**Introduction to Systems Programming (System I)**

**Exercise #4**

|  |
| --- |
| **You should save/rename this document using the naming convention MUid.docx (example: raodm.docx).**  **Objective**: The objective of this exercise is to:   1. Gain some familiarity with command-line arguments 2. Review reading & writing data with text files 3. Complete a program that generates HTML formatted output   Fill in answers to all of the questions. For some of the questions you can simply copy-paste appropriate text from the shell/output window into this document. You may discuss the questions with your instructor. |

|  |  |
| --- | --- |
| **Name:** |  |

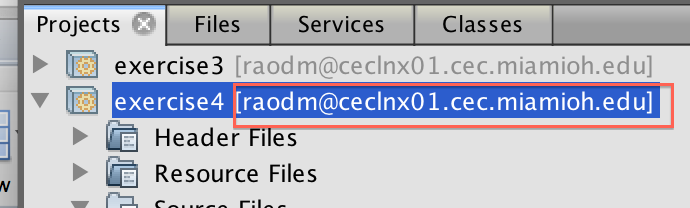
# Review copying files using scp

*Estimated time: 15 minutes*

|  |  |
| --- | --- |
|  | This part of the exercise just expects you to setup a NetBeans project and copy a couple of text files to your NetBeans project. If this part of the exercise takes you more than 10 – 12 minutes, then you need more practice! |

**Exercise**: Create a new NetBeans project and copy files to the project:

1. Using NetBeans, create a remote C++ project titled exercise4 on database.csi.miamiOH.edu.



* 1. Double check to ensure that the project is really on the server by ensuring that the server name is shown adjacent to the project name (see adjacent screenshot).

1. Download the starter code (exercise4.cpp) supplied for this exercise and copy-paste it to overwrite the template code created for you by NetBeans.
2. Note the directory where NetBeans created your remote project using your project's properties (NetBeans  Right-click-on-project  Properties  General) and
   1. Download and scp (from local terminal) lah.txt to your project directory
   2. Download and scp (from local terminal) ex4\_inputs.txt to your project directory.
   3. Download and scp (from local terminal) ex4\_expected\_output.txt to your project directory.

# File I/O using file streams

*Estimated time: 30 minutes*

We are going to review the concept I/O redirection of standard input and output streams at the shell prompt using < (less than) and > (greater than) operations as shown below:

|  |
| --- |
| $ ./exercise3 < inputs.txt > output.txt |

**Note:** As indicated in *ex-lab03.docx,* depending on the settings of NetBeans, the executable

may be placed in *./dist/Debug/GNU-Linux*  or similar directory (under ./dist). In the examples

below we assume the executable have been copied to the NetBeans project directory,

Else you can make a soft link:

**ln -s *./dist/Debug/GNU-Linux/exercise3 .***

**Background:**This exercise requires converting the solution from previous exercise (that relied on I/O redirection to read/write files) to explicitly read and write files that are:

1. Path to input file is specified as first command-line argument (i.e., argv[1])
2. Path to output file is specified as first command-line argument (i.e., argv[2])
3. Use std::ifstream and std::ofstream to read and write text file(s)

