Points off	1	2	3	4			Raw Points Off

Your Name: ____

Your UTEID: ____

Circle your TA's Name: **Aman** **Bersam** **Brayden** **Casey** **Devon** **Eliza** **Gracelynn** **Lauren** **Namish** **Nidhi** **Pavan**

Instructions:

1. There are **4** questions on this test. 100 points available. Scores shall be scaled to 250 points.
2. You have 2 hours to complete the test.
3. Place your final answers on this test. Not on the scratch paper. **Answer in pencil**. Exams not completed in pencil are not eligible for a regrade. You may use highlighters on the exam.
4. You may not use a calculator or **outside resources of any kind** while taking the test. Please remove any smart watches and put them and any mobile devices away.
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 2 and 3, you may implement your own helper methods. Not on 4.
8. On coding questions make your solutions as efficient as possible given the restrictions of the question.
9. **Test proctors shall not address 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 and give them the test. Please place used and used scratch paper in the appropriate boxes at the front of the room. Please leave the room quietly.

1. (2 points each, 50 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 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 shall be the most restrictive correct Big O function. For example, Selection Sort is average case O($N^{2)}$, but per the formal definition of Big O it is correct to say Selection Sort is O(N^3) , O(N^4) and so forth.
   Give the most restrictive, correct Big O function. (Closest without going under.)
e. Assume `log`$_2$`(1,000) = 10` and `log`$_2$`(1,000,000) = 20.`

A. What is returned by the method call **`a(9)`**?

```
public static int a(int x) {
    if (x <= 3) {
        return x;
    }
    int t = (x % 3) + 1;
    return 2 + a(x - t);
}
```

____

B. What is returned by the method call **`b(8, 0)`**? (*By Nidhi*) _______________

```
public static int b(int x, int y) {
    if (x <= 1) {
        return y;
    } else if (x % 2 == 0) {
        return y + b(x - 3, y + 1);
    } else {
        return x + b(x - 1, y + 1);
    }
}
```

C. What is returned by the method call **`c(7)`**? ____________________

```
public static int c(int x) {
    if (x <= 2) {
        return 1;
    } else  {
        return 1 + c(x - 2) + c(x - 4);
    }
}
```

D. What is output by the following code? ____________________

```
int[] count = {0};
d(4, count);
System.out.print(count[0]);

public static int d(int x, int[] count) {
    count[0]++;
    if (x <= 0)
        return 1;
    return 2 + d(x - 1, count) + d(x - 1, count);
}
```

E. Assume we need to use a **`java.util.Map`** in our code. We want operations on the map involving the key to be as fast as possible. What kind of map should we use? Pick the letter of the best answer.

```
A. java.util.HashMap
B. java.util.TreeMap
C. could use either java.util.HashMap or java.util.TreeMap
```

____

F. What is output by the following code? It uses the **`java.util.TreeMap`** class. Recall the output of the **`toString`** method for Java maps.
`{key_1=value_1, key_2=value_2, ..., key_n=value_n}`
(*By Gracelynn*)

____

```
TreeMap<String, Integer> map = new TreeMap<>();
map.put("N", 5);
map.put("L", 6);
map.put("E", 5);
map.put("G", 9);
map.put("A", map.get("G"));
map.put("S", map.remove("L"));
map.put("E", 7);
System.out. println(map);
```

G. Why does the following class not compile? (*By Gracelynn*)

____

```
public class Student {

    private String name;
    double gpa;

    public abstract void calculateGPA();

    public static void main(String[] args) {
        Student s = new Student();
        System.out.println(s.name);
        System.out.println(s.toString());
    }
}
```

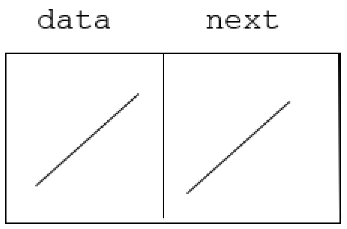
H. The following method takes 1 second to complete when **`t1.size()`** = 10,000. What is the expected time for the method to complete when **`t2.size()`** = 20,000? (*By Lauren)*

