

## CS 102: Practice Test Problems for Test 2

1. What is the output to the screen?

```
public class Question1{
    public static void main (String[] args){
        System.out.println(f(4));
    }
    public static int f (int x){
        if (x==1 || x==0)
            return x;
        else
            return f(x-1) + f(x-2);
    }
}
```

**Solution:**

3

2. (a) Write a class `PersonalData` which has three `String` data fields representing the first name, last name, and social-security number. Make the `PersonalData` class implement the `Comparable` interface, based on comparing social-security numbers. Also write a constructor for the class to set the data.  
(b) Write a class `SSNumberFormatException` which extends the `Exception` class and sets the exception message to be "Invalid Social Security Number Format". Modify your `PersonalData` class so that the constructor throws a `SSNumberFormatException` if the social security number is not of the format `xxx-xx-xxxx`, where each `x` represents a digit from '0' to '9'.

**Solution:**

```
public class SSNumberFormatException extends Exception{
    public SSNumberFormatException(){
        super("Invalid Social Security Number Format");
    }
}
```

```

}

public class PersonalData implements Comparable{
    String fName, lName, ssNum;
    public PersonalData (String f, String l, String s)
        throws SSNumberFormatException{
        fName = f;
        lName = l;
        ssNum = s;
        boolean valid = true;
        if (s.length() != 11)
            valid = false;
        else if (s.charAt(3) != '-' || s.charAt(6) != '-')
            valid = false;
        else
            for (int i=0; i<11; i++)
                if (i!=3 && i!=6 &&
                    (s.charAt(i)<'0' || s.charAt(i)>'9'))
                    valid = false;
        if (!valid)
            throw new SSNumberFormatException();
    }
    public int compareTo(Object other){
        return ssNum.compareTo( ((PersonalData)other).ssNum );
    }
    public String toString(){
        return ssNum + " " + lName + ", " + fName;
    }
}

```

3. Consider the following array of Strings:

```
"happy" "home" "area" "asp" "xylophone" "ire" "ape"
```

(a) Trace through the selection sort algorithm acting on the above String array.

**Solution:**

```
"happy" "home" "area" "asp" "xylophone" "ire" "ape"
```

```
"ape" "home" "area" "asp" "xylophone" "ire" "happy"
"ape" "area" "home" "asp" "xylophone" "ire" "happy"
"ape" "area" "asp" "home" "xylophone" "ire" "happy"
"ape" "area" "asp" "happy" "xylophone" "ire" "home"
"ape" "area" "asp" "happy" "home" "ire" "xylophone"
"ape" "area" "asp" "happy" "home" "ire" "xylophone"
```

- (b) Trace through the insertion sort algorithm acting on the above String array.

**Solution:**

```
"happy" "home" "area" "asp" "xylophone" "ire" "ape"
"happy" "home" "area" "asp" "xylophone" "ire" "ape"
"area" "happy" "home" "asp" "xylophone" "ire" "ape"
"area" "asp" "happy" "home" "xylophone" "ire" "ape"
"area" "asp" "happy" "home" "xylophone" "ire" "ape"
"area" "asp" "happy" "home" "ire" "xylophone" "ape"
"ape" "area" "asp" "happy" "home" "ire" "xylophone"
```

4. What is the output to the screen?

```
public class Question3{
    public static void main (String[] args){
        String[] a = new String[3];
        a[0] = new String("House");
        a[2] = new String("Home");
        for (int i=0; i<3; i++){
            try{
                System.out.println(i/(i-2));
                System.out.println(a[i].length());
                System.out.println("Pass " + i);
            }
            catch (NullPointerException e){
                System.out.println("Properly initialize.");
            }
            catch (ArithmeticException e){
                System.out.println("Don't divide by zero.");
            }
        }
    }
}
```

```
    }  
}
```

**Solution:**

```
0  
5  
Pass  
0  
-1  
Properly initialize.  
Don't divide by zero.
```

5. A class has an instance variable

```
int[] a;
```

Write a class method

```
void delete(int n)
```

which moves all elements of **a** with index  $(n + 1)$  or above one place to the left, and puts a 0 in the last element of **a**. (So if **a** contains the array elements 5 7 13 2 6 1, then after a method call `delete(2)`, **a** should contain 5 7 2 6 1 0.)

