? What was the basis of his recommendation?

B. Allen is the Senior Architect for the Yahoo User Interface (YUI), a library of utilities and controls for building interactive web applications. The library is open source, meaning the source code is available for all programmers to use and extend in any way they wish. Allen made the claim that making YUI open source was actually a selfish thing for Yahoo to do. Explain, according to Allen, why making YUI open source is actually a selfish thing to do.
C. What is returned by method `c(5)`?

```java
public int c(int x){
    if(x <= 0)
        return 3;
    else
        return x + c(x - 2);
}
```

D. What is output by the method call `d("leakey")`?

Recall the substring method:

```java
public String substring(int beginIndex)
{
    Returns a new string that is a substring of this string. The substring begins with the character at the specified index and extends to the end of this string.
}
```

```java
public void d(String s){
    if(s.length() <= 2)
        System.out.print(s);
    else{
        System.out.print(s.charAt(0));
        d(s.substring(2));
        System.out.print(s.charAt(s.length() - 1));
    }
}
```

E. What is returned by the method call `e(6)`?

```java
public int e(int x){
    if(x <= 3)
        return x;
    else
        return 2 * x + e(x - 1) + e(x - 3);
}
```

F. What is the Big O of method `f`? `N = list.length`.

```java
public int f(int[] list){
    int total = 0;
    for(int i = 0; i < list.length; i++)
        total += list[i];
    for(int i = 0; i < list.length; i += 3)
        total += list[i];
    return total;
}
```
G. What is the worst case Big O of method g? N = list.length.

```java
class Solution {

g(int[] list) {
    int total = 0;
    int limit = list.length - 1;
    for (int i = 0; i < limit; i++) {
        if (list[i] == list[i + 1]) {
            for (int j = i + 1; j < list.length; j++) {
                total += list[j] * i;
            }
        }
    }
    return total;
}
```

H. What is the Big O of method h?

```java
class Solution {

h(int N) {
    int[][] result = new int[N][N];
    return result;
}
```

I. What is the best case Big O of method i? N = data.length.

```java
class Solution {

i(double[] data) {
    double result = 0;
    for (int i = 0; i < data.length; i++) {
        for (int j = 1; j < data.length; j *= 2) {
            if (data[j] == data[i]) {
                result += data[j];
            }
        }
    }
    return result;
}
```

J. What is the Big O of method j? N = org.length. The ArrayList constructor is O(1).

```java
class Solution {

j(double[] org) {
    ArrayList<Double> result = new ArrayList<Double>();
    for (int i = 0; i < org.length; i++) {
        result.add(0, org[i]); // 0 is the position to insert in the list
    }
    return result;
}
```

K. What is the Big O of method showAll? N = list.size(). Assume the print method is O(1).

```java
class Solution {

showAll(LinkedList<Integer> list) {
    for (int i = 0; i < list.size(); i++) {
        System.out.print(list.get(i));
    }
}
```
L. A sorting method uses the insertion sort algorithm. It takes 7 seconds for the method to sort 50,000 distinct ints in random order. What is the expected time for the method to sort 150,000 distinct integers?

M. A method is $O(N \log_2 N)$. It takes 10 seconds for the method to run when $N = 500,000$. What is the expected run time when $N = 2,000,000$. Recall $\log_2 1,000,000 \approx 20$

N. A method is $O(N)$. When $N = 50,000$ the method takes 20 seconds to complete. Given 5 seconds how many pieces of data can the method process?

O. You have a 1,000,000 pieces of data that are unsorted. You expect to do 50,000 searches on the data before it changes. Should you sort the data with merge sort and then perform the 50,000 binary searches or just do 50,000 linear searches without sorting? Justify your answer mathematically.
2. (Array based lists, 20 points) Write a method for the GenericList class we developed in lecture that adds a given element after every occurrence of a target element.

- The GenericList is generic based on Java's inheritance requirement and polymorphism, not the Java generic syntax.
- GenericList's internal storage is a native array of Objects. There may be extra capacity in the array.
- The array variable named elements will never equal null.
- The elements of the list use 0 based indexing. The position of an element in the list is equal to its index in the array.
- You may not use any other methods from the GenericList class unless you implement them yourself in this question.
- You may not use any other Java classes except native arrays and the equals method from Object.

Complete the following method in the GenericList class:

```java
public void insertAfter(Object newValue, Object tgt) {
    // Implement this method on the next page:
}
```

Examples of results of insertAfter:
[A, B, A, C].insertAfter(D, A) -> [A, D, B, A, D, C]
[].insertAfter(D, A) -> []
[A, B, A, C].insertAfter(D, G) -> [A, B, A, C]
[A, A, A, A].insertAfter(D, A) -> [A, D, A, D, A, D, A, D]

public class GenericList {
    private Object[] elements;
    private int size;

