CSE-381: Systems 2

Homework #1: Part B

Due: Wed Sept 26 2020 before 10:00 PM

Email-based help Cutoff: 5:00 PM on Tues Sept 25 2020

Maximum Points: 25

Not including the time to read this document, you should be able to complete the actual programming in **less than 2 hours**. If it takes you more than 2-hours to complete the starter code, then you are missing important skills from prerequisite courses and you will need to invest some serious time into practicing the prerequisite skills.

Submission Instructions

This homework assignment must be turned-in electronically via Canvas CODE plug-in. Ensure your C++ source code is named *MUid_homework1*.cpp (where MUid is your Miami Unique ID, e.g., raodm_homeweork1.cpp). Ensure your program compiles successfully, without any warnings or style errors. Ensure you have documented the methods. Ensure you have tested operations of your program as indicated. Once you have tested your implementation, upload just the 1 source file onto Canvas via the CODE plug-in.

1. The 1 C++ source file developed for this part of the homework.

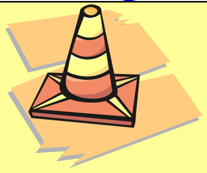
General Note: Upload each file associated with homework (or lab exercises) individually to Canvas. Do not upload archive file formats such as zip/tar/gz/7zip/rar etc.

Objective

The objective of this homework is to **extend lab exercise** and refresh your memory on the following concepts and skills from "CSE-278: Systems 1" prerequisite course:

- Working with HTTP GET request and generating HTTP response
- Basic of I/O operations in C++
- Developing C++ program involving simple string manipulation.
- Working with command-line arguments.
- Practice developing properly styled programs
- Gain experience with testing via web-browser

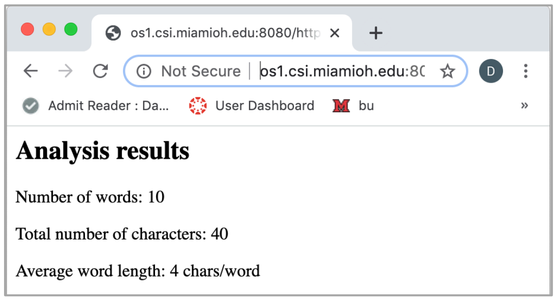
Grading Rubric:



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 or just skeleton code will be assigned zero score! Program that do not compile, **have even 1 method longer than 25 lines**, or just some skeleton code will be assigned zero score.

- **-1 Points**: for each warning generated by the compiler.
- **NOTE:** Violating CSE programming style guidelines is a compiler error! Your program should not have any style violations reported in `NetBeans` when you compile it.
- **Do not use global variables** – that is not good programming practice.
- **Base case:** 20 points: Since this homework is just a recap of lab exercise on prerequisite concepts you should be able to complete the 3 straightforward methods in this homework.
- **Formatting & Documentation:** 5 points (if your comments are poor, then you lose points int his category)

Homework overview



Analysis results

Number of words: 10

Total number of characters: 40

Average word length: 4 chars/word

In this homework you will be completing a web-server that can perform simple Natural Language Processing (NLP) of counting words and computing average word length. A screenshot of the program operating via a web-browser is shown in the adjacent figure. Specifically, your program must operate as follows:

1. Your program will operate as a web-server based on URLs of the form:
   a. `http://os1.csi.miamioh.edu:8080/http://www.users.miamioh.edu/raodm/ones.txt`
   b. `http://os1.csi.miamioh.edu:4432/http://ceclnx01.cec.mimaioh.edu/~raodm/miami_university.txt`
2. Read a HTTP-GET request with URL in the format `/http://Host:Port/Path` (e.g., `/http://os1.csi.miamioh.edu:8080/~raodm/words.txt`).
3. Obtain the specified file from the remote web-server using a suitable HTTP-GET request. Here, your web-server also acts a web-client. This type of operation is called proxying. It is commonly used in Tor (https://en.wikipedia.org/wiki/Tor_(anonymity_network)) and other anonymizing services, which are important concepts & tools in cybersecurity.
4. Process the response from the remote web-server and calculate: word count, total number of characters, and average word length. These are super-simple operations. So, don't overcomplicate them.
5. The output is generated in HTML format and returned as an HTTP response. See sample output below.

Starter Code

Several aspects of this homework have been covered in the previous lab exercise(s). Hence, ensure you review previous exercise. The starter code with skeleton methods are supplied. Ensure you `scp` all of the starter files to your NetBeans project. Your task is to:

1. Understand operations of the methods in the starter code. There are plenty of comments to help you understand the starter code. When in doubt, ensure you ask clarifications about the starter code so that you really understand the work you are doing.
2. The starter code can be run as a web-server, but troubleshooting incorrect output is cumbersome with a web-browser. Note that web-browsers will not work even if 1-character is incorrect in your output.
3. Hence the starter code is designed to read input from a file and print output so that it is easier to troubleshoot. For this you will need to set command-line arguments in NetBeans. Ensure you review the video on how to set command-line arguments from the Video Demonstrations Page on Canvas. See Tips below for screenshot specific to this homework.

Testing with web-browsers

The key requirement for this homework is that your program must work correctly with a web-browser. You may test operation of your program using the following URLs, by suitably changing the port number below:

- `http://os1.csi.miamioh.edu:8080/http://www.users.miamioh.edu/raodm/ones.txt`
- `http://os1.csi.miamioh.edu:4432/http://ceclnx01.cec.mimaioh.edu/~raodm/miami_university.txt`

Sample inputs and outputs

Testing your program is best accomplished by supplying exactly the same inputs a web-browser would generate. The starter code permits inputs to be supplied from a text file specified as a command-line argument. The sample outputs below illustrate testing using command-line arguments. This is the same approach that will be used when you submit your solution via Canvas CODE plug-in.

