ICS3U1

Mid Term Programming Assignment Evaluation Rubric

Name:________________________________

Program/Assignment Name: ____

Achievement Category	Level 1 (50 – 59%)	Level 2 (60 – 60%)	Level 3 (70 – 79%)	Level 4 (80 – 100%)
Boolean Operators, Math, String Manipulations **(/3 APP)**	Program uses at least 1 different variable type, performs a mathematical calculation or string manipulation	Program uses at least 2 different variable types, performs a mathematical calculation or string manipulation	Program uses at least 2 different variable types, performs mathematical calculations or string manipulations	Program uses at least 3 different variable types, performs many mathematical calculations or string manipulations
Selection Structures **(/3 APP)**	Program uses if statements.	Program uses if statements, “else if” statements.	Program uses if statements, “else if” statements, and else statements in a logical way	Program uses if statements, “else if” statements, and else statements in a logical and efficient way. If statements have multiple conditions
Repetition Structures **(/3 APP)**	Program uses while or for loops.	Program uses while or for loops in a logical way.	Program uses while and for loops in a logical way. Has multiple loops.	Program uses while and for loops in a logical and efficient way. Has multiple loops.
Internal Documentation **(/2 COM)**	Code has a comment in it.	Code has a header or comments at every major block of code.	Code has a header as well as comments at every major block of code.	Code has a header as well as comments at every major block of code. Comments are clear and help to understand the code.
Programming Style **(/2 COM)**	Code is indented. And the variable names use prefixes.	Code is sort of indented correctly, and the variable names have some correct prefixes (int, str, chr, bln, dbl)	Code is mostly indented correctly, and the variable names have the correct prefix (int, str, chr, bln, dbl)	Code is perfectly indented correctly, and the variable names have the correct prefix (int, str, chr, bln, dbl). Variable names make sense.
User Interface – UI Design Document **(/5 TIPS)**	UI document has less than 4 screens depicting different scenarios in your program.	UI document has at least 4 screens depicting different scenarios in your program.	UI document has at least 4 screens depicting different scenarios in your program. Each screen needs a description of the screen’s functionality.	UI document has at least 5 screens depicting different scenarios in your program. . Each screen needs a description of the screen’s functionality using arrows pointing to specific areas of the screen.
Problem Solving – Variable / Input / Calcuation/Output Document **(/5 TIPS)**	VICO document has at least 1 inputs described, 1 calculation or logic operations described, and 1 outputs described.	VICO document has at least 3 inputs described, 3 calculation or logic operations described, and 3 outputs described.	VICO document has at least 5 inputs described, 5 calculation or logic operations described, and 5 outputs described.	VICO document has at least 7 inputs described, 7 calculation or logic operations described, and 7 outputs described.
Daily Log **(/2 COMM)**	Daily Log has some of the days’ activities documented with tasks completed and tasks “to do”	Daily Log has most of the days’ activities documented with tasks completed and tasks “to do”	Daily Log has all of the days’ activities documented with tasks completed and tasks “to do”	Daily Log has all of the days’ activities documented with tasks completed and tasks “to do”. Daily log is formatted nicely and is easy to follow.
Solution of problem **(/6 APP)**	Program compiles and runs. Program meets at least one requirement.	Program compiles and runs. Program meets some of the requirements.	Program compiles and runs. Program is usable based on the requirements descriptions. No major logic errors.	Program compiles and runs. Program is usable based on the requirements descriptions. Program has more than the required features (like cheats, or a special level/mode)

Blue – Java Code and Comments

Green – User Interface Document

Grey – Input Calculation Output Document

Brown – Daily Log of Activities

ICS3U1 – Midterm Programming Assignments

1 - Guess The Secret Letter

Write the following game:
A person starts with 50 points and must find a secret letter of the alphabet. By typing any word, the program will respond with the number of occurrences of the secret letter. If the letter does not occur in your word, you lose 10 points. If the letter occurs at least once, but less than 5 times, you lose only 5 points. If the letter occurs 5 times or more, the program halts and you win. If you get less than 0 points, you lose.
There are many additional features that may be added.
Example: the secret letter is: F

```
Would you like to play "find the secret letter"? y
You have 50 points, what is the word? Maryhadalittlelamb
That word has 0 secret letters.
You have 40 points, what is the word? Herfleecewaswhiteassnow
That word has 1 secret letters.
You have 35 points, what is the word? Feefiefoefum
That word has 4 secret letters.
You have 30 points, what is the word? ffffffff
Congratulations, you have won.
```

2 - What Is The Word?

