Points off	1	2	3	4	5A	5B	Total off	Net Score

CS 314 – Exam 1 – Fall 2017

Your Name_________________________________

Your UTEID _________________________________

Circle your TAs Name: Gilbert Jacob Jorge Joseph Lucas Rebecca Shelby

Instructions:

1. There are **5** questions on this test. 100 points available. Scores will be scaled to 150 points.
2. You have 2 hours to complete the test.
3. Place your final answers on this test. Not on scratch paper. Answer in pencil.
4. You may not use a calculator or any other electronic devices while taking the test.
5. When answering coding questions, ensure you follow the restrictions of the question.
6. Do not write code to check the preconditions.
7. On coding questions, you may implement your own helper methods.
8. On coding questions make your solutions as efficient as possible given the restrictions of the question.
9. Test proctors will not answer any questions regarding the content of the exam. If you think a question is ambiguous or has an error, state your assumptions and answer based on those assumptions.
10. When you complete the test show the proctor your UTID, give them the test and all the scratch paper, used or not, and leave the room quietly.

1. (1 point each, 15 points total) Short answer. Place your answer on the line next to or under the question. Assume all necessary imports have been made.

   a. If a question contains a syntax error or other compile error, answer **compile error**.
   b. If a question would result in a runtime error or exception, answer **runtime error**.
   c. If a question results in an infinite loop, answer **infinite loop**.
   d. Recall 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. The following method takes 2 seconds to complete when N = 10,000. What is the expected time to complete when N = 30,000? Assume `Math.random()` is O(1).

____

```
public static ArrayList<Double> methodA(int N) {
     ArrayList<Double> result = new ArrayList<>(N);
     for (int i = 0; i < N; i++) {
          double val = Math.random();
          if (val < .25)
               result.add(0, val);
          else
               result.add(val);
     }
     return result;
}
```

B. What is the worst case order (Big O) of `methodB`? N = `list.size()` ____________________

Assume `ArrayList.size()` is O(1). Assume methods `check` and `process` are O(N) where N is the size of the list sent to the methods. Methods `check` and `process` do not alter the list in any way.

```
public static int methodB(ArrayList<Integer> list) {
     int count = 0;
     for (int i = 0; i < list.size(); i++) {
          for (int j = 0; j < list.size(); j++) {
               if (check(list, i, j)) {
                    count += process(list, i, j);
               }
          }
     }
     return count;
}
```

C. If the `ArrayList.size()` method were O(N) instead of O(1), what would the worst case order of `methodB` from question 1.B be? N = `list.size()`.

____

D. The following method takes 5 seconds to complete when `data.length` = 50,000 and all elements of `data` contain the `char tgt`. What is the expected time for the method to complete when `data.length` = 150,000 and all elements of `data` contain the `char tgt`?
Assume the length of the `String`s in `data` is small relative to the length of `data`.

____

```
// pre: no elements of data == null
public static ArrayList<String> methodD(String[] data, char tgt) {
     ArrayList<String> result = new ArrayList<>();
     for (int i = 0; i < data.length; i++) {
          String temp = data[i];
          if (temp.indexOf(tgt) != -1) {
               result.add(temp);
          }
     }
     return result;
}
```

E. A method is O(2^N). It takes 1 second for the method to complete when N = 50.
What is the expected time in seconds for the method to complete when N = 54?

____

F. A method is O($N^2\log_2 N$). It takes 20 seconds for the method to complete when N = 1,000,000. What is the expected time in seconds for the method to complete when N = 2,000,000?

____

G. What is output by the following code if `list`, an `ArrayList<String>`, initially contains the values `["C", "E", "B", "D", "A", "Z", "M"]`?

```
list.remove(1);
list.remove(3);
list.add(2, list.get(3));
list.add(list.size() + "");
list.remove(3);
System.out.print(list);
```

____

H. What is output by the following code? The `Map.toString` returns elements in the form `{key1=value1, key2=value2, ..., keyN=valueN}.`

```
Map<Integer, String> m = new TreeMap<>();
m.put(12, "M");
m.put(5, "A");
m.put(12, "D");
m.put(-5, "A");
m.put(0, m.get(12));
m.remove(10);
System.out.print(m);
```

____

I. What is output by the following code if `list2`, an `ArrayList<Integer>`, initially contains the values `[3, 5, -5, 4, 1, 10, -5]`?

```
Iterator<Integer> it = list2.iterator();
int total = it.next();
total += it.next();
total += it.next();
it.remove();
it.next();
it.next();
it.remove();
total += list2.get(2);
System.out.print(total + " " + list2);
```

____

<u>For questions J through O, refer to the classes defined on the sheet at the back of the test. You may detach that sheet for easier reference.</u>

J. What is output by the following code? ___________________________________

```
Ship s1 = new CruiseShip(100);
System.out.print(s1 + " " + s1.crew());
```

K. What is output by the following code? ___________________________________

```
Ship s2 = new CruiseShip(100);
System.out.print(s2 + " " + s2.getCap());
```

L. What is output by the following code? ___________________________________

```
Object obj = new Warship(10, 100);
System.out.print(obj);
```

M. What is output by the following code? ___________________________________

```
CruiseShip cs = new CruiseShip(1000);
System.out.print(cs.crew() + " " + cs.getCap());
```

N. What is output by the following code? ___________________________________

```
Ship s3 = new Yacht();
System.out.print( ((Warship) s3).weaps() + " " + s3.toString());
```

O. Assume we want to change the class header for the Ship class to the following:

```
public abstract class Ship implements Comparable<Ship> {
```

What are we required to do so that the `Ship` class compiles with the new class header?