Base case #1:

```
$ ./homework1 base_case1_inputs.txt
URL to be processed is: http://www.users.miamioh.edu/raodm/ones.txt
Processing file "/raodm/ones.txt" from "www.users.miamioh.edu":"80" ...
HTTP/1.1 200 OK
Server: localhost
Connection: Close
Content-Type: text/html
Content-Length: 182

<html>
  <body>
    <h2>Analysis results</h2>
    <p>Number of words: 10</p>
    <p>Total number of characters: 40</p>
    <p>Average word length: 4 chars/word</p>
  </body>
</html>
```

Base case #2:

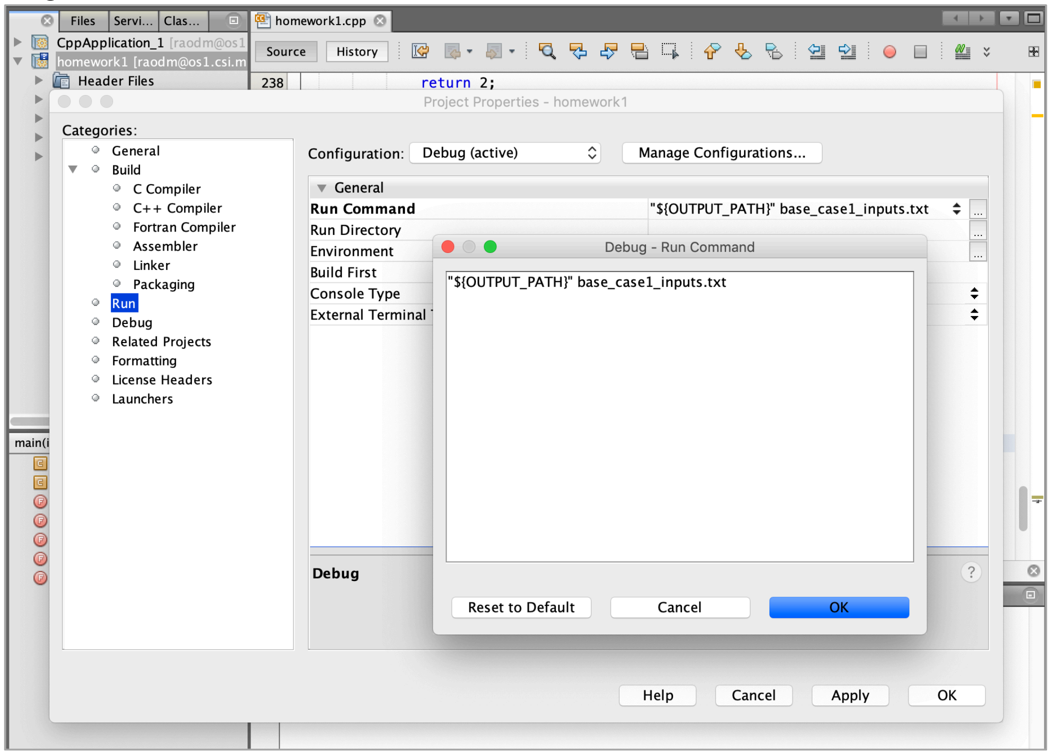
```
$ ./homework1 base_case2_inputs.txt
URL to be processed is: http://ceclnx01.cec.miamioh.edu/~raodm/miami_university.txt
Processing file "/~raodm/miami_university.txt" from "ceclnx01.cec.miamioh.edu":"80" ...
```

```
HTTP/1.1 200 OK
Server: localhost
Connection: Close
Content-Type: text/html
Content-Length: 195

<html>
  <body>
    <h2>Analysis results</h2>
    <p>Number of words: 15180</p>
    <p>Total number of characters: 188493</p>
    <p>Average word length: 12.4172 chars/word</p>
  </body>
</html>
```

Tips

1. Review and suitably use parts of code from previous exercise.
2. Review concepts of command-line arguments. Watch the video on setting command-line arguments in `NetBeans` on Canvas →`Pages`→`NetBeans video Demonstrations`. Here is a screenshot of an example of command-line argument settings for this homework in `NetBeans`:



Configuration: Debug (active)

Run Command: "${OUTPUT_PATH}" base_case1_inputs.txt

3. Rest assured that I/O is trivial. See example in C++ review video Part 1: Introduction to data types, conditionals, loops, and console I/O (PDF is in: Lecture Materials →Cpp_Review→Part1_Cpp-Basics.pdf)
4. First write skeleton code with plenty of comments to help you work out the solution.
5. **Save and compile often – you should type no more than 3-or-4 lines before saving and compiling.**
6. For troubleshooting use the debugger. There is video demonstrating the use of the debugger in NetBeans Canvas →`Pages`→`Video Demonstrations`. Ensure you review the video on how to use the debugger.

Turn-in:

This homework assignment must be turned-in electronically via Canvas CODE plug-in. Ensure your C++ source code is named `MUid_homework1.cpp` (where `MUid` is your Miami Unique ID). Ensure you have tested operations of your program with a web-browser. Once you have tested your implementation, upload just your source code to Canvas.