Quiz #3: Objects and Reading Classes INF-42, Winter 2011

Name _____

Lab _____

Recall the Academic Integrity statement that you signed. Write all answers clearly on these pages, ensuring your final answers are easily recognizable. The number of points for each problem is clearly marked, for a total of 25 points. I will post my solutions on the web on Monday, off the **Solutions** link, after class.

---

1. (6 pts) To the right of the line, draw a picture showing everything (variables/objects/etc.) after Java executes the following statements (do this on scratch paper first, and then redraw everything nicely here). Also, after each word **returns** (below the code) indicate what value each of the operators/methods returns.

```
String s1 = new String("abc");
String s2 = s1;
String s3 = new String("ABC");
String s4 = s2;
s1 = new String("abc");
```

`s2 == s4` returns

`s2 == s1` returns

`s2 == s3` returns

`s1.equals(s3)` returns

`s1.equals(s2)` returns

`s1.compareTo(s3)` returns

`s1.compareTo(s2)` returns

2. (6 pts) Complete the code fragment below. It prints the token in **`allwords`** that *directly precedes* the first occurrence of **`word`**: e.g., if **`allwords`** stores **`"ab lm pq xyz pq vl"`** and **`word`** stores **`"pq"`** it prints **`"lm"`** only; if **`word`** (a) appears first in **`allwords`**, print **`"*none*"`** (b) does not appear in **`allwords`**, print nothing. Use the **`StringTokenizer`** class in your code to help solve this problem. Hint: Declare/update variables **`current`** and **`previous`**, where appropriate, which remember values gotten from the **`StringTokenizer`**. Do the problem using Eclipse (see the helper file). First write code that just determines whether or not **`word`** is in **`allWords`**; then enhance it to meet all the specifications. See the examples of **`StringTokenizer`** in the notes.

```
String allwords = Prompt.forString("Enter list of all words separated by a space");
String word     = Prompt.forString("Enter one word");
```

3a. (5 pts) Examine the definition of the class **Quiz3** below. (a) To the right of each member, circle any abbreviation that describes it (in some cases none apply; in others more than one apply). The key is: as follows: **C** = Constructor; **M** = Method; **F** = Field; **O** = Overloaded. **At the end of each line that defines a method**, write either **a** or **m**, depending on whether you would classify that method most likely as an **accessor** or a **mutator**. You can infer all this information from the material in the lecture notes on reading classes.

```
public class Quiz3 {                                          C  M  F  O
  public  Quiz3         ()                    {...}          C  M  F  O
  public  Quiz3         (int l, int m, int r) {...}          C  M  F  O
  public  int  getMiddle()                    {...}          C  M  F  O
  private int  getRight ()                    {...}          C  M  F  O
  public  void incAll   (int delta)           {...}          C  M  F  O
  public  void incAll   (Quiz3 q)             {...}          C  M  F  O
  public  void switchAll()                    {...}          C  M  F  O
  public  void switchAll(Quiz3 q)             {...}          C  M  F  O
  public  int add       ()                    {...}          C  M  F  O
  public  static int add(int l, int m, int r) {...}          C  M  F  O
  public  int left;                                          C  M  F  O
  private int middle;                                        C  M  F  O
  private int right;                                         C  M  F  O
  public static int newCount = 0;                            C  M  F  O
}                                                            C  M  F  O
```

3b. (8 pts) Continue using the members of the **Quiz3** class defined above in this part of the question. Write a legal Java statement to meet each of the following requirements (or write **impossible**). Pay very close attention to the **access modifiers** (**public**, **private**, and **static**) appearing before each member of the class, because they will help differentiate correct from incorrect solutions: different access modifiers require different syntax. Assume, wherever useful, that the variables **x** and **y** have been declared and initialized as follows: `Quiz3 x = new Quiz3(), y = new Quiz3 (1,2,3);` Sometimes you will need to supply variables or literal arguments to these methods; use any, but ensure that they are of the correct type. Don't worry about what these constructors/methods/variables actually; just get the syntax correct.

(a) Print the value returned by any call to **getMiddle**:

(b) Print the value returned by calling **getRight**:

(c) Call **incAll** (using the "one object" form):

(d) Call **incAll** (using the "two objects" form):

(e) Call either form of **switchAll**:

(f) Print the value returned by calling the first form of **add**:

(g) Print the value returned by calling the second form of **add**:

(h) Print the value of any **left** instance variable without calling **Quiz3** methods:

(i) Print the value of **newCount** without calling **Quiz3** methods: