Objectives:
Practice using switch statements; do and for loops; and explore the use of dialog boxes (JOptionPane) and the use of System.currentTimeMillis() (current system time in milliseconds) in experimental settings.

Preparation: Practice using the conditional operator and the switch statement

1. Conditional Operator:
Revisit your code for the Snowman applet (or the FancySnowman) from Lab 5. (http://www.csc.villanova.edu/~map/1051/Chap02/Snowman.java).

a) Modify the code to add a boolean constant at the beginning of the code (along with the other constants, MID and TOP):

```java
final boolean HAPPY = true;
```
Run the applet and verify that this change has no effect on what is displayed.

b) Now change the code that draws the smile or a frown, depending on this constant:

```java
if (HAPPY)
    page.drawArc (MID-10, TOP+20, 20, 10, 180, 180);   // smile
else
    page.drawArc (MID-10, TOP+20, 20, 10, 0, 180);   // frown
```
Rerun to verify that this change again has practically no effect on what is displayed.

c) Change the value of HAPPINESS to false and run again. The Snowman should now look like he is frowning.

d) Rewrite the code from (b) to eliminate the if/else and use the conditional operator instead. Write your code below and try it in your program, both with HAPPINESS = true and with HAPPINESS = false. [Hint: it should now be a single line of code!]

---

e) Let’s try the conditional operator in another old program (the change making program from Project 2). For example, if you have some code like this:

```java
System.out.println ("Your change is " + num + " Dimes");
```
How would you replace it with code that prints “Dime” or “Dimes “depending on whether num is equal to 1?

---

2. Switch statement
Write a program to input a number n that symbolizes a version of the Mac OS X 10.n software. Output the name of that version of the Mac OS X 10.n software. For example, if the user inputs 8, then the program should output “Mountain Lion.”
[You will need to do a little googling to find the names of the different mac software. They run from 1-10 and most of them are cats, but more recently they are joined by mountains.]

When finished, upload and submit through Blackboard all your program files (parts 1c, 1e, and 2) and a scan of this page.

Remember to Bring This Worksheet To Class To Check With Your Partner
Part A. Practice using dialog boxes and do loops
Run EvenOdd.java and get familiar with how it works. Modify it so that it performs the function of the GPA calculator (from Lab 4b – version that asks each time whether to repeat). You will need to modify the prompts and other interaction (for example, "Enter quality points: " instead of "Enter an integer: " ) and to input an additional number for the credits.

Part B. Implement a simple experimental setup that uses timings
Download and run ResponseTimeExperiment.java. (shown on next page)

• Run the program a few times to observe its function.
• Note the use of System.currentTimeMillis() to find the current time at two different points during the run, and thus obtain the amount of time elapsed between the issuing of a question and the input of the answer from the user.
• Refer to the code shown on next page. Modify the code enclosed in the boxes so as to:
  o Use a single line of code to compute the value of outcome and,
  o Display the result ("Correct!" or “Incorrect.”) using a single System.out.println statement.
• Test your program to ensure that it functions exactly as before.

Part C. Repeat the experiment using a for-loop
Modify ResponseTimeExperiment to repeat the experiment 4 times, using a for-loop. Compute the number of correct answers and the average response time of the user and print these results at the end. See the algorithm below – note the new parts in boldface.

Algorithm:
input name
print personalized welcome message & instructions
numCorrect = 0
totalTime = 0
repeat 4 times:
  a = random int
  b = random int
  startTime = current time
  print question using a, b
  input answer
  endTime = current time
  outcome = 1 or 0 (result of evaluating answer)
  reactionTime = endTime – startTime
  print outcome as “Correct” or “Incorrect”
  numCorrect = numCorrect + outcome
  totalTime = totalTime + reactionTime
  averageTime = totalTime/4
print averageTime
print goodbye
// ResponseTimeExperiment.java Measure response time for addition problems.
// M A Papalaskari

import java.util.Scanner;
import java.util.Random;

public class ResponseTimeExperiment
{
    public static void main(String[] args)
    {
        Scanner in = new Scanner(System.in);
        Random rand = new Random();

        System.out.print("Please enter your name: ");
        String name = in.nextLine();

        System.out.println("Hello " + name
        + ". Please answer as fast as you can."
        + "\n\nHit <ENTER> when ready for the question.");
in.nextLine();

        int a = rand.nextInt(100);
        int b = rand.nextInt(100);

        long startTime = System.currentTimeMillis();

        System.out.print(a + " + " + b + " = ");
        String response = in.nextLine();
        int number = Integer.parseInt(response);

        long endTime = System.currentTimeMillis();

        if (number == a + b)
            outcome = 1;
        else
            outcome = 0;

        long reactionTime = endTime - startTime;

        System.out.println("Correct!");
        else
            System.out.println("Incorrect.");

        System.out.println("That took " + reactionTime + " milliseconds");
        System.out.println("Thank you " + name + ", goodbye.");
    }
}
Lab 9 Comments

Comments on this lab, please:

What was the most valuable thing you learned in this lab?

What did you like best about this lab?

Was there any particular problem?

Do you have any suggestions for improving this lab as an effective learning experience?