CISC 110 Assign 5: Die-Rolling Simulation

In this assignment, you will create an app that simulates throwing a six-sided die (only one die, rather than a pair of dice) many times in order to do a statistical analysis of how random the simulation actually is.

The user chooses the number of trials and specifies a particular die value (1,2,3,4,5, or 6). During each trial, the app will simulate rolling the die until the user's specified die value is rolled. After completing all of the trials, it calculates the average number of rolls it took to roll the user's die value over all of the trials. Since there are six numbers on the die, if there is an even probability of rolling any one of them, it should take an average of six rolls to roll a desired value. The user can also specify weights for each side of the die, so that the probability of it landing on one side will be greater than another. Your simulation will also include an animation of the die rolling and stopping on a random side, which will be controlled by a Timer.

Below are the main steps for your assignment.

1. In the lab, create a simplified version. For details, see the `Lab 5` instructions.

2. Design your die-rolling simulation, including what text fields and variables you need. The required user input from text fields is:
   - The number of trials to run
   - The die value to use in the simulation
   - Six weights, one for each side of the die, which add up to 1. For instance, if the user wanted to have approximately even probability that any of the six sides would be the result of a roll, the six weights could be: .16, .16., .17, .17, .17, and .17

3. Add your text fields and modify your runTrial function to read the input values from the text fields into variables and display the results in a text field.

4. Modify your rollDie function to include an if-else statement that assigns die1's roll value according to the weights specified by the user. For instance, if the weights are the example values given in step 2, you could assign a roll value of 1 for a result from Math.random( ) between 0 and .16, a roll value of 1 for a result from Math.random between .16 and .32, etc. The if-else must work no matter what weights the user enters, as long as they add up to 1, so variables must be used. You may assume that the user will only enter valid input values.

5. Modify your work from your lab by improving the image you use for your die and, if you wish, for your button.

6. Add an animation of your die rolling several times to simulate the rolls of the dice that that are being calculated. It should roll and display a random side each time and stay there for a second before it rolls again. Use a Timer to achieve this. For how to use

a Timer, look at the Wk5TimerEvents example on the CISC 110 web page in the Schedule table under *ActionScript and Flash Examples*.

Put the variable declaration for the Timer inside your class and just before the start of your constructor function, so that it's a global variable that you can refer to anywhere in your class. Add the listener for the Timer inside your constructor function. Start your Timer inside the button handler, so the dice-rolling animation only starts playing when the button is pressed. Right before you start the Timer, you could also reset it, so that your Timer will start over if the user presses the button a second time. If your Timer variable is called `timer`, the command to reset it is: `timer.reset( );`

Remember that you can use a second parameter when you create a new Timer, which specifies the number of times the Timer will go off before stopping, instead of running indefinitely. See the lecture notes for an example. For instance, you could play your die-rolling animation three times, once every two seconds, by specifying 2000 as the number of milliseconds for your Timer and specifying 3 as the second parameter. If you design your dice-rolling animation to be two seconds long, then it will finish at the end of each Timer time period.

At the start of your Timer event handler function, you could set the value of a text field in your die randomly with Math.random. Then you could play your die-rolling animation.

You can have a simple die-rolling animation, as long as the value of the die is set randomly with Math.random and as long as a Timer controls it. For instance, you could have the die be a simple square with a text field that shows the same random value for its entire roll. Then the next roll it would show another random value for its entire roll.

7. Once your die-rolling simulation is working the way you'd like, publish it and upload it to the CISC 110 web space.

Assignment 5 Marking Scheme (2% of final mark)

Marked out of 10:
1 mark: All input values read from text fields

1 mark: Output displayed in text field

3 marks: Weights added to allow "unfair advantage" when rolling die

3 marks: Die-rolling animation controlled by Timer

1 mark: Random roll value displayed in die-rolling animation on each roll

1 mark: Assignment 5 is published and uploaded on the CISC 110 website (`.swf` and `.html` files).