____

```
// pre: t1.size() == t2.size(). Uses Java's LinkedList.
public static int g(LinkedList<Double> t1, ArrayList<Double> t2) {
    int t = 0;
    for (int i = 0; i < t2.size(); i++) {
        double a = t2.get(i);
        for (int j = 0; j < t1.size(); j += 2)
            if (t1.get(j).equals(a))
                t += t1.get(j);
    }
    return t;
}
```

I. Consider the following **Node** class:.

```
public class Node {
    public Object data;
    public Node next;
}
```



Consider the following code. Draw the variables, references, and objects that exist after the following code executes. Draw node objects as shown above (to the right of the **Node** class code) and boxes for variables. The example has all instance variables set to **null**.

```
Node n1 = new Node();
n1.next = new Node();
n1.next.next = n1;
n1.next.data = new int[2];
```

J. Assume we have a **java.util.LinkedList** that stores **String** objects is sorted order. Which search algorithm is better, typically, if we want to search the linked list efficienctly without creating any new data structures? Pick the letter of the best answer.

A. binary search ________________
B. linear search
C. Could use either binary search or linear search

K. The following code takes 5 seconds to complete when **list.size()** = 10,000. What is the expected time for the code to complete when **list.size()** = 20,000?
Uses the **java.util.ArrayList** class, ______________

```
public static double k1(ArrayList<Double> list) {
    double r = 0.0;
    while (list.size() > 0) {
        r += list.get(0);
        list.remove(0); // remove based on position
    }
    return r;
}
```

L. The following code takes 5 seconds to complete when **list.size()** = 100,000. What is the expected time for the code to complete when **list.size()** = 200,000?
Uses the **java.util.LinkedList** class, ______________

```
public static double k2(LinkedList<Double> list) {
    double r = 0.0;
    while (list.size() > 0) {
        r += list.get(0);
        list.remove(0); // remove based on position
    }
    return r;
}
```

M. What is output by the following code? It uses the **`Stack314`** and **`Queue314`** classes developed in lecture.

```
Queue314<Integer> q = new Queue314<>();
Stack314<Integer> st = new Stack314<>();
for (int i = 0; i < 6; i++) {
    q.enqueue(i);
    st.push(i);
}
int t = 0;
while (!q.isEmpty())
    if (q.dequeue() == st.pop())
        t++;
 System.out.print(t);
```

____

N. We are going to store user input to a computer system in a data structure so that the system can respond to the input. User input consists of things such as keyboard presses, mouse events (clicks, movement, scrolls), and possible touchscreen events.

What kind of data structure would we most likely use? Pick the letter of the best answer.

A. stack
B. queue
C. could use either stack or queue

____

O. The following method takes 10 seconds to complete when **`list.size()`** = 6,000,000 and **`data.length`** = 25,000. Assume each element in **`data`** is present once in **`list`**. What is the expected time for the method to complete when **`list.size()`** = 2,000,000 and **`data.length`** = 150,000. Again, each element in **`data`** is present once in **`list`**.

____

```
public static int o(ArrayList<Integer> list, Integer[] data) {
    int t = 0;
    for (int i = 0; i < data.length; i++) {
        if (list.contains(data[i])) {
            t++;
        }
    }
    return t;
}
```

P. Which sort, insertion sort or quicksort, will typically be faster when sorting the data as described? Sorting into ascending order. length can be quite large (*By Eliza*) (1 point each)

All elements in the array are equal. _______________________

All elements distinct, sorted in descending order. _______________________

Q. Which of the following line(s) of code would cause a syntax error? ____________________

```
// By Bersam
Map<String, Object> map = new HashMap<>(); // 1
map = new TreeMap<>(); // 2
map = new Map<>(); // 3
map.put("CS314", new Object()); // 4
map.get("Namish").toString();  // 5
map.put("Casey", "Pavan"); // 6
map.get("Casey").toLowerCase(); // 7
```

R. Assume we want to sort a large array of objects. Typically, the elements of the array are not in any particular order based on our current sorting criteria. Of the sorts we studied (selection, insertion, radix, quick, merge) which one would most likely be used?

