Lab 2  

Software Concepts

Name: ____________________________

Prerequisites

Please complete the following prerequisite activities before beginning this lab:
- Read Chapter 2 of Java Software Solutions
- Review the various types of errors that can occur in a program
- Compare and contrast Java applications and applets
- Review the basic drawing methods
- Skim the contents of Appendix K of Java Software Solutions

Introduction

After reading Chapter 2, you have seen some fairly simple Java programs. Before we move on to additional programming constructs, it is important to get familiar with the process of getting a program designed, entered, compiled, and executed. The better you are at these fundamental activities, the easier it will be to move to more complex programming techniques.

Approach this lab with two primary goals in mind: to become familiar with the Java constructs discussed, and to make the process of program compilation and execution second nature. Do not concern yourself with the reserved words that are used but not discussed, such as static and void. They will be covered at the appropriate point in the future.

Lab Objectives

After completing this lab, you should understand and be able to describe the following:
- The basic structure of a Java application
- The print and println methods
- The repetitive edit-compile-execute process
- Program error types
- The basic structure of a Java applet
- The drawString, drawLine, and drawOval methods
- The process of executing an applet
Activities

Perform the following activities, in order, recording any information requested in the space provided. Your instructor will provide any specific information necessary to complete this lab.

Compiling a Program

1. Enter the following program, saving it in a file called Useless.java, or copy it from the location specified by your instructor:

   ```java
   // A source of useless information.
   class Useless {
       // Prints some simple messages.
       public static void main (String[] args) {
           System.out.println("Some useless facts:");
           System.out.println("Your ears never stop growing.");
           System.out.println("Catsup travels at 25 miles per year.");
       } // method main
   } // class Useless
   ```

2. Compile and execute the program. What is the output?

3. Add comments to the top of the program file specifying your name and other pertinent information.

4. Add a useless fact of your own to the bottom of the list. Compile and execute the program to see your change. What was the line you added?

5. Modify the program so that all of the useless facts are indented three spaces. Compile and execute the program to verify your change.

6. Modify the program so that a blank line is printed between the first and second lines of output. Compile and execute the program. What was your modification?
7 Replace the first call to `System.out.println` in the program with `System.out.print`. Compile and execute the program. How did the output change? Why?

8 Add one new line to the program that reverses the effects of the changes you made in Activity 7. Compile and execute the program. What was the new line?

9 Remove the semicolon from the end of the last `println` statement. Compile the program. What was the error message given? What is the line number of the error as reported by the compiler? What kind of programming error is this?

10 Correct the error by replacing the semicolon. Compile and execute the program to verify the correction.

11 Remove the word `public` from the `main` method declaration. Compile the program. Were compilation errors produced? Execute the program. Did it execute normally?

12 Correct the error by replacing the word `public`. Compile and execute the program to verify the correction.

13 Change the third `println` statement to:

   `System.out.println ("Catsup travels at \" + 25 + \" miles per year.\")`;

Compile and execute the program. Did the output change? Explain how the modified line works. What is the purpose of the `+` operators?
Replace the literal 25 in the modified line with the division expression 100/4. Compile and execute the program. Did the output change? Explain.

Replace the division expression with 100/0. Compile and execute the program. Record the output below. What happened? What kind of error is this?

Correct the error by replacing the division expression with 100/4. Compile and execute the program to verify the correction.

Print the final version of Useless.java and turn it in with the lab.

**Developing an Application**

You will now design and implement a program that prints your initials in large letters.

Create the basic structure of the program, including the class and main method headers. Follow the model of the Useless program. Call the class in the new program Initials. Save the program in a file called Initials.java.

Add comments to the top of the program file specifying your name and other pertinent information.

Design the format for your initials, similar to the example below. Make each letter five lines high and 7 characters wide. Use the pound symbol (#) to make the letter. Sketch your initials below.

```
#######
#    #
#    #
#    #
####
```

Add five calls to println to output your initials (so that they read left to right) in the format you designed. Compile and execute the program. Modify it as necessary until the initials are printed properly.

