Points off	1	2	3	4			Total off	Net Score

Your Name: _____

Your UTEID: _____

Circle your TA's Name: **Anthony Ashley David K. David T. Grace Henry Lilly Neal Nina Noah Pranav Skyler Sam**

Instructions:

1. There are **4** questions on this exam. 100 points available. Scores will be scaled to 250 points.
2. You have 2 hours to complete the exam.
3. Place your final answers on this exam. Not on the scratch paper. **Answer in pencil**.
4. You may not use a calculator or **outside resources of any kind** while taking this exam.
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. Exam 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 exam show the proctor your UTID, give them the exam and all the scratch paper, used or not, and 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 r1(0, 5)? _____

```
public static int r1(int x, int y) {
    if (y <= 2)
        return x;
    return x * y + r1(x + 1, y - 1);
}
```

B. What is returned by the method call r2(4)? _____

```
public static int r2(int x) {
    if (x <= 1)
        return 3;
    return 1 + r2(x - 1) + r2(x - 2);
}
```

C. What is output by the following client code? _____

```
int[] count = {0};
r3("", count);
System.out.println(count[0]);

public static int r3(String s, int[] count) {
    count[0]++;
    if (s.length() >= 3) {
        return s.length();
    } else {
        int r = 0;
        for (int i = 0; i < 3; i++) {
            r += r3(s + i, count);
        }
        return r;
    }
}
```

D. What is returned by the method call r4(8)? _____

```
public static int r4(int x) {
    if (x <= 0)
        return -1;
    return -1 * r4(x - 2);
}
```

E. What is output by the method call r5("NINA")? _____

```
public static void r5(String s) {
    if (s.length() <= 1) {
        System.out.print(s);
    } else {
        System.out.print(s.charAt(1));
        r5(s.substring(1));
        System.out.print(s.length());
    }
}
```

F. What is the order of method empty shown below? N = list.size(). The LinkedList314 class is the same as the one implemented in lecture using singly linked nodes.

_____

```
public static void empty(LinkedList314<String> list){
    while (list.size() != 0) {
        list.remove(list.size() - 1);
    }
}
```

G. What is the order of method empty shown below? N = list.size(). The LinkedList class is the Java implementation of a linked list that uses doubly linked nodes.

_____

```
public static void empty(LinkedList<String> list){
    while (list.size() != 0) {
        list.remove(list.size() - 1);
    }
}
```

H. What is the order of method empty shown below? N = list.size(). The LinkedList314 class is the same as the one implemented in lecture using singly linked nodes.

_____

```
public static int part(LinkedList314<Integer> list) {
    int r = 0;
    for (int i = 1; i < list.size(); i *= 2) {
        r += list.get(i);
    }
    return r;
}
```

I. In class we implemented a linked list using singly linked nodes. We maintained a reference to the last node in the linked structure of nodes. Which of the following method calls would have a different average case order (Big O) if the class did not have the reference to last node in the linked structure of nodes? Circle all of the methods that would have a different average case order.

add(E val) insert(int pos, E val) get(int pos) remove(int pos)

J. What is output by the following code? _____

```
Stack<Integer> st = new Stack<>();
for (int i = 1; i <= 4; i++) {
    st.push(i);
    st.push(i * 3);
}
for (int i = 0; i < 4; i++) {
    System.out.print(st.pop() + " ");
}
```

K. Consider the array based GenericList and the LinkedList using singly linked nodes we developed in lecture. Which of the following statements is true? List all letters of true statements.

_____

A. LinkedLists are faster than GenericLists for all operations.
B. LinkedLists always uses less memory than GenericLists.
C. Providing an iterator improved the order (big O) of traversing GenericLists and LinkedLists.
D. GenericLists always uses less memory than LinkedLists.

L. What is output by the following code? The size method returns the number of elements in the Queue.

_____

```
Queue<Integer> q = new Queue<>();
for (int i = 5; i >= 1; i--) {
    q.enqueue(i);
}
for (int i = 0; i < q.size(); i++) {
    System.out.print(q.dequeue() + " ");
}
```

M. What is the order of the method shown below? N = data.length. The sort method called uses the insertion sort algorithm presented in class. The array data contains N distinct elements in random order.