Write a program where you have listed 20 words in an array of String. The program will randomly select one of the words and let you guess the word by buying a letter, similar to "wheel of fortune". If you run out of points before the word is guessed, you lose.
There are many additional features that may be added; If you make the list longer, or use phrases instead of single words, it would make the game more challenging.

Assume that the word is: "MYSTERY"

```
You have 7 points, and the word is *******
please enter a letter: E
You have 7 points, and the word is ****E**
please enter a letter: N
You have 6 points, and the word is ****E**
please enter a letter: A
You have 5 points, and the word is ****E**
please enter a letter: O
You have 4 points, and the word is ****E**
please enter a letter: I
You have 3 points, and the word is ****E**
please enter a letter: Y
You have 3 points, and the word is *Y**E*Y
please enter a letter: S
You have 3 points, and the word is *YS*E*Y
please enter a letter: T
You have 3 points, and the word is *YSTE*Y
please enter a letter: R
You have 3 points, and the word is *YSTERY
please enter a letter: M
You have 3 points, and the word is MYSTERY
CONGRATULATIONS, you have guessed the word.
```

3 - Slice And Dice

An interesting game with dice between two players goes as follows: Each player will try to increase her total set of points, and the first one to reach 100 points wins. Players alternate turns. During a turn, a player may roll as many times as she pleases, and each sum of the pair of dice will be added to her total. However, if the player rolls doubles (i.e. the same value on each die) then that player loses all the points gained during that turn and must pass the dice to her opponent. If the player passes the dice to her opponent before she rolls doubles, her points will be permanently added to her total. Write a program to simulate this game.

4 - Tic-Tac-Toe

Write a two player tic-tac-toe game. Using print statements print the playing board to the screen. Because this is a command line game, user input needs to be done with coordinates. The program also needs to check if there is a tic-tac-toe. Lastly, the program needs to count the number of wins for the “x” and “o” players until they want the game to end.

5 - Math Training Game

Create a math game for elementary school kids. Addition and subtraction are the basic skills needed to be tested in this program. The program will randomly generate whole numbers and equations for the students. For example, the first question might ask the student to add two single digit numbers. The second question might ask to subtract two single digit numbers. The difficulty of the questions should increase every 16 questions. Difficulty can be increased by increasing the number of digits being added/subtracted.
When subtracting, ensure that the first number is larger than the second. The student’s current “score” should be written on the screen. Example: You got 20 of 25 right. 80%

6 – Red Light, Green Light

Create a game based on the childhood playground game. Four players. Four horizontal lines. The lines start short and grow to the right every “turn”. Each turn will be as follows:
Player 1 will be asked if he wants to move or stay
Player 2 will be asked if he wants to move or stay
Player 3 will be asked if he wants to move or stay
Player 4 will be asked if he wants to move or stay
After all players are asked, the computer will generate a random number between 1 and 10. If the number is 1-7, the light is green. If the number is red, the light is red.
Any players that have selected move while the light is red are eliminated. If the light is green, everyone is safe but the ones that selected move get to move forward (their line gets longer),
Whoever survives or who makes it to 40 places wins!

7 – Break The Code

Write a program where you have listed 20 words in an array of String. The program will randomly select one of the words. The program will measure the length of the word and randomly pull the letters of the word out and print it on the screen. From the scrambled word, the user will have to guess the code (word) right. If they do not do it in the correct time, the program will print “INTRUDER” to the screen. If they get the code right, the program will print “WELCOME” to the screen. The number of times to try will be determined by the number of letters divided by 3 (Rounded). 9 letters max in the code.

8 – Hangman

Write a program where you have listed 20 words in an array of String. The program will randomly select one of the words. The program will start of with blank letters “ _ _ _ _ _ _” (Canada). And each turn will randomly expose a letter. Each turn the user will take a guess what the word is. If the guess is incorrect, add a piece to the hanging man drawing. Once all pieces are drawn… GAME OVER. A typical hangman drawing has: Head, neck, body, arms, legs, rope, horizontal pole, vertical pole. Therefore, your puzzle should have at least 7- 9 letters in it. Each “piece” of the drawing could/should be done as separate drawings, called in different if/elseif blocks.