CSE-381: Systems 2

Homework #6 Part B

Due: Tuesday October 30 2018 before 11:59 PM

Email-based help Cutoff: 5:00 PM on Sun, Oct 21 2018

Maximum Points: 30

Submission Instructions

This homework assignment must be turned-in electronically via Canvas CODE plug-in. Ensure your C++ source code is named `MUID_hw6.cpp`, where `MUID` is your Miami University Unique ID. Ensure your program compiles without any warnings or style violations. Ensure you thoroughly test operations of your program as indicated. Once you have tested your implementation, upload the following:

1. The 1 C++ source file developed for this homework.

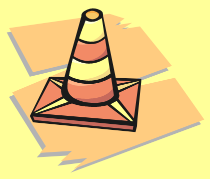
General Note: Upload each file associated with homework individually. Do not upload archive file formats such as zip/tar/gz/7zip/rar etc.

Objective

The objective of this homework is to:

- A multithreaded C++ program (synchronization is not needed)
- Continue to gain familiarity with use of HTTP protocol for communication
- Continue to gain familiarity with I/O streams & string processing

Grading Rubric:



The program submitted for this homework **must pass necessary base case test(s) in order to qualify for earning any score at all**. Programs that do not meet base case requirements will be assigned zero score!
Program that do not compile, have a method longer than 25 lines, or just some skeleton code will be assigned zero score.

- See points set for each command and overall structure, organization, **conciseness**, variable-names, etc. in the next page. **If your methods are not concise points will be deducted**.
- **-1 Points**: for each warning generated by the compiler (warnings are most likely sources of errors in C++ programs)

NOTE: Violating CSE programming style guidelines is an error! Your program should not have any style violations.

Problem summary

Develop a multithreaded C++ program that uses multiple threads to process a list of URLs specified as command-line arguments. For each URL, the program must print (in the same order in which URLs are specified) the number of words, and number of valid English words. A dictionary is supplied to determine valid English words.

Starter code

Starter code is provided to streamline the following operations for you. You should study these methods and be able to explain what these methods do –

1. Load valid English words from a given dictionary file (see: `loadDictionary`)
2. Determine if a given word is a valid English word (see: `isValidWord`)
3. A simple method to change punctuations and special characters to spaces to ease extracting/processing words in a line of data (see: `changePunct`)

Program Inputs

The program must use command-line arguments for obtaining inputs. The command line arguments will be in the following format:

1. The first command-line argument will specify the number of threads to use. For base case requirements, the number of threads will be 1. For other cases the number of threads will vary, but will always be lower than the number of URLs
2. Rest of the command-line arguments will be 1-or-more files to retrieve. Each file need to be added to a base URL `http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=`. For example, given a file `ex3.html`, the full URL will be: `http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html`.

Program Outputs

The result of processing URLs should be displayed in exactly the same order in which the URLs were specified. For each URL, the program should generate 1 line of output, in the following format: `<URL>,☐words:☐<#Words>,☐English☐words:☐<#EngWords>`, where ☐ is 1 blank space. Strings in blue are literal constants. See sample outputs for example.

Sample inputs and outputs

The command typed in is shown in bold. The commands and outputs are long because of URLs and have been wrapped in this document; however, they are on one single line. Video on setting command-line arguments in `NetBeans` is available on Canvas.

Base cases [12 points]

```
$ ./homework6 1 ex3.html
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html, words: 39,
English words: 16
```

```
./homework6 1 ex3.html cpp.txt
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html, words: 39,
English words: 16
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=cpp.txt, words: 6983,
English words: 5810
```

```
$ ./homework6 1 ex3.html cpp.txt us_constitution.txt miami_university.txt
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html, words: 39,
English words: 16
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=cpp.txt, words: 6983,
English words: 5810
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=us_constitution.txt,
words: 7667, English words: 7422
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=miami_university.txt,
words: 32331, English words: 15348
```

! The above 3 base case is relatively straightforward. Consequently, if the base case does not operate as expected, as per the course policy, the program will be assigned zero score.

Multi threading tests (change number of threads, but exact same outputs) [18 points]

```
$ ./homework6 2 ex3.html cpp.txt us_constitution.txt miami_university.txt
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html, words: 39,
English words: 16
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=cpp.txt, words: 6983,
English words: 5810
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=us_constitution.txt,
words: 7667, English words: 7422
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=miami_university.txt,
words: 32331, English words: 15348
```

```
$ ./homework6 3 ex3.html cpp.txt us_constitution.txt miami_university.txt
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html, words: 39,
English words: 16
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=cpp.txt, words: 6983,
English words: 5810
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=us_constitution.txt,
words: 7667, English words: 7422
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=miami_university.txt,
words: 32331, English words: 15348
```

```
$ ./homework6 4 ex3.html cpp.txt us_constitution.txt miami_university.txt
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=ex3.html, words: 39,
English words: 16
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=cpp.txt, words: 6983,
English words: 5810
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=us_constitution.txt,
words: 7667, English words: 7422
URL: http://ceclnx01.cec.miamioh.edu/~raodm/SlowGet.cgi?file=miami_university.txt,
words: 32331, English words: 15348
```

Tips

- It is easiest to store the files specified as command-line arguments into a vector to further process them.
- Use the simple HTTP client example from lecture slides to obtain response from the server.
- Extracting words and counting valid English is straightforward using an `istringstream`. Don't over complicate this part.

- Approach this problem as a data parallel application in which a subset of URLs is assigned to a thread to process. The logic of assigning threads to URLs is almost a copy-paste from lecture slides.
- In continuation with examples from lecture slides, it would be easiest to store results as a string in another vector and then finally print the vector after all the threads have finished running.
- If you find your program running too slowly in `NetBeans` you can use the `Release` mode (instead of `Debug`)

Submit to Canvas

This homework assignment must be turned-in electronically via Canvas CODE plug-in. Ensure your C++ source files are named appropriately. Ensure your program compiles (without any warnings or style errors) successfully. Ensure you have tested operations of your program as indicated. Once you have tested your implementation, upload the following:

- The 1 C++ source file you developed for this homework

Upload the C++ source files to onto Canvas. Do not submit zip/7zip/tar/gzip files. Upload each file independently.