Points off	1	2	3	4			Raw Points Off

Your Name: _____

Your UTEID: _____

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. **This exam shall be entirely your own work.** You may **not** use **outside resources of any kind** while taking the test. **Please silence your mobile devices**. 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, 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 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 unused 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(7)`**?

_____

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

B. Using the techniques and rules from lecture, what is the T(N) of the following method? N = the parameter **n** _____

```
public static double b(int n) {
    double t = 0;
    for (int i = 1; i <= 4; i++)
        for (int j = 1; j <= n; j++)
            for (int k = 1; k <= n; k++)
                t += 1.0 * (j * i) / k;
    return t;
}
```

C. What is output by the following code? _____

```
final int LIMIT = 1024;
int t = 0;
for (int i = 1; i <= LIMIT; i *= 2) {
    for (int j = 1; j <= i; j++) {
        t++;
    }
}
System.out.print(t);
```

D. In the code from question C the line that initializes **LIMIT** is altered to

```
final int LIMIT = 1_000_000;
```

The code is timed and takes .0001 seconds on a given computer. The code is changed again to

```
final int LIMIT = 4_000_000;
```

and rerun on the same machine under the same conditions.
What is the expected time for the code to complete now? _____

E. A method that sorts an array of non-negative **ints** uses the radix sort algorithm. It takes the method 1 second to sort an array of 500,000 distinct elements, the largest of which is 987,654. What is the expected time for the method to sort an array of 3,000,000 distinct elements the largest of which is 123,456,789?

_____

F. What is output by the following code? _____

```
ArrayList<Integer> list1 = new ArrayList<>();
list1.add(5);
ArrayList<Integer> list2 = list1;
list2.add(4);
f(list1, list2);
System.out.print(list1.size() + " " + list2.size());

public static void f(ArrayList<Integer> list1,
     ArrayList<Integer> list2) {

     list1.add(3);
     list2.add(6);
     list2 = new ArrayList<>();
     list1.add(12);
     list2.add(9);
     System.out.print(list1.size() + " " + list2.size() + " ");
}
```

G. The following method takes 15 seconds to complete when **n** = 1,000,000. What is the expected time for the method to complete when **n** = 2,000,000? The sort method called uses the **insertion sort algorithm** demonstrated in class.

_____

```
public static int[] g(int n) {
    int[] result = new int[n];
    for (int i = 0; i < result.length; i++) {
        result[i] = n - i;
    }
    sort(result); // into ascending order
    return result;
}
```

H. Method **h** takes 3 seconds to complete when **list.size()** is 20,000. What is the expected time for the method to complete when **list.size()** is 60,000? **list** is a **LinkedList314** object. Recall the **LinkedList314** used singly linked nodes and has references to the first and last nodes in the linked structure of nodes.

_____

```
public static void h(LinkedList314<String> list) {
     final int LIMIT = list.size() / 2;
     for (int i = 0; i < LIMIT; i++) {
          list.remove(list.size() - 1);
     }
}
```

I. The following values are inserted one at a time in the order shown left to right into an initially empty Red Black tree using the insertion algorithm demonstrated in lecture. What is the result of a pre order traversal of the resulting tree?

1 8 1 8 6 7 _____

J. What is output by the following code? The **TreeMap** class is the **java.util.TreeMap** class**.** Recall the output of the **toString** method from the **Map** interface: **{key_1=value_1, key_2=value_2, ..., key_n=value_n}**

_____

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

K. What is output by the code? The **Stack314** class is the one we developed in lecture.

_____

```
Stack314<String> stk = new Stack314<>();
String[] data2
     = {"Lauren", "Casey", "Eliza", "Bersam", "Brayden",
    "Aditya", "Aman", "Ahmad", "Nidhi", "Namish", "Gracelynn"};

for (String s : data2) {
     if (stk.isEmpty() || s.length() >= stk.top().length()) {
         stk.push(s);
     }
}

while (!stk.isEmpty())
     System.out.print(stk.pop().length() + " ");
```

L. We want to implement a traditional fair queue and we want the enqueue and dequeue operations to be average case O(1). Which of the following internal data structures can be used to implement the queue to meet these requirements? List all correct choices,

