**Tiffany Lourdraj**

**ITSS 3312.003**

**Professor Short**

**09/04/2020**

**Project 1**

**Analysis, Design, Testing of Elementary flashcard Project**

This program consists of many layers including methods and switch statements which helps create a comprehensive solution for kid’s math learning tool. The main obstacles I ran into while building this program was that it was a challenge at first to figure out how to make sure the answer the user gives is correct, and that the total tallying of correct to incorrect answers. However, using a traditional if else statement and the operatives “ ==” or “!=”, you can count up the incorrect and correct answers. In addition, the dividing switch statement at first was a little challenging as well, but then I realized you actually have to multiply the two random numbers so you can have a easy whole number quotient for the elementary level math. For example, if the random numbers are 9 and 4, you can multiply them, that way you can get 36 and ask what is 36/4, instead of asking 9/4. At the end of the entire program, I was able to successfully loop through the n amount of questions, tally up correct/incorrect responses and print out the name, percentage, operation, date and time.

The design aspect is important. If you have code that is not indented properly, or doesn’t have an easy flow to it, other people reading your code might have a hard time following along. The flow of it went like this for me:

1. Calling upon methods
2. Initialize and define variables used in switch statements
3. Switch statements for addition, subtraction, multiplication and division
   1. Start with a for loop for number of questions
   2. Multiply range by random numbers
   3. Ask question
   4. Use if else statement to see if answer is correct
   5. After for loop is done, print name, time, date, type of problem and percent correct
4. Methods for name, operation, number of questions and range of the numbers

The methods being after the main body was suggested by the professor, and it makes sense because it is not cluttering up space in the beginning of the program, and is instead at the end of it. The flow is easy to understand, and you can go back and see at any point where the variable are being defined. Also when you define methods and variables, make sure to name it something that makes it clear what it does. For example, if it a variable that inputs the number of problems the user wants, I call it numProblems, with the second word always capitalized.

To test this code, it is a lot more than what originally meets the eye. There are 4 main switch statements that one must take into account. The first one was the addition switch. Originally, I ran into a issue, which is that the Math.random(); statement did not meet my expectations for the program originally. It included the number 0 because the random statement includes numbers from 0 to 1, which is then multiplied with the range of lowest to highest number. The problem is that, the user may not want 0 in the range, but the random method still multiplies the the range by a 0. I fixed this by including a line of code that said the following

number1 = (int)(Math.floor(Math.random() \* (highValue - lowValue + 1)) + lowValue);

This helped clear up the issue of 0 in the range. I also wanted to make sure while testing, the error percentage was right. At first, I ran in to the problem that the incorrect questions tally was not working properly, but that is because my formula for calculating incorrect responses was incorrect ironically. The formula is (correct responses)/(total number of questions)\*100. I accidently put the the wrong numerator, and I made the value a double instead of an int, which I fixed later on, because the decimal points went on forever if it was say 33.333… percent, when you can just round it down to 33%. In conclusion, I fixed any problems that might have arisen through my code by simply testing each operation and trying multiple numbers, and making sure I got some questions incorrect. That way i can see if my total percent at the end is accurate, and it was. The end output included my name, percent correct and the date and time, which was the expected result from the project.

**Example code for the Division switch statement:**

case "D":

for (int i = 1; i<=numProblems; i++)

{

number1 = (int)(Math.floor(Math.random() \* (highValue - lowValue + 1)) + lowValue);

number2 = (int)(Math.floor(Math.random() \* (highValue - lowValue + 1)) + lowValue);

System.out.print

("What is " + number1\*number2 + " / " + number2 + "? ");

result = input.nextInt();

if (result == (int)(number1\*number2/number2))

{

System.out.print("\ncorrect\n");

tallyCorrect=tallyCorrect+1;

}

else if (result != (int)(number1\*number2/number2))

{

System.out.print("\nincorrect\n");

tallyIncorrect= tallyIncorrect+1;

}

}

double dendTime = System.currentTimeMillis();

double dtotalTime = (dendTime-startTime)/1000;

double dtotalScore = (tallyCorrect)/numProblems\*100.0;

System.out.print("Session Summary ");

System.out.print("\n" + numProblems + " problems," + (int)tallyCorrect + " correct");

System.out.print("\n score is " + (int)dtotalScore + " , Time is " + dtotalTime + " seconds ");

System.out.print("\n Session for " + name + " was Division on " + java.time.LocalDate.now() + " at " + java.time.LocalTime.now() );

break;

}

Screenshot of methods:screenshot of example output: