**Heterogeneous and Cloud Computing**

**Exercise #1**

Max Points: 10

**Note: If you are using your personal machine then prior to commencing work on this exercise, you may need to install XMing, Putty, and WinScp as illustrated in LinuxEnvironment.pdf (and shown in the videos in the Handouts folder).**

|  |
| --- |
| **You should save and rename this document using the naming convention MUid.docx (example: raodm.docx).**  **Objective**: The objective of this exercise is to gain some familiarity with developing C/C++ programs on a Linux machine. This exercise will also serve to refresh the basic C++ programming that you have learnt from earlier courses.  **Submission**: Once you have completed this exercise, upload this MS-Word document (duly filled with the necessary information) and the three C++ program that you developed with the following naming convention:   * ***MUid*\_ex1\_1.cpp**: A C++ program to perform simple arithmetic and trigonometric operations. * ***MUid*\_ex1\_2.cpp**: A C++ program to perform simple string operations.   Fill in answers to all of the questions as directed. For some of the questions that require outputs to be indicated, you can simply copy-paste appropriate text from the shell/PuTTY window into this document. You may discuss the questions with your instructor. |

|  |  |
| --- | --- |
| **Name:** | **Weiguo Xia** |

# Preliminaries

You may use any editor or IDE that you are familiar with for programming. However, it is highly recommended you get comfortable using emacs. It is a powerful and rich environment and a good compromise between various tools, particularly when running over a network connection.

# Configuring Niihka Forums

In this course we will be using Niihka forums for soliciting help and conducting discussions. In order to ensure you get timely copies of postings and discussions, it is highly encouraged you receive email notifications upon postings. Consequently, enable email notifications in the following manner:

1. Log into Niihka and select the Niihka site corresponding to this course.
2. Click on the Forums tab (typically to the right hand side) to get to the forums settings as shown in the screenshot below:

|  |
| --- |
|  |

1. Next click on the Watch button (highlighted in the screenshot above) and select the radio button for “Send me an email whenever a new message is posted”
2. Next click on the Save button.

# Optionally copy my emacs configuration

To provide consistent formatting and use of emacs, copy my emacs configuration into your home directory using the following commands:

|  |
| --- |
| $ cd  $ wget -q http://pc2lab.cec.miamiOH.edu/documents/.emacs |

# Compiling C++ Programs:

Initially we will be developing straightforward C++ programs that don’t utilize parallel and distributed computing libraries. For such programs, the GNU C++ compiler g++ can be directly used. The general syntax for compiling a C++ program with g++ is:

|  |
| --- |
| **$** g++ -g –Wall –std=c++11 *source\_cpp\_files* -o *executable\_name* |

In the above command-line,

* *source\_cpp\_files* is a space separated list of C++ source files to be compiled together to form the final executable. Typically C++ source files have a “.cpp” file name extension.
* *executable\_name* is the file name to which the binary executable is to be written. In Linux, executable files do not have an extension.
* The –g option tells g++ to include debugging information into the generated executable
* The –Wall (read as dash-double-you-all) tells g++ to report all warnings.

Here is an example command-line to compile a C++ source file named raodm\_ex1\_1.cpp to generate an executable file named raodm\_ex1\_1:

|  |
| --- |
| **$** g++ -g –Wall –std=c++11 raodm\_ex1\_1.cpp –o raodm\_ex1\_1 |

You may run the generated executable (from the appropriate directory in which the executable is present) by simply typing the executable name. For example, the following command can be used to run an executable named raodm\_ex1\_1:

|  |
| --- |
| **$** ./raodm\_ex1\_1 |

# Part #1: Simple numeric operations in C++

|  |  |
| --- | --- |
| C:\Users\Victoria\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\MARLET97\MP900314351[1].jpg | Ensure that the C++ source for this exercise is named with the convention *MUid*\_ex2\_1.cpp (example: raodm\_ex1\_1.cpp), where *MUid* is your Miami University unique ID. |

**Background**: Similar to almost all programming languages, C++ provides a standard set of algebraic and trigonometric functions that can be used to perform standard mathematical operations.