    // complete this method
    public void insertAfter(Object newValue, Object tgt) {
```
// complete this method
public void insertAfter(Object newValue, Object tgt){
    assert newValue != null && tgt != null
    && !newValue.equals(tgt);
3. (Linked Lists 15 points). Write a method for a LinkedList class that determines how many of the nodes in the list are storing null as their data. You may not use any other methods from the LinkedList class unless you implement them yourself in this question. Your method is to be O(1) space meaning you cannot use a temporary native array or ArrayList. The LinkedList class has the following properties:

- The list uses singly linked nodes that store one piece of data and a reference to the next node in the list.
- The list maintains references to the first and last nodes in the list
- If the list is empty the references to the first and last nodes are set to null.
- The list does not maintain a size instance variable.
- The last node's next reference is set to null.

When we implemented the add method for our LinkedList class we did not disallow users of the list to add nulls. So for example the following client code is allowed:

```java
LinkedList<String> list = new LinkedList<String>();
list.add(null); // not an error
list.add(null);
list.add("Kelly");
list.add(null);
list.add("Olivia");
```

After the above code executes the list would have 5 values, 3 of which are null. Given the above list your method would return 3.

Here is the Node class:

```java
public class Node<E> {
    public Node(E value, Node<E> next)

    public E getData()
    public Node<E> getNext()

    public void setValue(e value)
    public void setNext(Node<E> next)
}
```

Here is the LinkedList class:

```java
public class LinkedList<E> {

    private Node<E> first;
    private Node<E> last;

    // Complete the method on the next page:
```
// pre: none (This method is in the LinkedList class.)
// post: return the number of data elements in this list that are
// equal to null. This list is not altered as a result of this
// method call.
public int numNull() {
4. (Using data structures, 15 points) Recall the `UnsortedSet` class you implemented on an assignment. Write a method that takes an `UnsortedSet<Rectangle>` and removes all Rectangles from the `UnsortedSet` that have an area greater than a given value. The removed Rectangles are added to a resulting `UnsortedSet` that is returned by the method.

You may not use any other classes or methods except those listed below.

The methods you may use from the `UnsortedSet` class are shown below:

```java
public class UnsortedSet<E> {
    // create a new UnsortedSet
    public UnsortedSet() {
    }

    // add an element to this UnsortedSet. Return true if this
    // UnsortedSet changed as a result of this method call
    public boolean add(E obj) {
    }

    // return an iterator for this UnsortedSet
    public Iterator<E> iterator() {
    }
}
```

Recall the methods from the `Iterator` interface:

```java
public boolean hasNext() {
    Returns true if the iteration has more elements.
}

public E next() {
    Returns the next element in the iteration.
}

public void remove() {
    Removes from the underlying collection the last element returned by the iterator
}
```

The only method you need from the `Rectangle` class is:

```java
public double getArea() {
    Returns the area of this Rectangle object.
}
```

Complete the following method on the next page. Recall, this method is not part of the `UnsortedSet` class.

```java
// pre: set != null
// post: remove all elements from set that have an area greater
// than or equal to a. The elements removed from set are returned
// in a new UnsortedSet
public UnsortedSet<Rectangle> removeBigRectangles
    (UnsortedSet<Rectangle> set, double a) {
```
// pre: set != null
// post: remove all elements from set that have an area greater
// than or equal to a. The elements removed from set are returned
// in a new UnsortedSet
public UnsortedSet<Rectangle> removeBigRectangles
    (UnsortedSet<Rectangle> set, double a) {

5. (Recursion, 20 points) Given a set \( S \), the power set of \( S \) is the set of all subsets of \( S \). For example if we have this set:

\[
\{C, A, G\}
\]

then the power set of that set is

\[
\{\{\}, \{C\}, \{A\}, \{G\}, \{C, A\}, \{A, G\}, \{C, G\}, \{C, A, G\}\}
\]

Write a helper method for an instance method for the UnsortedSet class that returns the power set of the calling object.

```java
class UnsortedSet<E> {
    private ArrayList<E> con;

    // create an empty UnsortedSet
    public UnsortedSet() {
        // add an element to this set. Return true if the set changed
        // as a result of this method
        public boolean add(E obj) {
            // pre: none
            // post: return the power set of this set. This set is not
            // changed as a result of this method call.
            public UnsortedSet<UnsortedSet<E>> getPowerSet() {
                UnsortedSet<UnsortedSet<E>> result =
                    new UnsortedSet<UnsortedSet<E>>();
                ArrayList<E> subset = new ArrayList<E>();
                helper(result, subset, con);
                return result;
            }
        }
    }
}
```

Here are the methods from the ArrayList class you can use:

```java
public boolean add(E e)
    Appends the specified element to the end of this list.

public E get(int index)
    Returns the element at the specified position in this list.

public int size()
    Returns the number of elements in this list.

public E remove(int index)
    Removes the element at the specified position in this list.
```
// Complete the helper method below so that the getPowerSet method
// works correctly. Recall this method is in the UnsortedSet class.
private void helper(UnsortedSet<UnsortedSet<E>> result,
    ArrayList<E> subset, ArrayList<E> originalCon) {

Name: ________________________________

TA's Name: Guillermo Reza Shyamala (Circle One)

Answer sheet for question 1, short answer questions

A. __________________________________________  H. __________________________________________

B. __________________________________________  I. __________________________________________

C. __________________________________________  J. __________________________________________

D. __________________________________________  K. __________________________________________

E. __________________________________________  L. __________________________________________

F. __________________________________________  M. __________________________________________

G. __________________________________________  O. __________________________________________