_____

```
public static ArrayList<Integer> get(int[] data) {
    ArrayList<Integer> result = new ArrayList<>();
    for (int x : data) {
        result.add(x);
        sort(result);
    }
    return result;
}
```

N. What is the order of the get method shown below? N = data.length. The sort method called uses the mergesort algorithm presented in class. The array data contains N distinct elements in random order.

_____

```
public static ArrayList<Integer> get(int[] data) {
    ArrayList<Integer> result = new ArrayList<>();
    for (int x : data) {
        result.add(x);
        sort(result);
    }
    return result;
}
```

O. What is output by the following code? _____

```
TreeMap<Integer, Integer> m = new TreeMap<>();
m.put(5, -2); // key, value
m.put(0, -2);
m.put(7, 3);
m.put(0, 5);
m.put(5, 4);
System.out.print(m); /* Recall the toString of Maps:
            {key1=value1, key2=value2, ... keyN=valueN} */
```

P. What is the order of method shown below? N = list.size(). The LinkedList class is the Java implementation of a linked list that uses doubly linked nodes. Assume the implementation uses a header node and is circular.

_____

```
public static int tot(LinkedList<Integer> list) {
    int r = 0;
    for (int i = list.size() - 1; i >= 0; i--) {
        r += list.get(i);
    }
    return r;
}
```

Q. Consider the following timing data for a method that sorts arrays of ints. Of the sorts we studied in class which one does the method **most** likely use based on the timing data?

_____

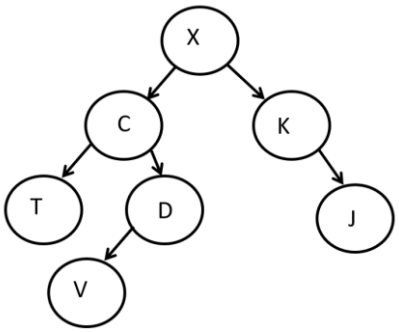
Number of elements in array	Time to sort array of N distinct elements in random order.	Time to sort array of N elements that all equal the same value.
1,000,000	10 seconds	10 seconds
2,000,000	21seconds	21 seconds
4,000,000	44 seconds	44 seconds

R. The following values are inserted one at a time in the order shown, left to right, into an initially empty binary search tree using the simple add algorithm demonstrated in class. What is the depth of the node in the resulting tree that contains 17?

_____

```
37  73  119  0  13  0  28  17
```

S. What is the result of a post order traversal of the tree to the right?



_____

T. What is output by the following code?
Recall the toString of Maps: {key1=value1, key2=value2, ... keyN=valueN}

_____

```
int[] data = {5, 2, 3, 5, 2};
TreeMap<Integer, Integer> m = new TreeMap<>();
for (int i = 0; i < data.length; i++) {
    m.put(data[i], i);
}
System.out.print(m);
```

U. What is output by the following code? _____

```
LinkedList<String[]> list = new LinkedList<>();
list.add("array 1");
String[] n1 = {"A", "CC"};
list.add(n1);
for (String[] s : list)
    System.out.print(s.length + " ");
```

V. What is the order of the following method? N = the parameter n. The Java TreeSet class uses a binary search tree to store the elements of the set. Assume the Math.random() method is O(1).

_____

```
public static TreeSet<Double> get(int n) {
    TreeSet<Double> tr = new TreeSet<>();
    for (int i = 0; i < n; i++) {
        tr.add(Math.random());
    }
    return tr;
}
```

W. Which of the following lists can be easily used as the internal storage container for a queue such that all of the queue operations are O(1).

_____

A. A circular, doubly linked list
B. A singly linked list with references to the first and last node.
C. An array based list.
D. A singly linked list with a reference to the first node only.