Add println statements above and below your initials, printing various information about you, such as your name, age, and major. Compile and execute the program.

4 Lab 2 Software Concepts
Add comments to the program, above the class header and above the main method header, describing the processing. Compile and execute the program to ensure that these modifications did not affect the program.

Print the final version of Initials.java and turn it in with the lab.

Working with Applets

Enter the following applet, saving it in a file called Sketch.java, or copy it from the location specified by your instructor:

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

// Testing various drawing routines.
public class Sketch extends JApplet {
   // Draws some simple figures.
   public void paint(Graphics page) {
      page.drawString("Testing some shapes", 30, 30);
      page.drawLine (30, 60, 50, 60);
      page.drawLine (30, 70, 60, 70);
      page.drawOval (90, 50, 20, 20);
      page.drawOval (120, 50, 40, 20);
   } // method paint
}
// class Sketch
```

Add comments to the top of the program file specifying your name and other pertinent information.

Compare the structure of this applet to the structure of the Useless application from Activity 1. What are the primary similarities and differences?
28 Compile the applet. Fix any problems that exist until you get a clean compile.

29 Enter the following HTML code, saving it in a file called Sketch.html, or copy it from the location specified by your instructor:

```html
<! Sketch.html>
<HTML>
 <HEAD>
   <TITLE>The Sketch Applet</TITLE>
 </HEAD>
 <BODY>
   <H3>The Sketch Applet</H3>
   <APPLET CODE="Sketch.class" WIDTH=300 HEIGHT=150>
   </APPLET>
   <HR>
 </BODY>
</HTML>
```

30 The HTML code refers to the file Sketch.class. What does that file contain? Where does it come from?

31 Execute the program by submitting the HTML file to appletviewer, or by using a web browser. What is the output?

32 Explain the parameters to the first call to drawLine.
33 Modify the program such that the left end point of the first line is the same as the left end point of the second line. Compile and execute the program to test the change. What was the modification?

34 Modify the program so that the first line is perpendicular to the second, forming a plus sign. Compile and execute the program. What was the modification?

35 Explain the parameters to the first call to `drawOval`.

36 Enlarge and reposition the second oval so that it completely contains the first. Compile and execute the program. Write down the modified call to `drawOval`.

37 Change the second oval so that it is the same size as the first, and reposition it so that their edges touch, forming an infinity sign (sideways figure eight). Compile and test your modification.

38 Print the final version of `Sketch.java` and turn it in with the lab.

**Developing an Applet**

You will now design and implement an applet that draws a snowman figure similar to the following:

![Snowman Diagram]
Create the basic structure of the applet, following the model of the Sketch program. Call the class in the new program Snowman. Save the program in a file called Snowman.java.

Add comments to the top of the program file specifying your name and other pertinent information.

Plan the positioning and shape of the three main ovals of the snowman. Sketch out the ovals below and indicate the key coordinates.

Add three calls to drawOval to draw the main body of the snowman. Compile and execute the program. Modify the program as needed until the drawing is appropriate.

Given the final coordinates of the snowman's body, plan the position and shape of the arms and eyes. Sketch them below, with coordinates.

Modify the program to draw the arms and eyes. Compile and execute the program. Modify as needed until it is correct.

Plan the position of the hat, drawn with four calls to drawLine. Modify the program to draw the hat. Compile and execute the program, modifying as needed until it looks appropriate.

Use drawString to write "A Virtual Snowman" centered under the snowman figure. Compile and execute the program, modifying as needed.

Add comments to the program, above the class header and above the main method header, describing the processing.

Add comments that identify the lines of code that draw the body, arms, eyes, and hat of the snowman. Compile and execute the program to ensure that these modifications did not affect the program.

Print the final version of the applet and turn it in with the lab.