____

For questions J - O, consider the following classes**.**

```
public abstract class Ship {

      private int cap;

      public Ship(int c) { cap = c; }

      public abstract int crew();

      public String toString() {return "S: " + getCap(); }

      public int getCap() { return cap; }
}


public class CruiseShip extends Ship {

      public CruiseShip (int c) {  super(c); }

      public int crew() { return getCap() / 2; }
}


public class Warship extends Ship {

      private int num;

      public Warship (int c, int n) {
            super(c / 2);
            num = n;
      }

      public int crew() { return num; }

      public int weaps() { return num / 10; }

      public int getCap() { return num + 100; };
}


public class Yacht extends Ship {

      public Yacht() { super(10);      }

      public int crew() { return 5; }

      public String toString() { return "Ahoy!"; }
}
```

2. The GenericList class (15 points) To demonstrate encapsulation and the syntax for building a class in Java, we developed a GenericList class that can store elements of any data type. Recall our GenericList class stores the elements of the list in the first N elements of a native array. An element's position in the list is the same as the element's position in the array. The array may be larger than the list it represents.

Complete a method that removes the first n values from the calling list.

```
/*   pre: 0 < n <= size()
     post: The first n elements of this list are removed.
     Remaining elements are shifted to the front of the list.
     size() = old size() - n
*/
public void removeFirstN(int n) {
```

Examples of calls to the removeFirstN method. (The values shown are String objects).
-> indicates the calling list after the method has completed.

```
[A, B, C, D, A].removeFirstN(1) -> [B, C, D, A]

[A, B, C, D, A].removeFirstN(2) -> [C, D, A]

[A, B, C, D, A].removeFirstN(5) -> []
```

The GenericList class:

```
public class GenericList<E> {

     private E[] con;
     private int size;
```

You may not use any methods from the GenericList class unless you implement them yourself as a part of your solution.

Complete the method on the next page.

```
/*   pre: 0 < n <= size()
     post: The first n elements of this list are removed.
     Remaining elements are shifted to the front of the list.
     size() = old size() - n
*/
public void removeFirstN(int n) {
```

3. GenericList (20 points) This question uses the same `GenericList` class as described in question 2.

Create an instance method for the `GenericList` class `inOtherList`. The method creates and returns a new `GenericList` that contains all the elements in the calling list that are also present in another list passed as a parameter.

Unlike the permutation problem from assignment 1, if there are multiple equal elements in the calling list, and only one instance of that element in the other list, all copies of the equal elements in the calling list are added to the result. Example method calls and lists returned by the method. Assume all elements shown are `String`s.

```
[].inOtherList([A, B, C]) -> returns []

[A, B, A, C].inOtherList([]) -> returns []

[A, B, A, C].inOtherList([C, A]) -> returns [A, A, C]

[A, B, A, C, C].inOtherList([C, A, B]) -> returns [A, B, A, C, C]

[A, B, A, C, C].inOtherList([Z, X, Z, Y]) -> returns []

[A, B, A, C, C, A].inOtherList([A]) -> returns [A, A, A]

[A, B, A, C, C, A].inOtherList([A, B, A]) -> returns [A, B, A, A]

[A, B, D, C, C].inOtherList([A, A, X, A]) -> returns [A]
```

The `GenericList` class:

```
public class GenericList<E> {

      private E[] con;
      private int size;

      public GenericList(int initialCapacity) {
            con = (E[]) (new Object[initialCapacity]);
      }
```

You may only use the constructor from the `GenericList` class shown above.

You may not use any other methods from the `GenericList` class unless you implement them yourself as a part of your solution. You may not use any other classes besides `GenericList` and native arrays.

You may call the `equals` method on objects.

You may assume none of the elements of either list equal `null`.

Complete the method on the next page.

```
// pre: other != null, no elements of this or other equal null
// Neither this or other are altered as a result of this method.
// post: per the problem description
public GenericList<E> inOtherList(GenericList<E> other) {
```

4. Baby Names (25 points) Based on an idea by Jose R. from the summer 2013 CS314 class. Write an instance method named `getComebackNames` for the `Names` class from assignment 3. The method returns an `ArrayList<String>` of names that have made a ***comeback***. A comeback is defined as a name that is ranked in the top 1000 in the initial decade (and possibly following decades), drops out of the top 1000 for at ***least three decades in a row***, and then returns to the top 1000 for at least one decade.