X. The following code takes 10 seconds to complete when N = 1,000,000. What is the expected time for the code to complete when N is set to 2,000,000 instead? The binary search tree uses the simple insertion algorithm shown in class, implemented iteratively, not recursively.

```
final int N = 1_000_000;
BST314<Integer> t = new BST314<>();
for (int i = N; i >=0; i--)
    t.add(i);
```

Y. What is returned by the method call **recy(0)?** _____

```
public static int recy(int x) {
    if (x == 1)
        return 3;
    return recy(x - 1) + 2 + recy(x - 1);
}
```

EXTRA CREDIT: If we hav an account where the user name was "Student" then the password would be:

_____

2. Maps (18 points) Complete a method that updates a **`Map<String, List<String>>`** with the authors of a new academic paper and returns the total number of times the first and last authors of the paper have collaborated on papers including this one.

Modern academic papers typically have multiple authors. For example, here are the authors of paper by UTCS's own William Cook:

The Design and Implementation of Object Grammars

Tijs van der Storm[a,*], William R. Cook[b], Alex Loh[b]

Centrum Wiskunde & Informatica (CWI), Science Park 123, 1098 XG Amsterdam, The Netherlands
[b] University of Texas at Austin, 1 University Station, Austin, Texas 78712, US

A map with Strings for keys and values of Lists of Strings is used to store authors and the authors they have collaborated with. All authors are represented by their last name. Assume last names are unique. The key is the authors last name and the value is a List of Strings the author has collaborated with.
Here are some examples of some of the key-value pairs stored in the map:

Storm -> [Cook, Loh]
Cook -> [Storm, Loh, Wang, Dillig, Lahiri]
Dillig -> [Chen, Wang, Fan, Cook, Wang, Geffen, Torlak, Wang]

The list stores the collaborators in reverse chronological order, with the most recent collaborator at the beginning of the list. So for example, Cook most recently collaborated with Storm. Dillig most recently collaborated with Chen. Note, the list of collaborators contains a name for each time an author has collaborated with the other author. So Dillig has collaborated with Wang 3 times.

If there is a tie for the most recent collaborator (because they all worked on the same paper) the order they appear relative to each other does not matter. Cook collaborated with Storm and Loh on the paper shown above and the names could appear in either order. (Storm, Loh OR Loh, Storm)

The method takes 2 parameters: the map of authors (as described above) and an array of Strings.
The array of Strings represents the authors of a new paper. The method updates the map based on the authors of the paper.

If a paper has a single author, the array shall be of length 1 and no changes are made to the map.

The method returns the number of times the first author of the paper (the first element of the array) and the last author of the paper (the last element of the array) have collaborated, including this time. If the paper has a single author the method returns 0.

So for example, if the map shown above (recall only a portion of the map is shown) is passed as a parameter and the array is [Dillig, Novak, Cook] is passed as the authors of the new paper, the map becomes:

Storm -> [Cook, Loh]
Cook -> [Dillig, Novak, Storm, Loh, Wang, Dillig, Lahiri]
Dillig -> [Novak, Cook, Chen, Wang, Fan, Cook, Wang, Geffen, Torlak, Wang]
Novak -> [Cook, Dillig]

and the method would return 2.

Complete the following method:

```
/* pre: map != null, authors != null, authors.length >= 1, none of the
elements of authors are null
post: per the problem description. The array is not altered as a
result of this method. */
public static int updateAuthors(Map<String, List<String>> map,
            String[] authors) {
```

You may use the following methods from the Map Interface:

V put(K, V), V get(Object), boolean containsKey(K), Set<K> keySet(), V remove(Object), int size().

You may create and use Iterator objects (implicit or explicit) for the set returned by the keySet method.

You may create Java LinkedLists and use any methods from the Java LinkedList class.

You may use the equals method on objects. Of course, you may use the array length field.

Do not use any other Java classes or methods. Do not create any new native arrays.

Do not use recursion.