_____

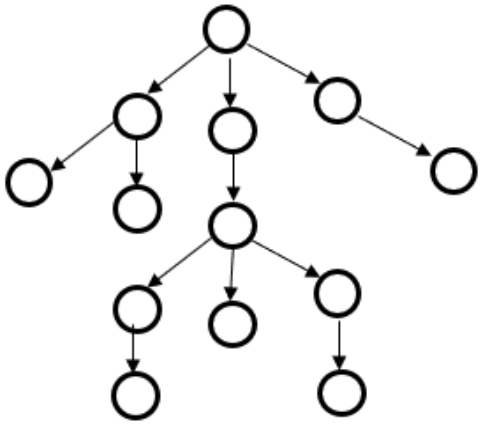
A. A doubly linked list that is circular and has a header node.
B. A singly linked list with a reference to just the first node in the structure.
C. A singly linked list with a reference to the first and last node in the structure.
D. A **java.util.HashSet.**
E. A native array of **Object**s using the techniques demonstrated in the lecture on queues.

M. The following method takes 0.1 seconds to complete when **n** = 1,000,000. What is the expected time for the method to complete when **n** = 4,000,000. The method uses the **java.util.HashSet** class. Assume the **Random.nextInt** method is O(1).

_____

```
public static HashSet<Integer> m(int n, Random r) {
     HashSet<Integer> result = new HashSet<>();
     final int LIMIT = n / 2;
     for (int i = 0; i < n; i++)
          result.add(r.nextInt(LIMIT));
     return result;
}
```

N. Consider the tree to the right. Assume it stores a Trie. Every node except the root stores a character. What is the **maximum** number of words (valid sequences) this Trie could represent?



_____

O. How many bits does it take to encode a Huffman Code Tree with 10 leaf nodes using the Standard Tree Format (STF) from assignment 10?

Do not include the 32 bits for the size of the tree that are written to the header before the bits that actually describe the tree, only the bits needed to represent the tree using the specification from assignment 10.

_____

P. Recall the two header formats specified for assignment 10, Huffman Coding. The Standard Count Format and the Standard Tree Format. For each format, as specified in the assignment, is it possible (answer **Yes** or **No**) to have a file that is **too large** for the header format to work? Consider only the header formats, not the specifics of your implementation of assignment 10. (1 point each)

SCF: _____ STF: _____

Q. Consider the following method. **data** contains 10,000 positive integers. The **BST314** class uses the simple add algorithm demonstrated in lecture. What is the minimum possible height and maximum possible height for the resulting binary search tree? The BST314 class uses the simple add algorithm demonstrated in lecture. (1 pt. each)

min: _____ max: _____

```
public static BST314<Integer> q(Set<Integer> data) {
     BST314<Integer> result = new BST314<>();
     for (Integer i : data)
          result.add(i % 7);
     return result;
}
```

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 insertion algorithm demonstrated in lecture. What is the result of a post order traversal of the resulting tree?

1 8 1 8 6 7 _____

S. The following values are inserted in the order shown, left to right, to an initially empty **max heap**. What is the result of a level order traversal of the resulting max heap?

17 3 12 8 8 17

_____

T. The following method takes 10 seconds to complete when **n** = 1,000,000. What is the expected time for the method to complete when **n** = 4,000,000. The **BST314** class uses the simple insertion algorithm demonstrated in lecture. Assume **Random.nextInt** in O(1).

_____

```
public static BST314<Integer> t(int n, Random r) {
    BST314<Integer> t = new BST314<>();
    for (int i = 0; i < n; i++) {
        t.add(r.nextInt());
    }
    return t;
}
```

U. Yes or No, all common computing problems have a Dynamic Programming Solution.

_____

V. When iterating through the keys of a Java **HashMap** the keys are in no discernible order. When iterating through a Python 3.7 **dictionary** (a synonym for map) the keys are presented in insertion order. The Python 3.7 **dictionary** uses a hash table as its main internal storage container. Which of the following data structures would most likely be used in conjunction with the hash table to allow iterating through the keys in insertion order? Pick the single best choice.