Examples of names that have made a comeback based on the given criteria:

Abbie	431	552	742	924	**0**	**0**	**0**	**0**	752	644	601
Caesar	956	843	**0**	**0**	**0**	946	0	0	0	0	0
Catalina	571	566	482	507	784	**0**	**0**	**0**	**0**	873	745
Charity	642	889	**0**	**0**	**0**	**0**	**0**	277	284	494	463

Examples of names that have not made a comeback based on the given criteria:

Cary	796	0	998	936	409	338	307	443	0	0	0
Cassandra	0	0	0	0	0	0	1000	758	531	523	723
Delilah	923	785	0	1000	0	805	0	0	892	0	679
Mystery	0	999	0	0	0	0	975	0	0	0	0

Note, the names Mystery and Cassandra in the example above do not meet the criteria for a comeback because they are not ranked in the initial decade for the data set.

Here are the `Names` and `NameRecord` classes for this question:

```
public class Names {
     private ArrayList<NameRecord> names;
}

public class NameRecord {
     public String getName() // return this NameRecord's name
     public int getNumRanks() // return the number of ranks

     // pre: 0 <= decade < getNumRanks(), returns rank for give decade
     // or 0 (zero) if not ranked in given decade
     public int getRank(int decade)
}
```

Do not use any other methods from the `NameRecord` class.

You may create and use an `ArrayList<String>`. You may use any methods from the `ArrayList` class.

Complete the following instance method for the `Names` class on. Do not use any other methods in the `Names` class unless you implement them yourself. Do not alter the `names` field.

```
// pre: getNumRanks >= 5 for all NameRecords in this Names object.
// post: per the problem description
public ArrayList<String> getComebackNames()
```

Complete the method on the next page.

```
// pre: getNumRanks >= 5 for all NameRecords in this Names object.
// post: per the problem description
public ArrayList<String> getComebackNames()
```

5. Maps, part A (13 points) Write a method to create and return a Map with college admissions statistics. The method is passed a set of parallel arrays. The first array is an array of Strings and the second array is an array of booleans. For example:

```
["A&M", "Rice", "UT", "Rice", "UT", "A&M", "UT", "Rice", "UT", "UT",  "A&M"]
[true,  false,  false, true,   true, true,  true, false,  true, false, true]
```

Each String represents the name of a college. The corresponding element (the one at the same index as the String) in the array of booleans indicates if a given student was offered admission (true) or not (false) to that school. The first entries in the arrays indicate one student that was accepted at A&M. The second entries indicate one student that was not accepted to Rice.

All elements in the array of Strings shall be non-null.

In part A, complete a method that returns a Map<String, int[]>.

The keys are the names of the colleges.

The values are arrays of length 2.

The first element indicates the number students admitted to the college.

The second element indicates the total number of students that applied to the college.

You must create and use a single Map whose keys are Strings and whose values are native arrays of ints with length 2 in your solution.

You may use the following methods from the Map interface.

put(K, V)	adds a mapping from the given key to the given value if key already present, replaces old value with given value
get(K)	returns the value mapped to the given key (null if none)
containsKey(Object)	returns true if this key is present in the map, false otherwise

Do not use any other methods from the Map interface.

You may create and use arrays of ints with length 2.

Complete the method on the next page.

```
// pre: schools != null, schools.length > 0, results != null,
//    results.length == schools.length, no elements of schools = null
// post: per the problem description
public Map<String, int[]> getStats(String[] schools, boolean[] results) {
```

5B. (12 points) Complete a method that determines which school has the lowest acceptance rate based on percentage. For example, given the arrays from part A, we have these results

A&M: 3 admits, 0 denials -> 100% acceptance rate
Rice: 1 admit, 2 denials -> 33.3% acceptance rate
UT: 3 admits, 2 denials -> 60% acceptance rate.

The method returns the `String` that represents the college with the lowest acceptance rate based on percentage. If there is a tie for lowest acceptance rate, pick the school that had the most total applications. If there is still a tie, you may arbitrarily pick any of the tied schools.

In addition to the methods from part A you may call the `keySet` method for `Map`s that returns a `Set` of the keys. You may call the `iterator` method on the key set and use all of the methods from the `Iterator` interface.

```
// pre: map is a Map of the form returned by the method in part A
// post: per the question description, map is not altered
public String getMostSelectiveSchool(Map<String, int[]> map) {
```

For questions J - O, consider the following classes**. You may detach this sheet from the test.**

```
public abstract class Ship {

      private int cap;

      public Ship(int c) { cap = c; }

      public abstract int crew();

      public String toString() {return "S: " + getCap(); }

      public int getCap() { return cap; }
}


public class CruiseShip extends Ship {

      public CruiseShip (int c) {  super(c); }

      public int crew() { return getCap() / 2; }
}


public class Warship extends Ship {

      private int num;

      public Warship (int c, int n) {
            super(c / 2);
            num = n;
      }

      public int crew() { return num; }

      public int weaps() { return num / 10; }

      public int getCap() { return num + 100; };
}


public class Yacht extends Ship {

      public Yacht() { super(10);      }

      public int crew() { return 5; }

      public String toString() { return "Ahoy!"; }
}
```