1 Quickies

Please explain what the output for each code segment will be (Compile Time Error, Run Time Error, Output) and *why*. Assume that each code segment is in a main function and the necessary library files are included if not otherwise noted.

1.1

```
String s1 = "Hello World";
String s2 = "Hello ";
s2 = s2 + "World";

Point a = new Point(3, 4);
Point b = new Point(3, 4);

if (s1 == s2)
  System.out.println("same strings");
if (a == b)
  System.out.println("same points");
if (a.equals(b))
  System.out.println("equal points");
```

Output: Compile Time Error, Run Time Error, or Output (please specify)

1.2

```
public class Test {

  int val;
  String name;

  public Test () {
    val = 10;
    name = "61b student";
  }

  public report () {
    System.out.println("Hi, my name is %s, I got a %f on the test", name, val);
  }

  public static void main (String[] args) {
    Test aa = new Test();

    report();
  }
}
```

Output: Compile Time Error, Run Time Error, or Output (please specify)

1.3

```
class NamedObj {
  String name;

  public NamedObj (String name) {
    this.name = name;
  }

  public static void main (String[] args) {
    Object p1 = new NamedObj("p1");
    Object p2 = p1;

    if (p1 == p2)
      System.out.println("Same Object");
  }
}
```

Output: Compile Time Error, Run Time Error, or Output (please specify)

1.4

```
class NamedObj {
  String name;

  public NamedObj (String name) {
    this.name = name;
  }

  public static void main (String[] args) {
    Object p1 = new NamedObj("p1");
    Object p2 = new NamedObj("p1");

    if (p1 == p2)
      System.out.println("Same Object");
  }
}
```

Output: Compile Time Error, Run Time Error, or Output (please specify)

1.5

```
int x1, x2;
x1 = 5;
x2 = 7;

x1 = x1 ^ x2;
x2 = x1 ^ x2;
x1 = x1 ^ x2;

System.out.println(x1 + x2);
System.out.println(x1 + " " + x2);
```

Output: Compile Time Error, Run Time Error, or Output (please specify)

2 Mathematical Run Time Analysis

Give True/False responses, you should understand *why*

2.1

$x^3 sin(x) \in \Omega(x)$

2.2

$3^x \in O(x^8)$

2.3

$3^x \in O(x!)$

2.4

$7x^2 + 45x + 9 \in O(x^2)$

2.5

$100n + 100x \in \Theta(n + x)$

3 Program Execution Run Time Analysis

Given a code segment or function, give the tightest bounds you can on asymptotic runtimes. Make sure to tell us what you are majoring on (length of list, number of elements, etc.). Also give a short description of how you arrived at the answer.

3.1

```
shuffle (ArrayList L) {
  Random R = new Random();
  for (int i = L.size; i > 0; i = i - 1)
    swap (L, i-1, R.nextInt(i));
}

swap (ArrayList L, int a, int b) {
  Object temp = L.get(a);
  L.set(a) = L.get(b);
  L.set(b) = L.get(a);
}
```

3.2

```
/* Linked List Documentation note:
All of the operations perform as could be expected for a doubly-linked list. Operations
that index into the list will traverse the list from the beginning or the end, whichever
is closer to the specified index.
*/

shuffle (LinkedList L) {
  Random R = new Random();
  for (int i = L.size; i > 0; i = i - 1)
    swap (L, i-1, R.nextInt(i));
}

swap (LinkedList L, int a, int b) {
  Object temp = L.get(a);
  L.set(a) = L.get(b);
  L.set(b) = L.get(a);
}
```

3.3

```
void mystery (LinkedList<String> L) {
  num = int(Math.log(double(L.size())) / Math.log(2.0)) + 1;
  LinkedList<String>[] storage = new LinkedList<String>[num];

  for(int i = 0; i < num; i++)
    storage[i] = new LinkedList<String> ();

  for (String S : L)
    addtoStorage(storage, S);
  L.clear();
}

static void addtoStorage (LinkedList<String> storage[], String s) {
  if (storage[0].size () == 0) {
    storage[0].add (s);
    return;
  } else if (storage[0].get(0).compareTo(s) < 0)
    storage[0].add (s);
  else
    storage[0].add (s, 0);

  int i;
  for (i = 1; storage[i].size () != 0; i += 1)
    mysteryHelper(storage[i], storage[i-1]);
  storage[i] = storage[i-1];
  storage[i-1] = new LinkedList<String> ();
}
```

