Name:_____ (PRINT CLEARLY)

Lab Section:_____ Grader:_____

Quiz 3C – November 8

CS 2102 B19

1. (2 points) Examine the program below and write what will be displayed to the console (via the `System.out.println` command) when the program is run:

```
import java.util.LinkedList;

public class Main {
     public static void main(String[] args) {
          LinkedList<String> names = new LinkedList<String>();
          names.add("alpha");
          names.add("beta");
          names.add("delta");
          names.add("gamma");

          String str = "";
          for (String aName: names)
          {
               str = str + aName;
          }
          System.out.println(str);
     }
}
```

NOTE: When the "+" operator is used on strings, it functions as a concatenator.

Answer:
alphabetadeltagamma (2 points – do not deduct if student uses spaces)

2. (4 points) On the back side of this page is the `wordsLongerThan()` method from the `WordList` class that we created in lecture. Examine the code and the two JUnit tests that follow it. For each test, indicate whether the test will pass, fail, or result in an error that keeps your code from compiling. Write PASS, FAIL, or ERROR where appropriate. (You do not need to provide an explanation.)

```
//import statements omitted to save space
public class WordList {

LinkedList<String> words;
//Additional code omitted to save space

//Return a list of the words longer than the given length
LinkedList<String> wordsLongerThan(int low) {
     LinkedList<String> resultList = new LinkedList<String>();
     for(String word: this.words) {
          if(word.length() > low)
               resultList.add(word);
     }
     return resultList;
}
}

public class Examples {
WordList wList;

public Examples() {
     wList = new WordList();
     wList.words.addFirst("apples");
     wList.words.addFirst("or");
     wList.words.addFirst("pears");
}

@Test
public void testWordsLongerThan1() {
     LinkedList<String> expected;
     expected.addFirst("apples");
     expected.addFirst("pears");
     assertEquals(wList.wordsLongerThan(3), expected);
}
```
ERROR (2 points)

```
@Test
public void testWordsLongerThan2() {
     LinkedList<String> expected = new LinkedList<String>();
     assertEquals(wList.wordsLongerThan(7), expected);
}
```
PASS (2 points)

```
}
```

3. (4 points) Describe the problem in the following code, why it's a problem, and the change that needs to be made to make the code correct:

```
import java.util.LinkedList;

public interface IDog {
     public boolean canHuntPeople();
}

public class  Wolf implements IDog {
     boolean isAlive;
     double age;
     // constructor omitted to save space

     public boolean canHuntPeople() {
          return isAlive && age > 1;
     }
}

public class Corgi {
     String name;
     // constructor omitted to save space

     public boolean canHuntPeople() {
          return false;
     }
}

public class PuppyDogs {
     LinkedList<IDog> dogs;

     public PuppyDogs() {
          this.dogs = new LinkedList<IDog>();
          Corgi clark = new Corgi("Clark Gable");
          Wolf alpha = new Wolf(true, 5);
          this.dogs.add(clark);
          this.dogs.add(alpha);
     }
}
```

Answer:
The LinkedList holds any objects that implement the `IDog` interface, but the `Corgi` does not implement that interface, so we cannot add it to the list. To fix it, add `implements IDog` after `public class Corgi`.

2 points – Description of the problems (first sentence)
2 points – How we fix the problem (second sentence) – Also acceptable if the student just shows the fix by editing the code.