Objectives:

Practice using the conditional operator; switch statements; do and for loops. Learn how to implement a simple experimental setup to measure response times using System.currentTimeMillis() (current system time in milliseconds).

Preparation: Response time experiment – accessing the system clock

1. Download and run ResponseTimeExperiment.java (shown on next page).

2. 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 issuing of a question and the input (answer) from the user.

3. Refer to the code shown on the next page. Modify the code enclosed in the boxes, as follows:
   - Using the conditional operator, compute the value of outcome in just one line of code (replace the if/else).
   - Display the result ("Correct!" or "Incorrect.") using a single System.out.println statement (again, use the conditional operator).

4. Test your program to ensure that it functions exactly as before.

5. Upload and submit ResponseTimeExperiment.java through blackboard under “Lab 6 Prep”.
// 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(); // wait for user to hit <ENTER>

        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();

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

        long reactionTime = endTime - startTime;

        if (outcome == 1)
            System.out.println("Correct!");
        else
            System.out.println("Incorrect.");

        System.out.println("That took " + reactionTime
            + " milliseconds");
        System.out.println("Thank you " + name + ", goodbye.");
    }
}
Part A: 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.]

Part B. do loops

Modify your code for Lab 4b to use a do loop (that is the version of the GPA program that asks each time whether to repeat).

Part C. for-loops

Modify your code for Lab 4d to use a for-loop (that is the version of the GPA program with the exact count).

Part D. Annotate the code of ResponseTimeExperiment.java

1. Examine the code of ResponseTimeExperiment.java printed on in Page 2. Annotate the code with the number to match each step in the algorithm (some of the numbers may correspond to more than one line of code).

Algorithm:

1. input name
2. print personalized welcome message & instructions
3. \( a = \) random int
4. \( b = \) random int
5. \( \text{startTime} = \) current time
6. print question using \( a, b \)
7. input answer
8. \( \text{endTime} = \) current time
9. outcome = 1 or 0 (answer is correct or incorrect)
10. \( \text{reactionTime} = \) endTime – startTime
11. print outcome as “Correct” or “Incorrect”
12. print goodbye
2. Now consider an algorithm that repeats the experiment \textit{four times} using a \textbf{for}-loop, while also keeping track of the number of correct answers and computing the average response time of the user – see the algorithm below. Some steps have been added (highlighted).

- Annotate the code for \texttt{ResponseTimeExperiment.java} in Page 2 again to add these new steps, as follows:
  - Draw a box around the code that needs to go inside the loop
  - Draw an arrow with a label in the appropriate positions for where A, B,...G need to be inserted (show an arrow and the label of the step where it should go in the code, eg, A $\rightarrow$ should go ) to prepare to implement the algorithm. Be sure to \textbf{mark clearly, with the appropriate letter}, the exact position where code must be added.

\textbf{Algorithm:}

1. input name
2. print personalized welcome message & instructions

\begin{itemize}
  \item \textbf{A. numCorrect} = 0
  \item \textbf{B. totalTime} = 0
  \item \textbf{C. repeat 4 times:}
    \begin{itemize}
      \item a = random int
      \item b = random int
      \item startTime = current time
      \item print question using a, b
      \item input answer
      \item endTime = current time
      \item \textbf{D. totalTime} = totalTime + reactionTime
      \item outcome = 1 or 0 (answer is correct or incorrect)
      \item reactionTime = endTime – startTime
      \item print outcome as “Correct” or “Incorrect”
      \item \textbf{E. numCorrect} = numCorrect + outcome
    \end{itemize}
  \item \textbf{F. averageTime} = totalTime/4
  \item \textbf{G. print averageTime}
  \item \textbf{H. print numCorrect}
  \item 12. print goodbye
\end{itemize}

3. Add some code and restructure \texttt{ResponseTimeExperiment.java} to implement the new steps, using your notes from 2, above. Be sure to use a \textbf{for-loop} to implement the repetition of the experiment.
Lab 6 Comments  Name:__________________  Checked: ______

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?