CS/IT115 Quiz       Spring 2014     Name:______________________________

You have the rest of our class (45 minutes) to complete this exam. You are allowed to use one
hand-written 8x11 inch sheet of notes. No calculators, phones, laptops or other electronic
devices are allowed.

1. Strings and Scanners (25 Points)

Given the following local variable declarations:

```
String futurama = “Bender Fry Leela Zoidberg”;
String simpsons = “Homer Marge Lisa Bart Maggie”;
Scanner scan = new Scanner(simpsons);
```

What is the value of the following expressions?

```
futurama.substring(0, 6); // a. ___Bender____

simpsons.charAt(6); // b. ____M______

scan.hasNextInt(); // c. ____false____

simpsons.compareTo(futurama); // d. ______6____

futurama.indexOf(“Fry”); // e. ____7____
```
import java.util.ArrayList;
import java.util.Collections;

public class ArrayListTests
{
    public static void main(String [] args)
    {
        ArrayList<String> list = new ArrayList<Integer>();
        // write code to add “Archer” and “Lana” to list

        list.add("Archer");
        list.add("Lana");

        // write a line of code to add “Pam” at beginning of list

        list.add(0, "Pam");

        // write a line of code to sort the list in ascending order

        Collection.sort(list);

        // write a loop to print out each element of the (now sorted) list, and then show what gets printed

        for (String s : list)
        {
            System.out.println(s);
        }

        // Show what gets printed
        Archer
        Lana
        Pam
3. The Fruit Interface and Classes (30 Points)

a. Write an interface called Fruit which contains a single method called getColor(). This method should take no parameters and return a String. Remember that the methods contained inside interfaces are abstract.

```java
public interface Fruit {
    public String getColor();
}
```

b. Write a class Apple that implements the Fruit interface. Remember that by implementing an interface, you’re promising to provide an implementation of the methods contained in that interface.

```java
public class Apple implements Fruit {
    public String color;
    public Apple(String c) {this.color = c;} //constructor (optional)
    public String getColor() { // required since we implement Fruit
        return this.color; // this could be hard-coded “Red” if you didn’t use constructor
    }
}
```

c. Similarly, write a class called Orange that also implements the Fruit interface.

```java
public class Orange implements Fruit {
    public String color;
    public Orange(String c) {this.color = c;} //constructor (optional)
    public String getColor() { // required since we implement Fruit
        return this.color; // this could be hard-coded “Orange” if you didn’t use constructor
    }
}
```
4. A Fruit Client (20 points)

Write a client class (which should contain a `main` method) in which you create an instance of an Apple and an instance of an Orange. Next, create a single ArrayList, to which you will add both of the above instances. Finally, loop over the ArrayList, printing out the color of each Fruit in the list.

// Just need a driver that uses the Fruit interface and Apple, Orange classes

```java
import java.util.ArrayList;
public class Driver {
    public static void main(String[] args) {
        Fruit a = new Apple("red"); // Declare interface as type, instantiate using the // implementing class
        Fruit o = new Orange("orange");
        // make an ArrayList of Fruits, so that we can add either Apples or Oranges (or // anything else that implements the Fruit interface
        ArrayList<Fruit> fruits = new ArrayList<Fruit>();
        fruits.add(a); // add an Apple to the list
        fruits.add(o); // add an Orange to the list
        for (Fruit f : fruits)
            System.out.println(f.getColor());
        // This loop works because we know that every type of object in the fruits list // has to provide an implementation of the getColor() method - that's what it // meansto implement the Fruit interface!
    }
}
```