____

S. The following method takes 2 seconds to complete when **`list.size()`** = 1,000,000. What is the expected time for the method to complete when **`list.size()`** = 2,000,000? All elements of the list have a **`length`** of 20 or less. Uses the **`java.util.LinkedList`** class.

```
public static int s(LinkedList<String> list) {
    int t = 0;
    while (list.size() > 0)
        t += list.remove(list.size() - 1).length();
    return t;
}
```

____

T. Method **`s`** from question 1.S is altered so that it uses the **`LinkList`** class developed in lecture instead of the **`java.util.LinkedList`** class. The method takes 2 seconds to complete when **`list.size()`** = 100,000. What is the expected time for the method to complete when **`list.size()`** = 200,000? All elements of the list have a **`length`** of 20 or less.

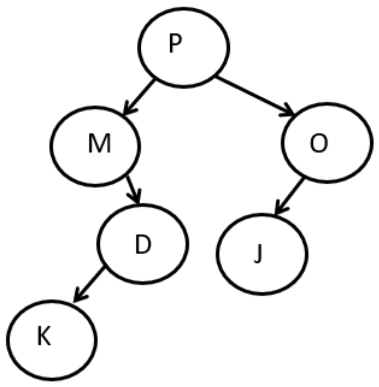
____

U. In a complete binary tree with 19 nodes how many nodes does the deepest level of the tree contain?

____

V. What is the result of a post order traversal of the binary tree shown to the right?

____



W. What is the result of a pre order traversal of the binary tree shown to the right?

____

X. The binary tree shown above is NOT a full binary tree. What is the minimum number of nodes required to be added to the tree to make it a full binary tree? No nodes are removed.

____

Y. What is the height of the tree created by the code below? It uses the **binary search tree** class developed in lecture which uses the simple add algorithm. (*By Brayden*)

____

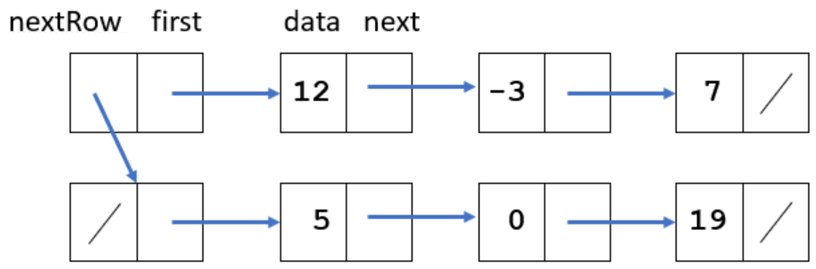
```
String[] names = {"Devon", "G-lynn", "Brayden", "Aman", "Pavan",
     "Lauren", "Namish", "Nidhi", "Lauren", "Eliza", "Bersam",
     "Sumaya"};

BST<Integer> t = new BST<>();
for (String n : names) {
    t.add(names.length());
}
```

2. **Linked Lists.** (18 points) (*By Aman)* Imagine implementing a matrix with a linked list like structure. Each row is a singly linked list with a header node. The header node for each row does **not** contain any data. Each header node has 2 instance variables. A link to the first node with data in the row and a link to another header node (if it exists), the header node for the next row.

Consider this abstract representation of a matrix:

12	-3	7
5	0	19

The concrete version of the above matrix is shown by the linked structure to the right. The first node in each "row" is a **`RowHeader`** node and each element with an integer value is a **`DataNode`**. Note, the data elements of the **`DataNode`**s are references to objects, but the illustration has been simplified by placing the data inside the **`data`** instance variables as if they were value variables. / in the drawing indicates a variable that stores **`null`**.



Complete a private instance method for the **`LinkedMatrix`** class that returns **`true`** if the linked structure is a ***rectangular matrix***, **`false`** otherwise. **Recall, a rectangular matrix is one in which every row has the same number of elements. In other words, every row has the same number of columns. An empty matrix is considered rectangular. Likewise, if the number of rows is greater than 0, but all rows have 0 elements, the matrix is also considered rectangular.**

The **`LinkedMatrix`** class for this question:

```
public class LinkedMatrix<E> {
     /* refers to the header node for the first row. Stores null if
        matrix is empty. */
    private RowHeader<E> firstRow;

    private static class RowHeader<E> {
        private RowHeader nextRow; // null if last row header node
        // Refers to data node with first element or null if row empty
        private DataNode<E>  first;
    }

    private static class DataNode<E> {
        private E data;
        private DataNode<E> next; //Next element, stores null if none.
    }
}
```

Note, it is possible that the matrix is empty, in which case **`firstRow`** stores **`null`**. It is also possible some rows have 0 elements and the **`first`** variable of the **`RowHeader`** stores **`null`**.