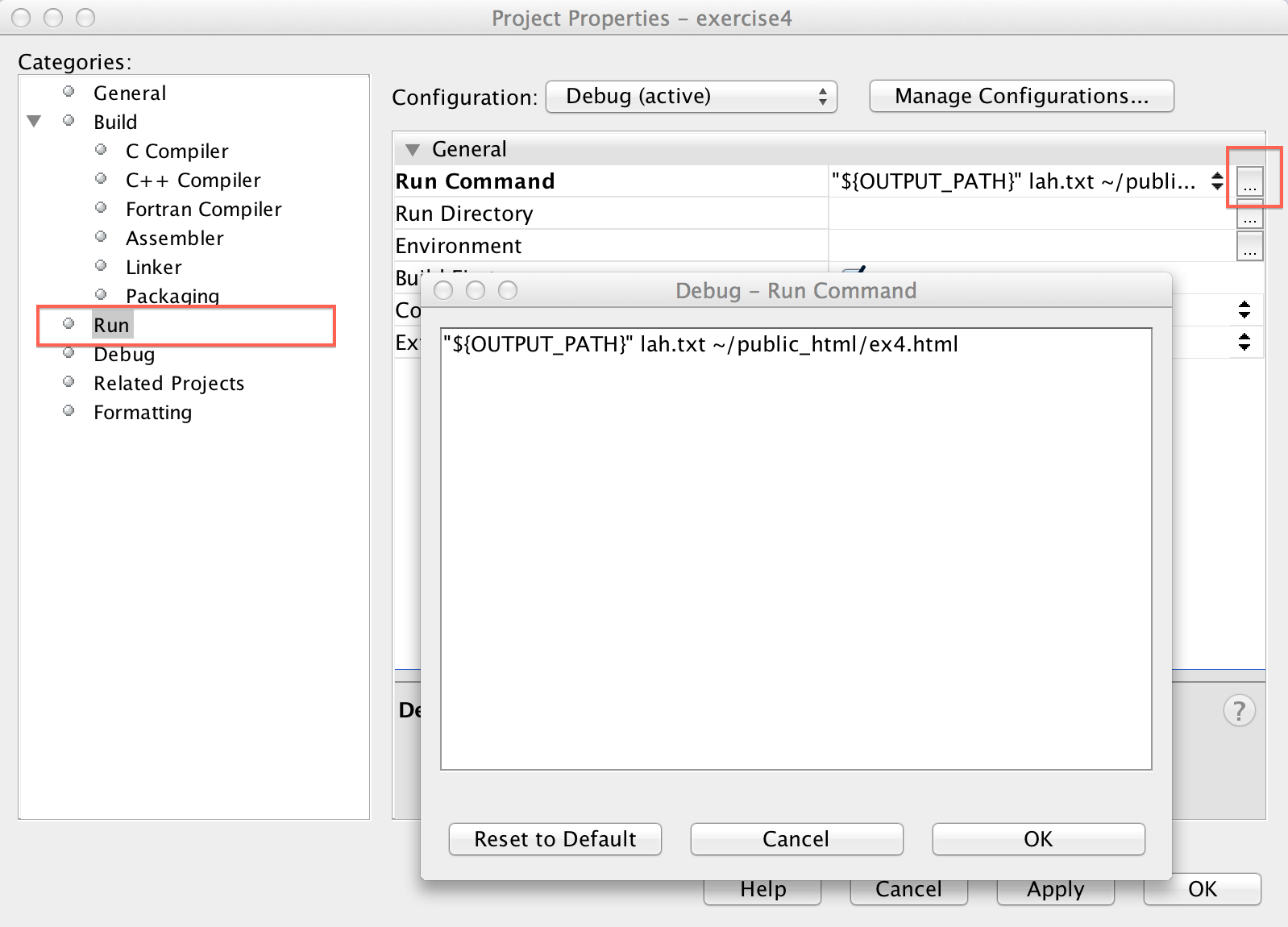
|  |  |
| --- | --- |
|  | When following the procedure below, you should be periodically saving and compiling your source code to ensure it compiles successfully! In general, when coding you should save and compile your C++ program after every 4/5 lines of change. |

**Procedure**: Convert the supplied starter code using the following steps:

1. Add the necessary #include <fstream>
2. Add the necessary variables to main function to enable use of command-line arguments
3. Add a if-check to ensure that the user has specified exactly 2 command-line arguments (i.e., argc == 3).
   1. If not, report the following error: "Specify input & output files" and return 1 (recollect that non-zero return value suggests an error) from main.
4. Open input and output file streams using command-line arguments argv[1] and argv[2] respectively.
   1. Ensure the streams were created successfully using the good method.
   2. If the streams are not good, report an error "Error opening input or output streams." and return from main with value of 2.
5. Suitably modify use of std::cin with your input file stream object.
6. Suitably modify use of std::cout with your out file stream object.

**Basic Testing:**

Once you have completed the development and successfully compiled your program, the next step is to ensure it is operating correctly. You can supply command-line arguments in NetBeans via: Project's properties  Run  Run command (click on … button) and just type in the command-line arguments as shown in the screenshot below:



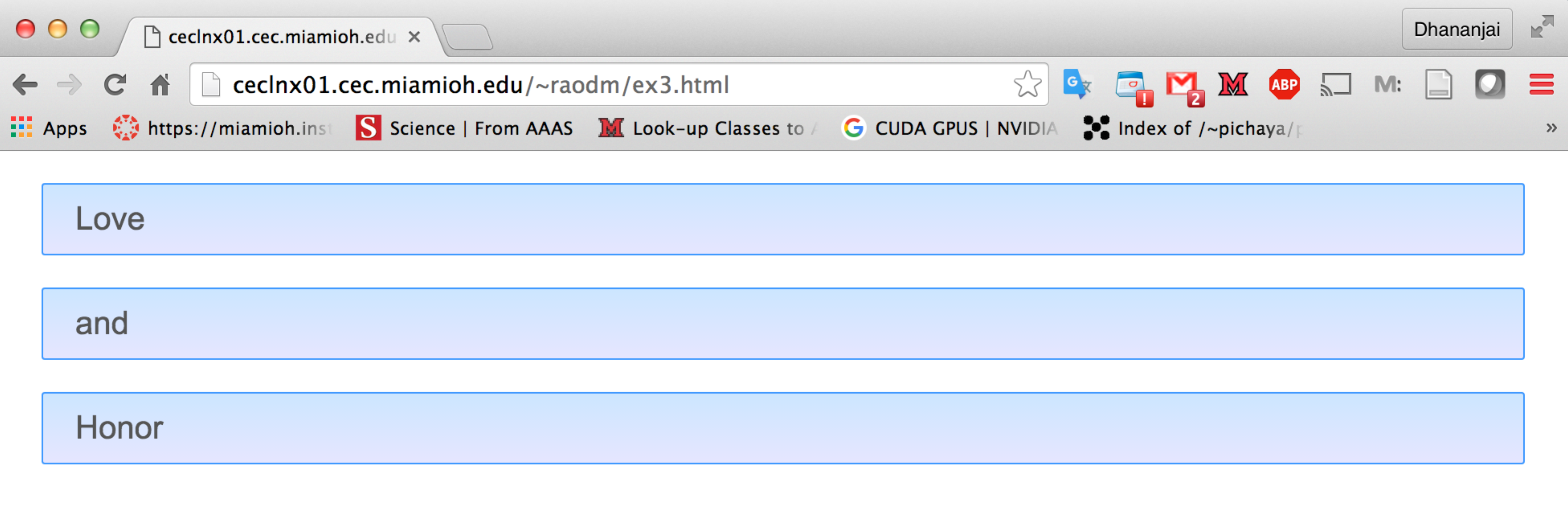
Run the program in NetBeans. If the program runs successfully it will not generate any output.

The output is an HTML file that you may upload to the ceclnx01 server in the ~/public\_html

directory and display as before:

|  |
| --- |
| http://ceclnx01.cec.miamioh.edu/~MUid/ex4.html |

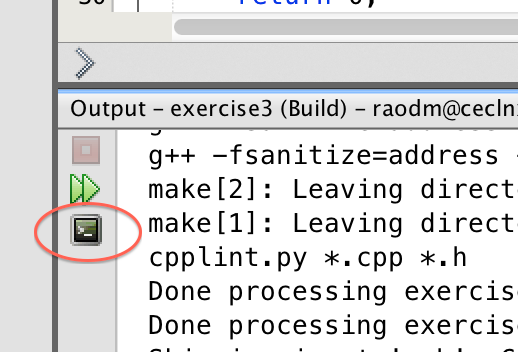
The web-page should appear similar to the screenshot shown below:



# Functional Testing

*Estimated time: 10 minutes*

**Background**: Functional testing is the process of checking to ensure that a program is operating correctly – i.e., generates the expected output for a given input. Functional testing is accomplished by comparing the output from a program against a known or expected output.

**Exercise**: In this part of the exercise you will be verifying that your program is operating correctly in the following manner:

1. Open a **Terminal in NetBeans** by clicking the terminal icon in the output window as shown in the adjacent screenshot. Note that this terminal (in NetBeans) is the same as the Mac terminal or using PuTTY in Windows. It uses SSH to connect to ceclnx01.
2. Run your program with ex4\_inputs.txt as input file and ex4\_my\_output.txt as output file:

|  |
| --- |
| $ ./exercise4 ex4\_inputs.txt ex4\_my\_output.html |

1. Use the standard Linux diff command to compare your output against the expected output:

|  |
| --- |
| $ diff ex4\_my\_output.html ex4\_expected\_output.html |

If your program is operating correctly, then the above diff command will generate zero differences – i.e., it will produce absolutely no output

1. Copy-paste a screenshot of the NetBeans terminal showing the above 2 commands in the space below:

|  |
| --- |
|  |

# Submit to Canvas

Once you successfully completed the aforementioned exercises upload the following files to Canvas.

* 1. This MS-Word document (duly filled-in) saved as a PDF document.
  2. The C++ source file of the program modified by you.

Ensure you actually **submit** the files after uploading them to Canvas.