Complete the method on the next page.

```
public static int updateAuthors(Map<String, List<String>> map,
            String[] authors) {
```

3. (18 points) Linked Lists. Complete the following instance method for the LinkedList314 class:

```
/* pre: other != null. Neither this or other are empty lists.
In other words, it is guaranteed that this and other have at
least one element each. post: per the problem description.
Neither this or other are altered. */
public LinkedList314<E> interleave(LinkedList314<E> other) {
```

The method creates and returns a new **LinkedList314** with elements from the calling object and other interleaved as shown below.

```
[A, B, C].interleave([D, E, F]) -> returns [A, D, B, E, C, F]

[A].interleave([D, E, F]) -> returns [A, D, E, F]

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

[A, B, C, D].interleave([Q, K]) -> returns [A, Q, B, K, C, D]

[X, Y].interleave([K, C, D, M]) -> returns [X, K, Y, C, D, M]
```

The **LinkedList314** class for this question.

```
public class LinkedList314<E> {

    private Node<E> first; // first == null iff list is empty
    /* If the list isn't empty the last node in the linked
       structure of nodes has its next reference set to null. */

    public LinkedList314() { first = null; }

    private static class Node<E> {
        private E data;
        private Node<E> next;
        private Node(E data) { this.data = data; }
```

Recall the precondition that each list has at least one element. Neither list will be empty.

You may not use any methods from the LinkedList314 class other than the given constructor unless you implement them as a part of your solution. Do not add any instance or class variables to the LinkedList314 class.

Do not use any other Java classes or methods except the nested Node class. Do not add any methods or instance variables to the Node class.

Do not use recursion.

Complete the method on the next page.

```
/* pre: other != null. Neither this or other are empty lists.
In other words, it is guaranteed that this and other have at
least one element each. post: per the problem description.
Neither this or other are altered. */
public LinkedList314<E> interleave(LinkedList314<E> other) {
```

4. Recursive Backtracking (14 points) Complete a method that uses recursive backtracking to that determine what words can be formed by reducing a **`String`**.

In a comment at the end of your answer, state what you think the order (Big O) of your method is. N = word.length(). This is worth 1 point.

```
/* pre: word != null, all chars in word are lower case English
letters. d != null, all elements of d consist of lower case English
letters. results != null, results.size() == 0
d is not altered as a result of this method. */
public static void findWords(String word, Set<String> d,
                             List<String> results)
```

Reducing a **`String`** means removing one character from the **`String`**, but maintaining the same relative order of the other characters.

For example, the four reductions of "camp" are "amp", "cmp", "cap", and "cam". Note amp, camp, and cam are typically considered actually words but cmp is not.

Note, we will consider a word a reduction of itself. So "camp" is considered a reduction of "camp".

Obviously the words that can be obtained by reducing a **`String`** vary based on the dictionary used. Here is one set of words obtained by reducing "computer" for a given dictionary.

[come, compute, computer, cop, cope, copter, cot, cote, cue, cut, cute, cuter, me, mute, muter, op, opt, or, our, out, outer, per, put]

A **`Set<String`**> is passed as a parameter and is the dictionary used to determine if a given **`String`** is a word or not. If a String is in the dictionary it is considered a word, otherwise it is not.

We use a **`List<String>`** that is initially empty to store all the reductions of the initial **`String`** that are words.

You may use the following methods:

```
String class: int length(), String substring(int start),
                            String substring(int start, int end)

Set class: boolean contains(Object val)

List class: add(E val), boolean contains(Object val)
```

The **`List<String>`** with results shall not contain any duplicates.

You must use recursive backtracking.
You may NOT add any helper methods on this question.
You may not alter the method header or parameter list for the method. (You cannot add parameters to or remove parameters from the findWords method. No class variables either.)
You may not use any Java classes or methods besides those listed above.

```
/* pre: word != null, all chars in word are lower case English
letters. d != null, all elements of d consist of lower case English
letters. results != null, results.size() == 0
d is not altered as a result of this method. */
public static void findWords(String word, Set<String> d,
                             List<String> results)
```

// What is the order of your method?