4 Getting to the Finish

We provide you with the following declarations and methods that do *exactly* what the comments describe.

```
final static int UP = 0, DOWN = 1, LEFT = 2, RIGHT = 3;

public class MazeNode {
  //private fields
  <omitted>

  //methods

  public String getName() {
  //returns the name of this MazeNode
  }

  public void markNode () {
  // marks this node, subsequent calls to haveVisit() will always return true
  }

  public boolean haveVisit () {
  // returns true iff markNode() has ever been called on this node
  // false otherwise
  }

  public boolean canMove (int dir) {
  // returns true iff if there is a reachable neighbor in direction dir
  // dir must be UP, DOWN, LEFT, or RIGHT, else throws NoSuchDirectionException
  }

  public MazeNode move (int dir) {
  // returns the neighbor of this node in direction dir if such a reachable neighbor exists
  // else throws DeadEndException
  // dir must be UP, DOWN, LEFT, RIGHT, else throws NoSuchDirectionException
  }
}
```

4.1

You are tasked to write a function, that given a MazeNode as a starting point, you must find the least amount of steps (succesive *move(int dir)* calls) needed to get to a MazeNode with a name that includes the phrase "finish" (in any combination of upper/lowercase letters) in it.

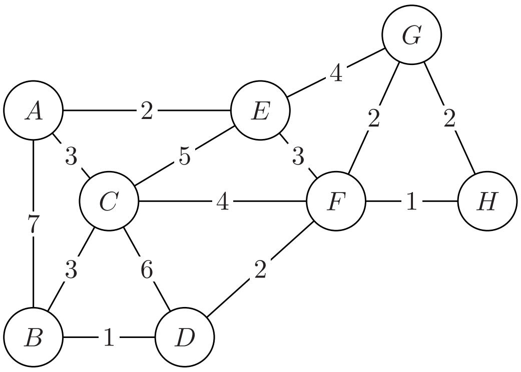
```
public int findFinish (MazeNode start) {
//this is most likely not enough space to finish the method




}
```

4.2

Please give an asymptotic run time analysis of your method provided that there are N mazeNodes that are in the connected graph extending from the start node.

5 Handturning some graph algorithms

Given a graph:



5.1

Please give a Depth First Traversal of the graph starting from node A. (Always visit in lexical order when given a choice)

5.2

Please give a Breadth First Traversal of the graph starting from node E. (Always visit in lexical order when given a choice)

5.3

Please give a list of the edges and node sets added to the MST using Kruskal's algorithm.
Please put an edge recently added, and all the nodes sets (each set is a set of nodes that are already connected) on each line for this list.

```
Ex.
  <edge added>, {first set of Nodes}, {second set of Nodes}, ...  - pass 1
  <edge added>, {first set of Nodes}, {second set of Nodes}, ...  - pass 2
  .
  .
  .
```

6 Family Trees

Given this type of tree structure:

```
public class FamilyTree {
  private FamilyTreeNode myRoot;

  <omitted methods and fields>
}

public class FamilyTreeNode {

  public FamilyTreeNode myParent;
  public ArrayList<FamilyTreeNode> myChildren;
  public String myName;

}
```

We want to implement the following method in FamilyTree class:

```
public string nameOfClosestCommonAncestor(String name1, String name2) {
  FamilyTreeNode f1 = myRoot.search(name1);
  FamilyTreeNode f2 = myRoot.search(name2);

  FamilyTreeNode ans = closestCommonAncestor (f1, f2);

  return ans.myName;
}
```

6.1

Please help implement the search function in FamilyTreeNode class:

```
public FamilyTreeNode search (String name) {
//looks through this node and all its children and descendants
//for a node with myName equal to name
//returns this node if found, null if no such node exists


}
```

What is the runtime of this function?

6.2

Please help implement the closestCommonAncestor function in FamilyTree class:

```
private static FamilyTreeNode closeCommonAncestor (FamilyTreeNode f1, FamilyTreeNode f2) {
//looks for the closest common ancestor of f1 and f2.
//The closest common ancestor of two nodes f1 and f2 is the closest node  the
//furthest from the root  thats an ancestor of both f1 and f2. An ancestor being
//itself, its parents, grandparents and so on.



}
```