Question 1: (25 points)

Part (a): (7 points)
Write in the box on the right the output that will be produced by executing the following program.

```java
public class Q1a {

    public static void main(String[] args) {
        int n=3, p=6;
        int w = junk(p);
        System.out.println("w is " + w);
        System.out.println("n is " + n);
        System.out.println("p is " + p);
    }

    public static int junk(int n) {
        int p=1;
        n= n + p;
        System.out.println("n is " + n);
        System.out.println("p is " + p);
        return p;
    }
}
```

Output

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>is</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>p</td>
<td>is</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>is</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>is</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>p</td>
<td>is</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

Part (b): (18 points)  Consider class `Counter` below.

```java
public class Counter {
    private int tally;
    public int getTally() { return tally; }
    public void stepCount() { tally = tally + 1; }
    public static void showName() { System.out.println("Class Counter"); }
    public void funTally1(int t) { tally = t; }
    public void funTally2(int t) { this.tally = t; }
    public void funTally3(int tally) { this.tally = tally; }
    public void funTally4(int tally) { tally = tally; }
}
```

For each sentence below, indicate whether it is correct by writing “true” or “false” on the blank:

- **false** Variable `tally` is a class variable.
- **true** Variable `tally` is an instance variable.
- **true** Variable `tally` is a field.
- **false** Method `getTally()` is a procedure.
- **true** Without changing the method header, `stepCount()` may be changed to contain a `return` statement.
- **true** Method `stepCount()` may be called from an instance of class `Counter`.
- **true** Method `showName()` may be called from an instance of class `Counter`.
- **true** Methods `funTally1` and `funTally2` have the same functionality.
- **false** Methods `funTally1` and `funTally3` have the same functionality.
- **false** Methods `funTally3` and `funTally4` have the same functionality.

Write a call to method `showName()`. (E.g., call `showName()` in DrJava’s interaction pane.)

`Counter.showName()`
Question 2 (25 points)

A textile company mixes dyes to formulate special colors. Complete the method below to determine and print the color that results from mixing black and yellow dyes and from adding a metal oxide. The company’s super secret formula is as follows:

- Using more yellow dye than black dye yields “banana brown”
- Using the same amounts of black and yellow dyes or using more black than yellow yields “gooey grey,” but if over 80% of the mix is black dye, then the color becomes “bean black.”
- Adding a metal oxide to the dye mix will add a metallic sheen, resulting in “metallic banana brown,” “metallic gooey grey,” or “metallic bean black.”

Hint: Remember that you can concatenate Strings using the + operator.

```java
/** Mix dyes and metal oxide to form special colors as specified above.
 * b is fraction of black dye (e.g., 80% black dye means b is 0.8)
 * y is fraction of yellow dye
 * addOxide has the value true if metal oxide is added to the dye mix
 */
public static void makeColor(double b, double y, boolean addOxide) {
    String color;  //the color created by mixing the dyes and metal oxide

    if ( y > b )
        color = "banana brown";
    else  // {b>=y}
        if ( b > 0.8 )
            color = "bean black";
        else
            color = "gooey grey";

    if (addOxide)
        color = "metallic " + color;

    System.out.println("The final color is " + color);
}
```
Question 3: (20 points)

Write a class `PyramidFrame` that customizes `JFrame` to have one procedure, `makePyramid()`. The task of method `makePyramid()` in an instance of this class is to create and show one other `JFrame` centered above this one (the original frame), see diagram. The top `JFrame` is half the width of the original frame and has the same height as the original frame. Below are the specifications of some useful instance methods from class `JFrame`:

- `show()`: Show the frame
- `getHeight()`: (int) the height of the window in pixels
- `getWidth()`: (int) the width of the window in pixels
- `setSize(w,h)`: Set the width and height of the window to `w` and `h`
- `getX()`: (int) x-coordinate of the top left corner of the window
- `getY()`: (int) y-coordinate of the top left corner of the window
- `setLocation(u,v)`: Set the x- and y-coordinates of the top left corner of the window to `u` and `v`

```java
import javax.swing.*;

public class PyramidFrame extends JFrame {

    private JFrame top;  //top frame of the pyramid

    /** Create and show two frames stacked as a pyramid */
    public void makePyramid() {

        show();  //show the bottom frame
        //OK if this statement is not used

        top = new JFrame();

        top.setSize(getWidth()/2, getHeight());

        top.setLocation(getX() + getWidth()/4,
                        getY() - getHeight());

        top.show();
    }

}
```
Question 4: (30 points)

A positive, even number \( n \) is divisible by 2. For example,

- 8 is divisible by 2 three times (8/2 gives 4; 4/2 gives 2; 2/2 gives 1; 1 is not divisible by 2)
- 2 is divisible by 2 once
- 10 is divisible by 2 once (10/2 gives 5; 5 is not divisible by 2)

Given a positive integer value in variable \( n \) (type \textbf{int}), write a program fragment to determine the number of times that \( n \) is divisible by 2 and store this number in a variable \( d2 \) (type \textbf{int}). If variable \( n \) stores an odd number, set \( d2 \) to zero and display the message “\( n \) is not divisible by 2.”

Do not use any pre-defined methods other than \textbf{System.out.println}.

```java
//Write your code fragment below assuming that n has been declared and initialized. //n>0

    int d2=0;  //no. of times n gets divided by 2 so far
            //OK if student assumes d2 has been declared

    if (n%2==0)  // {n is even}
        for (; n%2==0; n/=2)  //enter loop if n is still even
            // (loop stops when n becomes odd)
            d2++;

    else
        System.out.println(“n is not divisible by 2”);
```

//