Question 2. (12 points) [The predictable "what does this print" question]

Consider the following class definitions.

```
public abstract class Foo {

   public abstract int f(int n);

   public void g(String s) {
      System.out.println("Foo g " + s);
   }
}
```

```
public class Bar extends Foo {

   public int f(int n) {
      System.out.println("n=" + n);
      return 2*n;
   }

   public void g(double x) {
      System.out.println("x=" + x);
   }

   public void h(boolean p) {
      if (p) {
         System.out.println("true");
      } else {
         System.out.println("not");
      }
   }

   public void h(double z) {
      System.out.println("z=" + z);
   }
}
```

```
public class Baz extends Bar {

   public int f(int k) {
      System.out.println("Baz f");
      return 2*super.f(k);
   }
}
```

Answer the questions on the next page. You can tear this page out if helps avoid having to flip the page back and forth.

Question 2. (cont) Assume that the following groups of statements, which use the classes defined on page 2, are typed into a newly reset DrJava interactions window. For each set of statements, if they cause a compile- or run-time error, explain what the error is, or if they execute successfully, explain what output is produced, if any. Be sure to show the output produced by earlier statements in the group, even if a later one causes an error.

(a)
```
Bar b = new Bar();

int x = b.f(3);

System.out.println(x);
```

(b)
```
Foo f = new Bar();

f.g(10.0);
```

(c)
```
Baz z = new Bar();

z.h(true);

z.h(17);
```

(d)
```
Bar a = new Baz();

System.out.println(a.f(2));

a.h(true);
```

The buggy code page.

Question 3. (2 points) One of your colleagues is trying to create a new class that represents points in a plane. Here's the code so far:

```
public class Point {
   private double x, y;       // coordinates

   public double getx() { return x; }
   public double gety() { return y; }

   public String toString() { return "x,y = " + x + "," + y; }

   // create a point and print their coordinates
   public static void main(String[] args) {
      // create point and set coordinates
      Point p = new Point();
      x = 17;
      y = 42;
      // print coordinates
      System.out.println(p);
   }
}
```

Style issues aside, this code won't even compile. What is the problem?

Question 4. (2 points) Another colleague is learning Swing graphics and has created the following code to draw a couple of nested circles inside a `JPanel` that will later be added to a `JFrame` window. But it doesn't work. What's the problem?

```
import java.awt.*;
import javax.swing.*;

public class SimpleWindow extends JPanel {

  /** Paint a red circle when called */
  public void paintComponent(Graphics g) {
    g = new Graphics();
    g.setColor(Color.red);
    g.fillOval(100,100,200,200);
    g.setColor(Color.blue);
    g.fillOval(150,150,100,100);
  }
}
```

Question 5. (3 points) We've claimed that the model-view-controller design is much better than, for instance, having all of the code intermingled in the same class(es). Give two reasons why model-view-controller is the better design.

Question 6. (2 points) Typically when we use container classes we put an import statements at the beginning of the file.

```
import java.util.*;
public class Example {
   public static void main(String[] args) {
      ArrayList a = new ArrayList();
      …
   }
}
```

Recently, the boss read something on the web that claimed that `import` statements are evil. You have been ordered to remove all of the `import` statements from your code. Is this possible? Can you rewrite the code to remove every `import` and still have the program work as before? If so, how?