Program #2
Due: Thursday February 1st, 2018 at 11:59:00 pm

Instructor	Dr. Stephen Perkins
Office Location	ECSS 4.702
Office Phone	(972) 883-3891
Email Address	stephen.perkins@utdallas.edu
Office Hours	Tuesday and Thursday 1:30pm – 4:00pm and by appointment

TAs

CS/SE 3377.002 - Vatsalkumar Patel - drp140230@utdallas.edu

CS/SE 3377.501 - Alexander Lee Hayes - alh170730@utdallas.edu

CS/SE 3377.502 - Dongye Meng - dxm163630@utdallas.edu

Purpose

Demonstrate the ability to download, install, and use 3rd party code. Demonstrate the ability to control compiler include paths. Demonstrate the ability to parse and use command line arguments. Demonstrate the ability to utilize basic C++ File I/O to read and write files to the file system.

Assignment

You are to create a C++ program that opens an input file, reads and process the data in that file, and then writes the (possibly modified) data to a new output file. Your program should:

- Accept arguments that are presented on the command line (see ***Usage*** box below).
- Use the *Templatized C++ Command Line Parser* for argument processing (See the "***Templatized C++ Command Line Parser***" below). Illegal or missing options should be handled properly.
- Store the parsing results into a C++ STL map. The map file must be indexed by a C++ enumerated list (enum) and should hold strings for values.
- Within your code, you must query your map to get the values of the parsed command line arguments.

Finally, your program must present the correct behavior. The behavior is:

```
USAGE:

   ./program2  [-l] [-u] [-o <output filename>] [--] [--version] [-h] <input filename>


Where:

   -l,  --lower
     Convert all text to lowercase
```

```
-u,  --upper
  Convert all text to uppercase

-o <output filename>,  --outfile <output filename>
  The name of the output file

--,  --ignore_rest
  Ignores the rest of the labeled arguments following this flag.

--version
  Displays version information and exits.

-h,  --help
  Displays usage information and exits.

<input filename>
  (required)  Input file
```

Your program should open the input file for reading and the output file for writing. It should copy all data from the input file to the output file while optionally converting case as directed by the command line arguments. -u and -l are mutually exclusive. Only one may be given but neither is required. If the output file is not specified (it is optional), you should write to a file with the default name of "output.txt".

Deliverables

You must submit your homework through ELearning. You must include your .cc files and any header files you create (not TCLAP header files). All source files need to have your name, email, and course number commented at the top.

Notes

If the compiler complains that /tmp is full ("No space left on device"), then set your TMPDIR environment variable to /scratch. You can do this in your bash shell and in your shell scripts like this

```
TMPDIR=/scratch; export TMPDIR
```

You will need to use the g++ -I (Capital I) command line argument to point to the include folder that contains the TCLAP folder of header files.

All editing must be done within the shell using a text editor and must compile on cs1.utdallas.edu

No late homework is accepted.

Templatized C++ Command Line Parser

It is often the case that you will want to use third party code. For Program 2, we will use a library from a well-known open source repository. Specifically, we will utilize the Templatized C++ Command Line Parser (TCLAP).

TCLAP is a SourceForge project. SourceForge is a web-based source code repository. It acts as a centralized location for software developers to control and manage free and open-source software development. All information about the parser may be found at:

http://tclap.sourceforge.net

Installing TCLAP

From the Unix Shell, you can download the TCLAP tarball by using this command:

```
wget http://sourceforge.net/projects/tclap/files/tclap-1.2.1.tar.gz
```

Executing this command should leave you with a new file in your folder called tclap-1.2.1.tar.gz.

Once you have downloaded the TCLAP tarball, you must untar the results. To uncompress and untar the file, you can execute this command:

```
tar zxfv tclap-1.2.1.tar.gz
```

This will create a folder called tclap-1.2.1 with a lot of stuff in it. For our purposes, the only thing of interest will be the folder called include.

Finally, you must copy of the TCLAP folder named "include" into the project folder for your homework assignment. The command will be similar to this:

```
cp –R tclap-1.2.1/include  ~/Program2
```

The –R flag means "recursive". It will copy all the contents of the include folder. The destination listed above (~/Program2) will need to be modified so that it points to your folder where you are creating the code for Program 2.

When the cp command completes successfully, you should have a s folder called "include" inside your Program2 folder. If you change into the include folder, you should see a second folder called tclap. Inside that folder, you should see approximately 25 header files (.h files).

Using TCLAP

The full directions on using TCLAP in your C++ code are listed on the TCLAP site. However, you will need to also use a new compiler flag to help with the compilation of your source files that use TCLAP. Specifically, you will need to use the –I flag to tell the compiler where to search for the new TCLAP header files you just downloaded and installed. So, when calling your compiler you will need to do something similar to:

```
g++ –I ~/Program2/include program2.cc –o program2
```

The –I command should be followed by the path to the include folder that contains the TCLAP folder of header files.

See class notes for more details!