Assignment 1 VB program (15%)
(Due: March 20, 2017 at 23:59)

Learning Outcomes

1. Apply programming concepts to solve business problems
2. Describe the logic and flows of given programs
3. Predict the output of a program
4. Write programs with common programming practices
5. Identify and fix logical and run-time errors in programs

Background

In a re-union dinner, Samuel re-called from the “collective memories” a Mathematics game in his high school life: The first player has a “Lucky number” from 1 to 30 randomly in his / her mind. And the second player tries to guess what the number is in as few guesses as possible, with the help of “hints” from the first player. Once the “Lucky number” is correctly guessed, the game cycle will repeat with a different “Lucky number”. And the whole game ends when the second player has made 3 correct guesses (i.e. 3 game cycles completed) or he / she has used up the maximum of 21 guesses (shared by all 3 different “Lucky numbers”). Now, you are asked to implement this game in Visual Basic. The computer will be the first player while the user will be the second player.

Requirements

1. Graphical User Interface (GUI): You are free to design your own interface. But here are the required components in the interface:
   I. A title showing the program name, e.g., “Samuel’s Guessing Game”.
   II. A label showing the remaining number of guesses available. (counting from 21 down to 0)
   III. A label showing the ordinal number of the guess for the current game cycle. (counting from 1 up to 21, reset to 1 when the game cycle repeats)
   IV. A label displaying the current score.
   V. A label showing the current number input by the user.
   VI. A label displaying the guessing results, with a hint for a wrong guess.
2. Here are the game logics and rules:
   a. When the game cycle starts, the program randomly generates a “Lucky number” NOT shown to the user. The user input the first guess number using the textbox. Only input within the **integer** range (1-30) is permitted.
   b. If the guess number is NOT EQUAL to the “Lucky number”, do the following:
      i. Display the guessing result, e.g. “This is NOT the Lucky number!”
      ii. Display ONE of the following hints:
      “The Lucky number is larger than your guess.”
      “The Lucky number is smaller than your guess.”

        iii. Increase the ordinal number of the guess for the current game cycle by 1.
        iv. Deduct the remaining number of guesses available by 1.
    c. If the guess number is EQUAL to the "Lucky number", do the following:
        i. Display the guessing result, e.g. "Congratulations! This is the Lucky number!"
        ii. Increase the score by 10 points.
        iii. If the "Lucky number" is found in 5 guesses or less (i.e. the ordinal number of the guess for the current game cycle is less than or equal to 5), increase the score by 5 points as a bonus.
        iv. Reset the ordinal number of the guess for the next game cycle to 1. (Not needed for the third game cycle.)
        v. Deduct the remaining number of guesses available by 1.
    d. The game will end after 3 game cycles completed or the maximum of 21 guesses has reached. In case of maximum number of guesses NOT reached (ie. the remaining number of guesses available is greater than 0), increase the score by points equal to the remaining number of guesses as a bonus.

3. Your program must have comments (or documentation) to explain the code.
4. Your program must use condition statement; i.e., if-then-else or select-case.
5. Your program must set both Option Explicit and Option Strict to be On.

Marking Criteria

Area	Percentage
Logic Flow and Accurate Calculation • Satisfying the requirement stated in #2 and #3	**40%**
User Interface • A user-friendly interface	**20%**
Condition statements • Appropriate use of if-then-else or select-case statements	**20%**
Comments and Programming Styles • Providing meaningful comments • Using meaningful controls, variable and constant names	**20%**
Total :	**<u>100%</u>**

FAQs

1. **I do not know how to generate a random number. What should I do?**
   You may simply use these two lines of code:

```
Randomize()
CInt(29 * Rnd() + 1)
```

   where CInt() is to convert a number into an integer, Randomize() is the preparation of random number generator, and Rnd() is to generate a random number.
   This statement will generate a random number in the range of 1 to 30.

2. **I think that I am not good at programming. What should I do?**
   We encourage students to start thinking about the programming logics first. For example, how to get the number input by the user and compare it with the “Lucky number”? You may also need to think about the logics of the remaining number of guesses, ordinal number of the guess for the current game cycle and the score calculation.
   If you are still very lost, come and talk to Karen, Kelvin or James.
3. **I do not know how to exit the program. What should I do?**
   You may use the following statement to terminate the program:

   ```
   Application.Exit()
   ```

Submission Guidelines

1. Please zip your program files and name the zip file as your student ID (e.g., 07123456.zip).
2. Submit your zip file to Canvas. Multiple submissions are allowed but only the last submission will be graded. Therefore, you need to make sure the last submission before the deadline is the best and more importantly it works.
3. Late submission within 24 hours will result in a penalty of 30% deduction in your total marks. No further late submission is allowed.
4. Plagiarism (Copying) is a serious concern. Students might search for information/reference/program on the Internet and "Copy & Paste" it directly in their assignments. Hence, a strict rule is applied that **MORE THAN SEVEN WORDS** copied from a source is considered as a cheating. The **MINIMUM** penalty is zero marks for the particular assignment.