**Exercise**: This exercise expects you to develop a simple program that performs the following operations:

1. Read a real number (double) from the user after prompting for input (see sample output below for message details).
2. The program must display the following information about the number entered by the user (see sample output for message details):
   1. If the number is a positive number or not.
   2. Print sine, cosine, and tangent of the number by calling the std::sin, std::cos, and std::tan functions that are defined in the C++ cmath header file (#include <cmath>). See <http://en.cppreference.com/w/cpp/numeric/math> for documentation on these methods.

|  |
| --- |
| std::cout << std::sin(num); |

**Sample Output**:

User inputs in two different runs of the program are shown in red color in the sample outputs below:

|  |  |  |
| --- | --- | --- |
| $ ./raodm\_ex1\_1  Enter a number: **10**  The number you entered is : 10  Is the number positive? : Yes  The sine of the number is : -0.544021  The cosine of the number is : -0.839072  The tangent of the number is:0.648361 |  | $ ./raodm\_ex1\_1  Enter a number: **-3.142**  The number you entered is : -3.142  Is the number positive? : No  The sine of the number is : 0.000407346  The cosine of the number is : -1  The tangent of the number is:-0.000407346 |

**Actual Output:** Once you have successfully developed and tested your program, copy-paste the output from your program into the space provided below for the input number **1.572**:

|  |
| --- |
| Enter a number:1.572  The number you entered is :1.572  Is the number positive? Yes  The sine of the number is :0.999999  The cosine of the number is :-0.00120367  The tangent of the number is :-830.79 |

# Part #2: Simple string operations in C++

|  |  |
| --- | --- |
| C:\Users\Victoria\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\MARLET97\MP900314351[1].jpg | Ensure that the C++ source for this exercise is named with the convention *MUid*\_ex2\_2.cpp (example: raodm\_ex2\_2.cpp), where *MUid* is your Miami University unique ID. |

**Background**: Similar to almost all programming languages, C++ provides a standard set of string functions that can be used to perform standard string operations.

**Exercise**: This exercise expects you to develop a simple program that performs the following string:

1. Read a line of input (using std::getline method) from the user after prompting for input (see sample output below for message details).
2. The program must display the following information about the line entered by the user (see sample output for message details):
   1. Number of characters in the input.
   2. The first and last character in the string.
   3. The first and last word in the string. You may assume that there is only one blank space between words and there are no leading or trailing blanks. (use substr, find, and rfind methods defined on std::string)

**Sample Output**:

User inputs in two different runs of the program are shown in red color in the sample outputs below:

|  |  |  |
| --- | --- | --- |
| [raodm@mualhpcp01 exercise2]$ ./raodm\_ex1\_2  Enter a line: **i will go to spring ice**  Length of line : 23  First character: 'i'  Last character : 'i'  First word : 'i'  Last word : 'ice' |  | [raodm@mualhpcp01 exercise2]$ ./raodm\_ex1\_2  Enter a line: **checking 1 two 3**  Length of line : 16  First character: 'c'  Last character : 'c'  First word : 'checking'  Last word : '3' |

**Actual Output:** Once you have successfully developed and tested your program, copy-paste the output from your program into the space provided below for the input string “oneWord”:

|  |
| --- |
| Enter a line:oneWord  Length of line:7  First character: 'o'  Lasr character: 'd'  First word: oneWord  First word: oneWord |

# Part #3: Fixing stylistic issues

**Background**: The C/C++ programs developed in this course must be developed to adhere to modern conventions and stylistic guidelines. In this course we will adopt the stylistic guidelines from Google Inc. (<http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml>):

**Exercise**: This exercise expects you to fix the stylistic issues in a given C++ program in the following manner:

1. Download the slightly updated version of lint script off Niihka from Resources 🡪 Handouts and Tutorials folder.
2. Download the supplied C++ program vector\_test.cpp
3. Run the C++ link script on the supplied program as shown below:

|  |
| --- |
| $ chmod +x cpplint.py  $ ./cpplint.py vector\_test.cpp |

1. Fix all the errors reported by lint. Needless to add, ensure that the C++ program compiles correctly without any warnings after you address the style issues.
2. Now, run the C++ lint script on your C++ source files you developed in the previous parts and ensure you do not have any stylistic violations.

# Part #4: Submit files to Niihka

Upload the following four files to Nihhka:

1. Upload this MS-Word document (duly filled with the necessary information) using the naming convention MUid.docx.

* ***MUid*\_ex1\_1.cpp**: C++ program to perform simple arithmetic and trigonometric operations.
* ***MUid*\_ex1\_2.cpp**: A C++ program to perform simple string operations.