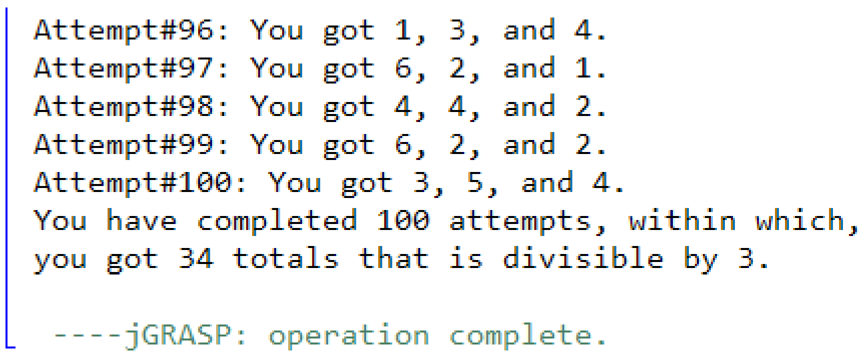
HW 3 Counter Controlled and Time Controlled Repetition

Develop a Java program for a 3-dice game. This game will roll the three dice together for 100 times. In each time, it will display the face value from each dice. The program will count how many times (out of these 100 attempts) did the three dice generate a total point that is divisible by 3, and display the result at the end.

Part of the execution example (due to the lengthy output)



```
Attempt#96: You got 1, 3, and 4.
Attempt#97: You got 6, 2, and 1.
Attempt#98: You got 4, 4, and 2.
Attempt#99: You got 6, 2, and 2.
Attempt#100: You got 3, 5, and 4.
You have completed 100 attempts, within which,
you got 34 totals that is divisible by 3.

 ----jGRASP: operation complete.
```

1. **Develop a flow chart for this problem. (6 pts)** In a word document, PPT file or a PDF file, develop a flowchart for this problem. Put your name, date, and HW3 on the top of the document.

Suggestion:

- You can start from a simple version of flowchart. It may take a few rounds of modification to get good ones.
- This program includes sequence structure, selection structure and repetition structure. Make sure you show them clearly.
- Make sure the texts on the shapes, and labels like true and false on the arrows are clear to read.
- You can use hand drawing, Word, PPT, Visio, LucidChart, etc. Make sure the submission have to be a PPT file or a PDF.
- No pseudocode is needed for this question.

2. **Write a Java program for this 3-dice game. This game will roll the three dice together for 100 times (i.e., a repetition structure). In each time, it will display the face values from the three dice. The program will add up the three values to get a total for each attempt, and count how many times (out of these 100 attempts) did the three dice generate a total that is divisible by 3 (i.e., when divided by 3, the remainder is 0), and display the result at the end. See the execution example above. (7 pts)**

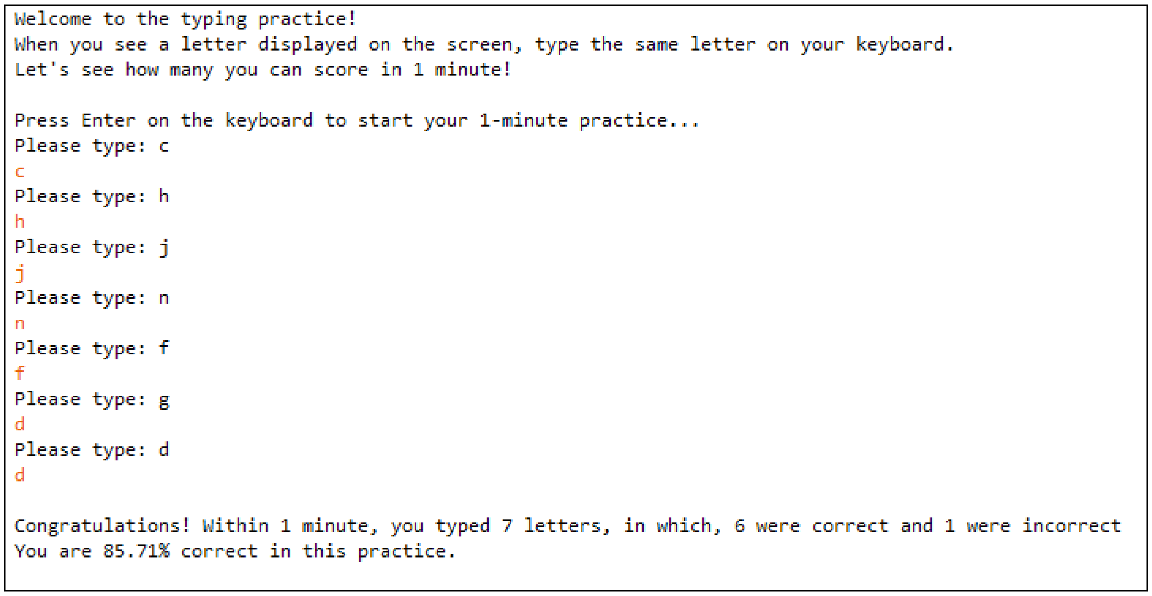
   Requirements:

   a) Your class should be named **ThreeDiceGame**, and your java file should be named **ThreeDiceGame.java**.