**Solution:**

```
public void delete(int n){  
    for (int i=n; i<a.length-1; i++)  
        a[i] = a[i+1];  
    a[a.length-1] = 0;  
}
```

6. A function **f** is defined for positive integers by

$$f(n) = \begin{cases} n^3 + f(n-1) & n > 0 \\ 0 & n = 0 \end{cases}$$

Implement **f** as a **static** method.

**Solution:**

```

public static int f(int n){
    if (n==0)
        return 0;
    else
        return n*n*n + f(n-1);
}

```

7. Write a Java program which reads integers from a file `ints.dat` into an array (you assume the file contains at most 1000 integers). Then use the `Sorting.selectionSort` method to sort the array. Print all the elements of the sorted array to the screen, separated by spaces. (You do not have to write the selection sort algorithm; just make an appropriate method call.)

**Solution:**

```

import java.util.Scanner;
import java.io.*;
public class readSortInts{
    public static void main (String[] args) throws IOException{
        Scanner fileScan = new Scanner(new File("ints.dat"));
        int count = 0;
        int[] readArray = new int[1000];
        int[] storeArray;
        while (fileScan.hasNext()){
            readArray[count] = fileScan.nextInt();
            count++;
        }
        storeArray = new int[count];
        for (int i=0; i<count; i++)
            storeArray[i] = readArray[i];
        Sorting.selectionSort(storeArray);
        for (int i=0; i<count; i++)
            System.out.print(storeArray[i] + " ");
        System.out.println();
    }
}

```

8. Write a method

```
public void paintComponent(Graphics page)
```

which paints a bullseye picture consisting of 5 concentric black and white rings of width 25 pixels. The outermost circle should have radius 150 and center (150,150). Inside the innermost ring, draw a red disk (the bullseye) of radius 25. The picture is available at

<http://www.pegasus.rutgers.edu/~loftin/cs101/bullseye/bullseye.html>

**Solution:**

```
public void paintComponent (Graphics page){
    super.paintComponent (page);
    Color[] colors = {Color.white,Color.black,Color.white,
                      Color.black,Color.black,Color.red};
    int x = 0, y = 0, diameter = 300;
    page.setColor (Color.white);
    for (int count = 0; count < 6; count++){
        page.setColor (colors[count]);
        page.fillOval (x, y, diameter, diameter);
        diameter -= 2*25;
        x += 25;
        y += 25;
    }
}
```

9. Describe what happens when the up and down buttons are pushed in the following applet. In particular, what is the text in the label after the up button is pushed four times and then the down button is pushed once?

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class App extends JApplet{
    int num;
    private JLabel label;
    private JButton upButton, downButton;
```

```

public void init(){
    Container content = getContentPane();
    JPanel panel = new JPanel();
    content.add(panel);
    num = 0;
    label = new JLabel("");
    upButton = new JButton("Up");
    downButton = new JButton("Down");
    ButtonListener listener = new ButtonListener();
    upButton.addActionListener(listener);
    downButton.addActionListener(listener);
    panel.add(label);
    panel.add(upButton);
    panel.add(downButton);
}
private class ButtonListener implements ActionListener{
    public void actionPerformed(ActionEvent event){
        if (event.getSource() == upButton)
            num++;
        else if (event.getSource() == downButton)
            num--;
        String stars = "";
        for (int i=0; i<num; i++)
            stars += "*";
        label.setText(stars);
    }
}
}

```

**Solution:** The text in the label consists of a number of \*s. The number of \*s is given by the number of time the Up button has been pushed minus the number of times the Down button has been pushed (the number of \*s is stored in the variable num). In particular, if the Up button has been pushed 4 times and the Down button once, the label text is \*\*\*.