|  |  |
| --- | --- |
| Computer Science 111 Computer Science with Java IFall, 2015 |  |

**Lab Report – Week *12* – Event-Driven Multiplication Guessing Game**

*Leo Vergnetti*

*CSCI 111-online Fall*

**Assignment Analysis and Design**

*The assignment was to set up a flash cards style multiplication application. Two integers are randomly selected, and the user is asked to multiply them. The application will multiply the random integers and check it against the user’s answer. If the two match, then the user is informed via a JLabel with green text. If the answer is wrong red text is displalyed informing the user the problem was wrong, and moving on to the next problem. This is done inside a try-catch block in case the user entered something other than an integer in the box. If this happens, it tells the user the answer was invalid, and re asks for the answer. This will continue until the user quits the program. The FlashCardsFrame inherits from JFrame. It has a constructor, and two methods. The nextProblem() method sets up the next problem with two random integers generated by the Random class, and the checkAnswer() method checks the answer and dipslays the message (right, wrong, inadmissible). I also added an inner class listener which implements ActionListener. I set up two final integers in the class called QUESTION\_MODE which is set equal to 1, and ANSWER\_MODE which is set equal to 2. Then I used a third integer called mode to track the current mode through an if statement. This lets me use the same button for submitting the answer, and moving on to the next question. I also set up the listener on the answerBox textfield as well. This allows the user to press the return key instead of the button to move to the next question/submit answer.*

**Assignment Code**

*The source code is included in the zipped folder.*

**Assignment Testing**

*To test this program, I ran it a few times. I could only think of a few changing states. The first thing I checked was that the modes were switching properly when the button was pressed. I then tested that the integers were generating properly, and that the application was correctly evaluating the answer as correct/incorrect. Last I tested my try-catch block to ensure there wasn’t a parsing error if the user entered something other than an integer.*

**Assignment Evaluation**

*I did not particularly struggle with this assignment, however, I’m still not entirely satisfied with the code itself. It performs as the problem specifies it should, however there is slightly too much jammed into the GUI frame class than I would like. I think I could have set this up using something like an MVC pattern, so that the functionality (ie, generating the integers, testing the correct answer, passing string messages etc) was done inside a controller class, or at least the main class if I didn’t want to go that far. However, I left it as is since only 3 or 4 lines of functional code were placed inside the class. If I had done it this way, then I would have set my listener class as an inner class of FlashCardsController instead of inside the logic. Then I could have caught the ActionEvent there and passed off ALL functionality to the controller class. Then I’d let the GUI class be FlashCardsView and simply have the next problem be placed there instead.*