b) Although the Pseudocode is not required, you can start this part of the question by providing the class and method blocks with { }, indentation, and comment lines for major tasks in the program.
c) In this question, use **Math.random()** to simulate dice rolling. Make sure your dice will generate integer values in the range of 1 - 6. Use **DO-WHILE loop** for the 100 times dice rolling.
d) You will need two counter variables. One is to count for how many attempts you have rolled the three dice – to control the loop for 100 times. The other is to count how many times that your three dices gave a total that is divisible by 3.
e) As a counter variable, the attempt value starts at 0 and stops at 99 for 100 times. However, to display the face values in each dice rolling, you may want to use (attempt + 1), so it will print out the attempt# from 1 to 100.
f) Programming styles are always required.

3. Write a Java program for a typing practice for young kids to learn and practice typing on a computer keyboard. Let's assume this is the level 1 practice, where kids only type single letters in lower case. Your program should first display a welcome message and some instruction. Next, it will display one random letter in lower case (a – z) at a time, and the users should type this letter on the computer keyboard. See the sample execution of this program below. Users have 1-minute practice time. After 1 minute, the program will end the practice and display a practice summary, including the total letters the users have completed, and the percentage correctness in their practice. (12 pts)

Sample execution:



```
Welcome to the typing practice!
When you see a letter displayed on the screen, type the same letter on your keyboard.
Let's see how many you can score in 1 minute!

Press Enter on the keyboard to start your 1-minute practice...
Please type: c
c
Please type: h
h
Please type: j
j
Please type: n
n
Please type: f
f
Please type: g
d
Please type: d
d

Congratulations! Within 1 minute, you typed 7 letters, in which, 6 were correct and 1 were incorrect
You are 85.71% correct in this practice.
```

Requirements:
a) Your class should be named **TypingPractice**, and your source code file should be named **TypingPractice.java**.
b) Although you don't need to submit a pseudocode for this HW, it is always helpful to start your

Java file with a pseudocode writing, that provides the program structure and blocks, and some single-line comments for the major tasks.

c) You need a welcome and instruction message at the start of this program, and a summary message at the end. If your messages are too long to show in one line, you can use \n in the String part to break it into multiple lines.

d) **Some fun and useful feature**: Kids may read slowly, and their instruction reading time should not be included in this one-minute practice. So you can ask them: "When you are ready, press the enter key to start." In coding, you can have a statement to take user input, but no need to store the input in any variable, like:
```
      input.nextLine();
```
This is like a pretending action. We don't really need the user input, but just hold the program to allow as much reading time as the user needs. Only when the user presses the Enter key (meaning user has finished the input), the program will move on to the next task – starting the timer for practice.

e) In this question, use a **WHILE loop** for repetition, and use **the Random class** to take random values.

f) **To generate a random char value:** You need to take random char values in the range of lowercase a to z. Use the skills we learned in class for it.

g) **To take user input char value:** In the Scanner class and object, we don't have a convenient method to take char input directly. You can use next() to take a short String input, and then use charAt(0) to take the first character in this String and use it as a char value. (See textbook, 4.4.2 on page 131). So the sample code could be like:
```
      char answerLetter = input.next( ).charAt(0)
```

h) You can count the correct typing and incorrect typing separately. The expression (correct + incorrect) will provide total typing attempts. The expression
```
      correct / (correct  + incorrect)
```
will provide the correctness rate. Be careful that we need an accurate, double value for the correctness rate. How to make sure the result is in double form? (Hint: at least one of the operands for division should be in double.)

i) To display a correctness rate in a percentage, you can have the double rate value * 100, and in the output message, you can show this value followed by a % mark.

j) In the printf statement, a %f is needed to display a double value. You can use %.2f to display this double value with 2 decimal places only. To display a % mark after this value, you can use %% in the output message String. So like,
```
      %.2f%%
```

k) Programming styles are always required.

Submission:

- Upload the PPT or PDF document for the flowchart in Part A.
- For Part B, submit two java files.
- Make sure you hit the SUBMIT button to complete the submission, and wait a while until the files are fully uploaded. Check each file on Blackboard to avoid crashing files or incorrect files (like the .class files).