Input validation

Setup

1. Create a new Visual Studio C++ project called **InputValidation**.
2. Add 3 new source files to the project, called **Main.cpp**, **ConsoleInput.cpp**, and **ConsoleInputTest.cpp**.
3. Add 2 new header files to the project, called **ConsoleInput.h** and **ConsoleInputTest.h**.
4. Each **.cpp** file (except **Main.cpp**) should **#include** only the corresponding **.h** file (using quotes, not angle brackets).
5. Each **.h** file (as well as **Main.cpp**) should:
    a. **#include** any required library headers, such as **iostream**, **string**, etc. (using angle brackets).
    b. **#include** any required project headers, such as **StringConversions.h**, etc. (using quotes).

A: ConsoleInput functions

1. char **readChar()**
    a. Read a character from the user through the console, using **cin**.
    b. Ignore all remaining characters in **cin**, up to and including the newline character.
    c. Return the character.
2. string **readWord()**
    a. Read a string from the user through the console, using **cin**.
    b. Ignore all remaining characters in **cin**, up to and including the newline character.
    c. Return the string.
3. string **readLine()**
    a. Declare a string.
    b. Call the **getLine(…)** function, passing **cin** and the string variable as arguments.
    This function, from standard library header **string**, will fill the given string with a line of text from the given input stream (in this case, **cin**).
    c. Return the string.
4. int **readInt()**
    a. Read an integer from the user through the console, using **cin**.
    b. Repeatedly, as long as the input was not a valid integer:
        i. Clear **cin**'s error state.
        ii. Ignore all remaining characters in **cin**, up to and including the newline character.
        iii. Display a clear and informative error message to the user.
        iv. Re-read an integer from the user.
    c. Return the integer.
5. double **readDouble()**
    a. Read a double from the user through the console, using **cin**.
    b. Repeatedly, as long as the input was not a valid double:
        i. Clear **cin**'s error state.
        ii. Ignore all remaining characters in **cin**, up to and including the newline character.
        iii. Display a clear and informative error message to the user.
        iv. Re-read a double from the user.
    c. Return the double.

A: ConsoleInput functions (continued)

6. char **readChar(string message)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readChar()** and return the result.
7. string **readWord(string message)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readWord()** and return the result.
8. string **readLine(string message)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readLine()** and return the result.
9. int **readInt(string message)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readInt()** and return the result.
10. double **readDouble(string message)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readDouble()** and return the result.

A: ConsoleInputTest functions

1. Declare a constant integer global variable **iterations**, and set it to be 5, for example.
2. void **testReadChar()**
    a. Display an appropriate test title in the console.
    b. Declare an array of characters, with size equal to **iterations**.
    c. Using a loop, call the **readChar(string)** function a total number of times equal to **iterations**.
    d. In each case, store the return value of the function call in the corresponding array element.
    e. Using a second loop, display each element of the array.
3. void **testReadWord()**
    a. Test the **readWord(string)** function in an analogous way.
4. void **testReadLine()**
    a. Test the **readLine(string)** function in an analogous way.
5. void **testReadInt()**
    a. Test the **readInt(string)** function in an analogous way.
6. void **testReadDouble()**
    a. Test the **readDouble(string)** function in an analogous way.

B: ConsoleInput functions

1. char **readChar(char options[], int optionsCount)**
    a. Read a character from the user using **readChar()**.
    b. If the user's character is <u>not</u> in the **options** array:
        i. The character is invalid, so display a clear and informative error message.
        ii. Repeat from (a).
    c. Else, return the character.
2. int **readInt(int min, int max)**
    a. Read an integer from the user using **readInt()**.
    b. If the user's integer is <u>not</u> between **min** and **max** (inclusive):
        i. The integer is invalid, so display a clear and informative error message.
        ii. Repeat from (a).
    c. Else, return the integer.
3. double **readDouble(double min, double max)**
    a. Read an double from the user using **readDouble()**.
    b. If the user's double is <u>not</u> between **min** and **max** (inclusive):
        i. The double is invalid, so display a clear and informative error message.
        ii. Repeat from (a).
    c. Else, return the double.
4. char **readChar(string message, char options[], int optionsCount)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readChar(string, char)**, passing the appropriate arguments, and return the result.
5. int **readInt(string message, int min, int max)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readInt(int, int)**, passing the appropriate arguments, and return the result.
6. double **readDouble(string message, double min, double max)**
    a. Display the message to the user through the console, using **cout**.
    b. Call **readDouble(double, double)**, passing the appropriate arguments, and return the result.

B: ConsoleInputTest functions

1. void **testReadChar(char options[], int optionsCount)**
    a. Test the **readCharInt(string, char[], int)** function.
2. void **testReadInt(int min, int max)**
    a. Test the **readInt(string, int, int)** function.
3. void **testReadDouble(double min, double max)**
    a. Test the **readDouble(string, double, double)** function.