

Q1. [15 pts] A class C is defined with:

- an instance method **void** `inst()`
- a method **static int** `stat()`
- an instance variable `data` of type **int**

1	2	3	4	5	6	total

Inside the body of <code>inst</code> , it is impossible to have the statement " <code>inst();</code> "	(TRUE / FALSE)
Inside the body of <code>stat</code> , it is impossible to have the statement " <code>data = 20;</code> "	(TRUE / FALSE)
Inside the body of <code>stat</code> , it is impossible to have the statement " <code>inst();</code> "	(TRUE / FALSE)
Inside the body of <code>inst</code> , it is impossible to have the statement " <code>data = new Integer(17);</code> "	(TRUE / FALSE)

Q2. [20 pts] These questions focus on Object-oriented Concepts:

(a) Can a Java class have any number of derived classes? If yes, why is this useful?

(b) Can a derived class override an inherited method and redefine its return type? If yes, why is this useful?

(c) Can a constructor be declared to throw a checked exception? If yes, why is this useful?

(d) Can an interface declare a static method for use by its implementing classes? If yes, why is this useful?

Q3. [2 pts.] The Dilbert Zone. Extra points if I laugh so hard that milk comes out of my nose.



Q4. [15 pts.] Given an *ArrayList* object that contains *String* values, complete the implementation of the following method using the *Iterator* provided by the *ArrayList*.

```
/**
 * Return String "[value1,value2,value3, ..., valueN]" representing a comma-
 * separated String of the values found in an ArrayList. Note that the String
 * starts with "[", ends with "]", and there is no "," after the nth value.
 */
public static String format (ArrayList al) {
```

Q5. [25 pts] You are given the now familiar *NumberList* and *NumberNode* classes below:

```
/** List of numbers. */
public class NumberList {
    NumberNode head; /* first one. */

    /** Prepend NumberNode with i to list. */
    public void add (int i) { ... }
}
```

```
/** Node class representing an integer. */
public class NumberNode {
    int value; /* node value. */
    NumberNode next; /* next one. */

    public NumberNode (int i) {
        value = i;
        next = null;
    }
}
```

Write the appropriate **equals** method in *NumberList*, which determines if two *NumberList* entities are equivalent, namely, that the nodes within the two lists contain the same values in the exact same order.

Q6. [25 pts.] You have been hired as a consultant to prepare an initial object oriented design for a project that aims to create a personal weekly planner for scheduled meetings:

A personal weekly planner can store meetings during an ordinary five-day (Monday-Friday) business week. Some meetings occupy a single time slot (i.e., Tuesday at 11:00 AM) while a multi-day meeting represents a set of time slots throughout the week (i.e., Monday-Tuesday-Thursday-Friday at 11:00 AM). Each meeting has a room location. A time slot consists of a starting time and an ending time. It must be impossible to add to the weekly planner a meeting whose time slot(s) conflict with an existing meeting in the planner.

(a) [20 pts.] Show your object-oriented design below with boxes representing classes (both operations and data) and show the relationships between your classes

(b) [5 pts.] Without showing any code, explain how your planner prevents the addition of conflicting meetings