_____

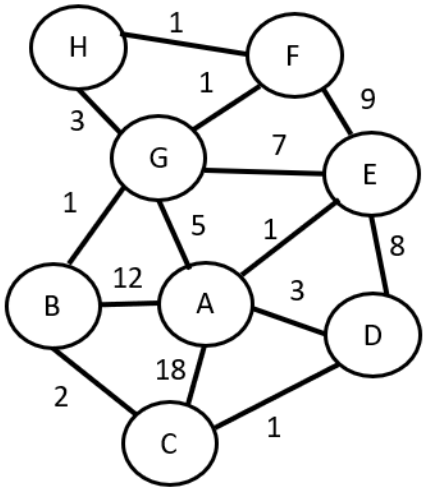
A. a doubly linked list
B. a stack
C. a priority queue
D. a fair queue
E. a binary search tree

W. Recall the given **Graph** class for assignment 11. Yes or No, could that **Graph** class be used, with no changes to represent a multi-graph, a graph that allows multiple edges between a pair of vertices?

_____

X. Consider the weighted, undirected graph to the right.

We perform Dijkstra's algorithm as demonstrated in lecture to find the shortest paths from **H** to all other vertices in the graph **H** is connected to. What are the first 6 vertices marked as visited (added to the set of vertices for which we have found the shortest path) when Dijkstra's algorithm is performed? H is the first, what are the next 5?



H _____

Y. What is output by the following code? _____

```
int[] result =
        IntStream.of(11, 40, 3, 22, 12, 24, 37)
        .filter(n -> n > 10 && n % 2 == 0)
        .map(n -> n / 2)
        .filter(x -> x % 2 == 0)
        .map(x -> x / 2)
        .toArray();
System.out.print(Arrays.toString(result));
```

Extra Credit the First: 1 point: What was the name of the submarine your instructor served on?

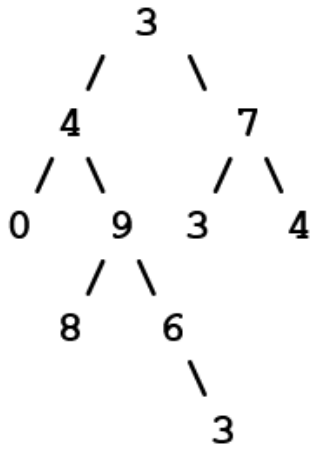
_____

Extra Credit the Second: 1 point: GDC opened in 2013. What was the name of the building it replaced?

_____

2. **Trees (17 points)** - Complete a method for a binary tree class that returns the depth of the deepest node that contains some target value. The binary tree in this question is NOT a binary search tree.

Consider this example with a **`BinTree`** that contains **`Integer`** objects. The root contains the value 3. Assume the variable **`t`** refers to the tree to the right.




```
t.deepestDepth(3) -> returns 4

t.deepestDepth(4) -> returns 2

t.deepestDepth(7) -> returns 1

t.deepestDepth(9) -> returns 2

t.deepestDepth(1) -> returns -1 (not present)

t.deepestDepth(-13) -> returns -1 (not present)
```

The **`BinTree`** and nested **`BNode`** classes:

```
public class BinTree<E> {
    private BNode<E> root; // stores null iff tree is empty

    public int deepestDepth(E tgt) // to be completed

    private static class BNode {
        private E data; // Never null.
        private BNode<E> left;  // null if left child doesn't exist.
        private BNode<E> right; // null if right child doesn't exist.
    }
}
```

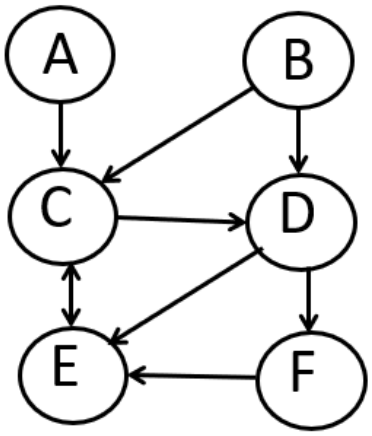
You may use the `Math.max` and `Object.equals` methods and the given `BinTree` and `BNode` classes.

Do not use any other Java classes or methods besides those.

Do not create any new objects, not even arrays of length 1.

```
/* pre: tgt != null
   post: return the depth of the deepest node that contains tgt
   or -1 if tgt is not present in this BinTree. This BinTree is
   not altered as a result of this method. */
public int deepestDepth(E tgt) {
```

3. **Graphs (16 points**) - Complete the **`numSource`** instance method for the **`Graph`** class from assignment 11. In a directed graph a vertex is a source vertex if it has an indegree of 0. In other words, there are no edges in the graph with the vertex as the destination of the edge.



Consider the very simple graph to the right. The vertices that contain A and B are source vertices.

The **`Graph`**, **`Vertex`**, and **`Edge`** classes for this question.

```
public class Graph {
     // The vertices in the graph.
     private Map<String, Vertex> vertices;

     private static class Vertex {
          private String name;
          private List<Edge> adjacent;

          private int scratch;
     }

     private static class Edge {
          private Vertex dest;
          private double cost;
     }
}
```

The **`Graph`** object is not altered other than changing the **`scratch`** variables of **`Vertex`** objects.
You **MAY NOT** use the **`clearAll`** method from the **`Graph`** class. You may use the following methods:

From the **`Map`** interface:
```
public Set<K> keySet()
public V get(K key)
public V put(K key, V val)
public V remove(K key)
```

From the **`Set`** interface:
```
public Iterator<E> iterator
public boolean add(E val)
```

From the **`List`** interface:
```
public Iterator<E> iterator()
public int size()
public E get(int pos)
```

You may use all methods from the **`Iterator`** interface either explicitly or implicitly.

Create and return a **`TreeSet<String>`** that contains the names of the **`Vertex`** objects in the **`Graph`** that are source vertices. If there are no source vertices in the Graph return an empty **`TreeSet<String>`**.

Do not use any other Java classes or methods other than those listed above.
Do NOT use recursion in your solution.

```
/* pre: none, post: per the problem description */
public TreeSet<String> getSourceVertices() {
```

4. **Encoding (17 points) -** Recall in the Huffman Programming assignment we used **`BitInputStream`** objects that could read a variable number of bits and return the integer equivalent of the bits read. This question involves using a **`BitInputStream`** to decode a file encoded with delta encoding. (not Huffman encoding)

Write a method that decodes a file that was encoded using ***Delta Encoding.*** Delta encoding is an alternative way of encoding sequences of values, typically integers, that may take up less space if the change from one number to the next is relatively small compared to the numbers themselves. Here is a very simple example:

Actual values: **`12 15 14 15 15 13 14 15 17 20 20 19`**

Delta encoding: **`12 3 -1 1 0 -2 1 1 2 3 0 -1`**

In delta encoding the first value in the sequence is the only value explicitly stated. Instead of listing each subsequent value the difference (or delta) from the previous value is listed. If the initial value is 12 and the second value is 15 the delta encoding stores 3 instead of 15. (12 + 3 = 15). The third value is 14. 14 is one less than the previous value 15 so the delta encoding stores -1, not 14.

The delta encoded file format for this question is:

```
<5 bits indicating bits per delta> <32 bits for first value>
<sign bit 1> <delta 1>  <sign bit 2> <delta 2>  ...  <sign bit n> <delta n>
```

The first 5 bits encodes an integer that represents how many bits each delta is. In the example above, the absolute values of the deltas range from 0 to 3. Therefore, we would only need **2 bits** to encode the deltas. Each of the deltas in the sequence would be encoded with 2 bits. (plus a sign bit)

The next 32 bits encodes an integer that represents the first value in the sequence.

Next come the deltas. Each delta consists of 2 parts.

The first is a single bit that represents the sign of the delta. We use the convention that if the sign bit is 0 the delta is a positive number and if the sign bit is 1 the delta is a negative number. (In CS429 you will learn the 2's complement approach to representing negative numbers in binary.)

The next x bits are the magnitude of the delta. The number of bits, x, in every delta equals the int represented by the initial 5 bits of the file.

The first 5 values in the sequence above (**`12 15 14 15 15`**) would be encoded as follows. (Spacing and line breaks added for clarity)

size of deltas (2 bits each) (initial value 12 as a 32 bit integer) (first delta +3)
```
00010 00000000 00000000 00000000 00001100 0 11
```

(second delta -1) (third delta +1) (fourth delta 0)
```
1 01 0 01 0 00
```

Use the **`readBits`** methods from the **`BitInputStream`** class.

```
public int readBits(int howManyBits) // from BitInputStream
```
Returns the number of bits requested as rightmost bits in returned value, returns -1 if not enough bits available to satisfy the request.

Create and return a single **`ArrayList<Integer>`** that stores the decoded sequence. You may use the **`add`** method for **`ArrayList`**. **Do not use any other classes or methods. Do not use recursion.**

```
/* pre: bis != null, bis is positioned at the start of a file properly
encoded with the specification described. There is at least 1 number in the
sequence. post: per the problem statement*/
public static ArrayList<Integer> decodeDeltaEncoding(BitInputStream bis) {
```