Console I/O Library

When testing concepts or building simple solutions using a console application, it is often handy to have a menu of actions that may be taken. Writing a menu that facilitates numeric parsing, input validation, and exception handling can be a bit of work. Writing similar menus over and over again is just insanity, especially when we have the opportunity make our own DLLs (Dynamic Linked Libraries)!

For this lab, you will create a DLL (NOT an executable program) for generating menus in console applications. Your DLL will have methods that prompt the user for a valid byte, short, int, float, double, decimal, or long AND will return the input as the proper data type (which means you need to parse the input). These numeric prompts should also allow you to dictate a range of acceptable values depending on your current needs (ex. Between 1 and 10, pick door 1 or 2 or 3, enter your year of birth, etc). Your DLL will also have methods that prompt for a string, a single char, or a bool (typically for Yes/No questions). *All* your methods should handle exceptions and bad input (such as alpha characters for numbers, numbers outside the bounds of the range, and empty responses for strings when input is required) by informing the user the input was bad and looping through the prompt until a valid input is given. Each menu function that is public will return the proper data type and ONLY valid values.

You will be provided with a class containing a defined namespace, class name, and method stubs. **You are NOT allowed to change the names or signatures of the namespace, class, or methods.** Your job is to fill in the methods to meet the requirements described above. You are allowed to add any helper methods or additional functionality. However, only the functionality listed in this document will be tested and graded.

# Special Delivery Instructions

Your deliverable for this assignment is the DLL itself, not the code base. **As a final requirement,** please, name your DLL file LastnameF\_CIO, where you replace “LastnameF” with your last name and first initial. For example, were it my DLL, I would go into the project properties in Visual Studio and change the Assembly Name field to read “KrebsJ\_CIO”. When I deliver my submission, I would upload the KrebsJ\_CIO.dll file directly to the LMS (no zipping or compressing required).

# Rubric

Please note that the following items are graded in a binary fashion; either you met *all* the requirements of the item, or you didn’t. No partial points will be given for a single line item.

**Automatic Zero:** Your deliverable is NOT a DLL, is not named correctly, is not testable and/or does not conform to the provided class interface.

(5 points) PromptForBool prompts for a bool, allowing a message to be dictated by the calling program. This function then returns the bool.

(8 points) PromptForByte prompts for a byte, allowing a message and range to be dictated by the calling program. This function then returns the byte.

(8 points) PromptForChar prompts for a char, allowing a message and range to be dictated by the calling program. This function then returns the char.

(8 points) PromptForDecimal prompts for a decimal, allowing a message and range to be dictated by the calling program. This function then returns the decimal.

(8 points) PromptForDouble prompts for a double, allowing a message and range to be dictated by the calling program. This function then returns the double.

(8 points) PromptForFloat prompts for a float, allowing a message and range to be dictated by the calling program. This function then returns the float.

(5 points) PromptForInput prompts for a string, allowing a message to be dictated by the calling program. This function then returns the string.

(8 points) PromptForInt prompts for an int, allowing a message and range to be dictated by the calling program. This function then returns the int.

(8 points) PromptForLong prompts for a long, allowing a message and range to be dictated by the calling program. This function then returns the long.

(10 points) PromptForMenuSelection generates a menu from a collection of strings, allowing the calling program to optionally include a quit option. The normal options are numbered starting at 1, with 0 reserved for quit.

(8 points) PromptForShort prompts for a short, allowing a message and range to be dictated by the calling program. This function then returns the short.

(16 points) The DLL functions handle exceptions gracefully, validate input, and loop until a valid input is given.