1. The following code snippets are legal.
   (a) `ArrayList<String> a = new LinkedList<String>();`
       True / False
   (b) `List<Integer> l = new List<Integer>();`
       True / False
   (c) `Set<Double> s = new TreeSet<Double>();`
       True / False

2. We generally assume that code inside of the `finally` portion of a `try-catch` block will always be executed. Describe one situation in which the `finally` will not be executed.

3. What is the difference between a class and an object?

4. What does the `super` keyword do? When must you use it?

5. When do you have access to another object’s private variables and methods? What two methods are likely to take advantage of this?

6. Identify the errors in the following code:

```java
public interface TestInterface {
    public TestInterface();
    public void splat(String message);
}

public class AbstractClass {
    abstract protected void myMethod();
    public void callMyMethod() {
        myMethod();
    }
}
```
public class ConcreteClass extends AbstractClass implements TestInterface {
    protected int myMethod() {
        System.out.println("I got called!");
        return 7;
    }
    public void splat(String message) {
        System.out.println("SPLAT!");
        System.out.println(message);
    }
}

public class SimpleProgram {
    public static void main(String[] args) {
        AbstractClass myclass = new AbstractClass();
        myclass.callMyMethod();
        myclass.splat("woohoo");
    }
}

7. What are the two type of exceptions? What is an Error?

8. What will be printed by the following code?

public class ExceptionTest {
    public static void main(String[] args) {
        try {
            for (int i = 0; i < 5; i++) {
                process(i);
            }
        } catch (Exception e) {
            System.err.println(e.getMessage());
        } finally{
            System.out.println("Done processing");
        }
    }
}
public static void process(int num) throws Exception {
    if (num != 2) {
        System.out.println("Good number: " + num);
    } else {
        throw new Exception("Bad number: " + num);
    }
}

9. Instantiate a new BufferedReader which decorates a ForumReader. ForumReaders only use the default constructor but can throw a TooLongDidntReadException if the forum post is too long to be worth reading.

10. Why is buffering a good idea when doing file I/O?

11. Name and describe 3 Swing layout managers.
12. Describe what happens when you click on a JButton.

13. Write an ActionListener implementation called SimpleAction that simply prints “action!” when an ActionEvent is received.

14. Describe List, Set and Map. What interface do they all inherit?

15. Explain the difference between Comparator and Comparable.

16. Use an iterator to count the number of elements in the Collection victims.

   public int numVictims(Collection<Person> victims) {

   }
17. Write a Comparator that compares strings by their size modulo (%) 13 and use this comparator for sorting a TreeSet.
18. For each design pattern, briefly describe and how it works and why you would use it.

(a) Model-view-controller

(b) Observer

(c) Decorator

(d) Iterator

(e) Command

(f) Factory

(g) Template Method

(h) Composite