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

Do not use recursion. Do not create any new data structures. Do not alter the calling object. You may not use any other Java classes or methods.

```
// pre: None. post: Per the problem description.
private boolean isRectangular() {
```

3. **Maps (18 points)** Complete a method that determines from a given **`Map`** of people and the activities they enjoy, the number of activities that some given number of people or more enjoy.

The Map passed to this method has keys that are **`String`**s and values that are **`Set`**s of **`String`**s. Example:

Keys (String)	Values (Set of Strings)
Gracelynn	(Teaching, Programming, Working, Canoeing, Making Donuts)
Namish	(Programming, Checking Ed, Teaching, Skydiving, Volleyball)
Brayden	(Skydiving, Canoeing, Kayaking, Chess, Programming)
Aman	(Checking Ed, Programming, Teaching, Breakdancing, Wordle)
Lauren	(Traveling, Teaching, Programming, Volleyball, Line Dancing)

If the given number of people was 4 and the map was as shown above, the method returns 2.

There are 2 activities that 4 or more people enjoy: Teaching (Gracelynn, Namish, Aman, Lauren) and Programming (all 5).

Your solution can create and use a single **`Map`**. (**`TreeMap`** or **`HashMap`**)

You may us the following methods:

<u>From the **`Map`** interface:</u>

```
public Set<K> keySet()
public boolean containsKey(K key)
public V get(K key)
public V put(K key, V val)
public V remove(K key)
```

<u>From the **`Set`** interface:</u>

```
public Iterator<E> iterator
```

You may use the **`hasNext`**, **`next`**, and **`remove`** methods from the **`Iterator`** interface either explicitly or implicitly.

Do not use any other Java classes or methods other than those listed above.

Do NOT use recursion in your solution.

The `Map` passed in as a parameter is not to be altered.

```
/* pre: map != null, minPeople >= 2
  post: per the problem description. map is not altered by this method.*/
public static int getNum(Map<String, Set<String>> map, int minPeople) {
```

4. **Recursive Backtracking** (14 points) Complete a recursive helper method that determines if ***frequent flier miles*** on one airline can be redeemed at another airline. Frequent flier programs are offered by airlines to reward travelers for flying a lot. (Sidetrack: Airlines provide many ways of earning miles other than actually flying. For example, there are credit cards where a user earns one frequent flier mile for every dollar put on the credit card.) One feature of these programs is that one airline will accept and honor some other airlines' frequent flyer miles. but not every airline. This is referred to as a ***codeshare agreement*** or ***partner airline***. For example, assume **United** airlines honors miles from these other airlines:

Alaska Airlines, Hawaiian Airlines, Qantas, Singapore Airlines, WestJet

United accepts frequent flier miles from these airlines and in turn, all these airlines accept frequent flier miles from **United**.

Now assume **Alaskan Airlines** has codeshare agreements with the following airlines:

Air France, British Airways, Delta, French Bee, Japan Airlines, Korean Air, Qantas, United, WestJet.

Theoretically if we had frequent flier miles on **Delta** we could convert them to **Alaskan Airlines** miles and then to **United** miles. It can get even more convoluted. For example, if miles on **Singapore Airlines** and wanted to convert them to **Japan Airlines** miles we could go **Singapore Airlines** -> **United** -> **Alaskan Airlines** -> **Japan Airlines**

Use the following `Airline` class for this question.

```
public class Airline {
    // Return an array of partner airlines for this Airline.
    public Airline[] getPartners()

    // Overrides equals. Return true if this object equals other.
    public boolean equals(Object other)
}
```

Here is the kickoff method:

```
// pre: org != null, dest != null
public static boolean canConvert(Airline org, Airline dest) {
    Set<Airline> tried = new HashSet<>();
    return helper(org, dest, tried);
}
```

Of course, you may use the `length` field of arrays and the provided `Airline` class.

You may use the `add` and `contains` methods for `Sets`.

Your helper must use recursion.

Do not create any new data structures. Do not alter any of the arrays returned by `getPartners`.

You may NOT add any other helper methods.

```
private static boolean help(Airline org, Airline dest,
                            Set<Airline> tried) {
```