GDAPS1 – Practice Exercise

Multiple Forms

# Objective

Create a number guessing game to practice creating additional forms in one project, opening them programmatically and getting them to communicate.

# Details

Start with a new Windows Forms project in Visual Studio.

## Setup: The Parameters Form

Form1 (or whatever you name it) will let the user set up the guessing game parameters. Set up it up as follows:

* A start button
* Several text boxes for user input (with appropriate labels):
  + The **low end** of the random number range
  + The **high end** of the random number range
  + How much **time** they’d like for the game
  + Pre-populate the text boxes above with default values

## Setup: The Game Form

Create a second windows form class called GameForm. Be sure to select “Windows Form” instead of “Class” when creating it, as that will create the designer file appropriately and give you the required using statements.

It should have the following controls:

* A text box for the user’s **current guess**
* A label for **output** (whether the current guess is high or low)
* A **guess button** the user can click
* A **progress bar** that counts down while the game is in progress
* A **timer** control that ticks once per second

The GameForm class itself will need a parameterized constructor to accept the data for the game (low, high and time). It’ll also need to do things like generate a random number, start the timer, etc.

The timer’s tick event will need to update the progress bar accordingly and potentially cause the game to be over.

## Playing the Game

This part will require you to add code to both form classes.

When the program starts up, the user should be able to start the game or change the values in the text boxes to customize the game. When the user clicks start, verify the game parameters are valid:

* There’s a valid value for the “low” number (it’s actually a number)
* There’s a valid value for the “high” number (it’s actually a number)
* Low is less than high
* There’s at least 10 numbers between low and high
* Time is greater than zero

If the input is invalid for any reason, pop up a message box with a useful error message. Do not start the game until the input is valid.

Once you’ve verified the data is valid, create a new GameForm and show it to the user. Use ShowDialog( ), so that the user can’t start a second game while the first is active.

At this point, the user should type numbers into the guess text box and click the guess button. Update the output label after each guess, letting the user know if they were high or low.

* If the user guesses the correct number before the timer runs out, stop the timer and pop up a congratulatory message box.
* If the user fails to guess the number, pop up a failure message box.
* In either case, once the user closes the message box, close the form itself by calling this.Close( ).

The user should be able to play again as many times as they’d like. Feel free to experiment with KeyDown events so the user can just press enter instead of having to click the guess button each time.

# Submission

All of your work must be commented and follow this course’s coding standards. **Read through the Coding Standards document (located in MyCourses) to check over your code before you complete your program. Make sure you follow the coding standards for all code you create.**

1) Submit: Submit your program to the appropriate Assignments dropbox in MyCourses.

2) Check-off: Show your working program to the instructor or TA. If you do not finish before class ends, complete the exercise for homework and show one of us in-class on the next class period. If your program works as expected, you will be “checked off” to earn credit for the exercise.