CS 3005: Programming in C++

User Interaction

Introduction

Interactive applications need to send text messages to users and receive text messages from users. We will create a class to help us manage these interactions. The class will keep track of the input and output streams used to communicate with the user, and standardize our interactions.

Assignment

Create a program that will ask the user for an integer (int), a floating point number (double), and a word (std::string). The program then should display a number of lines of text based on the value of the integer. For example, if the integer is 5, there will be 5 lines of text. If the integer is 0 or less, there will be 0 lines of text. The program will have an exit status (return value from main) that is equal to the user's integer.

Each line of text will have this format:

int double word

Where int is the line number, starting at 1, double is the user's floating point input, and word is the user's word input.

For example, the program interaction may look like this:

\$./questions3
Favorite integer? 3
Favorite double? 1.23
Favorite word? yellow
1 1.23 yellow
2 1.23 yellow
3 1.23 yellow

Programming Requirements

You must create following classes and functions, in the required locations, with the required API and functionality.

Create [lib/.gitignore]

Create a directory named [lib]. This will store the libraries of object code you will build for this project.

The file lib/.gitignore needs to store one line of text:

*.a

This will prevent the library files, which are *derived files* from being committed to the repository. We will always build them in the working directory.

Create include/.gitignore

Create a directory named *include*. This will store the header files for all of the files in the *lib* directory.

The file include/.gitignore needs to store one line of text:

*.h

This will prevent these copies of the header files, which are *derived files* from being committed to the repository. We will always maintain the originals in other directories.

Create [library-application/ApplicationData.{h,cpp}]

ApplicationData Class

Data Members:

- std::istream& mInputStream; Stream to read input from (e.g. std::cin).
- std::ostream& mOutputStream; Stream to write output to (e.g. std::cout).

public Methods:

- ApplicationData(std::istream& input_stream, std::ostream& output_stream);
- int getInteger(const std::string& prompt); Writes the given prompt to moutputStream and then reads in an integer and returns it. If there is any error, getInteger will return 0.
- double getDouble(const std::string& prompt); Writes the given prompt to moutputStream and then reads in a double and returns it. If there is any error, getDouble will return 0.
- std::string getString(const std::string& prompt); Writes the given prompt to mOutputStream and then
 reads in a string and returns it. If there is any error, getString will return the empty string.
- std::istream& getInputStream(); Returns the mInputStream.
- std::ostream& getOutputStream(); Returns the moutputStream.

Create [library-application/Makefile]

This file must contain rules such that any of the following commands will build the <code>libapplication.a</code> library:

- make
- make all

This file must contain rules such that the following command will install the <code>libapplication.a</code> library into the <code>../lib</code> directory, and <code>ApplicationData.h</code> into the <code>../include</code>:

• make install

Create [library-commands/questions3_aux.{h,cpp}]

Functions:

• int questions3 (ApplicationData& app_data); Runs the program explained above, using app_data to read/write from/to the user. Returns the value of the given integer.

Create library-commands/Makefile

This file must contain rules such that any of the following commands will build the libcommands.a library:

- make
- make all

This file must contain rules such that the following command will install the <code>libcommands.a</code> library into the <code>../lib</code> directory, and <code>questions3_aux.h</code> into the <code>../include</code>:

• make install

Create program-questions3/questions3.cpp

Functions:

• int main(); The entry point to your program. Creates an ApplicationData and calls questions3 with it.
Returns the value given from questions3.

Create program-questions3/Makefile

This file must contain rules such that any of the following commands will build the questions3 program:

- make
- make all
- make questions3

Create program-questions3/.gitignore

The file program-questions3/.gitignore needs to store one line of text:

questions3

This will prevent the executable program questions3 from being committed to the repository. It is a *derived file*.

Update Makefile

Update the project-level Makefile so that make and make all in the project directory will call make install in the library-application and library-commands directories, and make in the program-questions3 directory.

Additional Documentation

- for loop
- <u>cin</u>
- <u>>> operator</u>
- <u>cout</u>
- << operator
- <u>References/&</u>
- <u>string</u>
- <u>const and const references</u>
- <u>#include</u>

Grading Instructions

To receive credit for this assignment:

- your code must be pushed to your repository for this class on GitHub
- all unit tests must pass
- all acceptance tests must pass
- all programs must build, run, and execute as described in the assignment descriptions.

Extra Challenges (Not Required)

• Can you submit a googol (10^100) as your favorite double? Can you find a way to do it without typing out 100 zeroes?