ENSF 614- Winter 2023

LAB 1

Instructor: Anja Slama (anja.slama@ucalgary.ca)

Department of Electrical & Computer Engineering

University of Calgary

Due Date:

Your lab reports must be **submitted electronically on the D2L,** into the **Lab 1 Dropbox**, **11:59 PM on Sunday, January 22, 2023**. All your work should be in a **single PDF**.

Objectives:

This lab assignment contains a few very simple exercises that are designed to give you some experience with the process of developing, compiling, and running a simple C++ program and learning basic constructs of C++ programming such as:

- C++ operators and control structures
- Using standard input/output objects cout and cin.
- Defining a C++ program with two or more user-defined functions.

Important Notes:

1. When submitting your source code (C++ code, files with the extensions: .cpp, or .h), make sure the following information appears at the top of your file:
    a. File Name
    b. Assignment and exercise number
    c. Lab section
    d. Your name
    e. Submission Date:

Here is an example:

```
/*
File Name: lab1exe_F.cpp
Assignment: Lab 1 Exercise F
Completed by: Your Name (or your team's name for group exercises)
Submission Date: Jan 19, 2023
*/
```

2. Some lab exercises may ask you to draw a diagram, and most students prefer to hand-draw them. In this case, you need to scan your diagram with a scanner or an appropriate device and insert the scanned picture of your diagram into your PDF file. Microsoft Lens is a possible mobile app to scan your documents, which you can install on your mobile device for free. Please make sure your diagram is clear and readable. Otherwise, you may lose marks, or it could be impossible for TAs to mark it.

3. Some exercises in this lab and future labs will not be marked. Please do not skip them, as unmarked exercises are as important as other exercises.

4. Some students skip directly to the exercises that involve writing code, skipping the sections such as "Read This First", or postponing the diagram-drawing. That is a bad idea for several reasons:
    a. "Read This First" sections normally explain technical or syntax details that may help you solve the problem or provide you with some hints.
    b. Drawing diagrams is an important part of learning how to visualize memory used in computer programs. If you do diagram-drawing exercises at the last minute, you won't learn the material very well. If you do the diagrams first, you may find it easier to understand the code-writing.

Marking Schema:

You should not submit anything for the exercises that are not marked.

Exercise	Marks
A	No marks
B	8 marks
C	No marks
D	5 marks (only for part 2. Part 1 will not be marked)
E	4 marks
Total:	**17 Marks**

Exercise A: Creating a C++ source file, building and running an executable

This exercise will not be marked, and you shouldn't submit anything.

Read This First:

If this is your first attempt to develop a C++ program, please do this exercise so that you become comfortable with program development in C++, particularly using your favourite development environments such as Cygwin, Xcode, or a simple editor such as Notepad++.

What to Do:

- Startup favourite editor or IDE.
- Into the editing area, type exactly in all the following C++ code:

```
#include <iostream>
using namespace std;

int main( void ){
    int a = 0, b =0;
    cout << "Please enter a value for variable a:\n";
    cin >> a;
    cout << "Please enter a value for variable b:" << endl;
    cin >> b;
    cout << "The values of a and b are: " << a  << " for a, and " << b << " for b.\n";
    cout << "The value of " <<  a << " % " << b << " is " << a % b << endl;
    return 0;
}
```

- Now save your file as: `lab1_exeA.cpp`
- If you are using a text editor and Cygwin on your Windows OS, within Cygwin terminal navigate to your working directory (the same directory that you saved your `lab1exe_A.cpp`, then on the Cygwin command line enter:

```
g++ -Wall lab1_exeA.cpp
```

An executable file called `a.exe` will be created. If the command fails -- which will be indicated by one or more error Messages--go back to your editor and fix the code, save the file again and try `g++` again.

- If you are using Mac OS and Xcode IDE, press the run button to compile and run your program. Also, Mac users can use the Mac terminal and follow the same steps mentioned above for Cygwin to use the g++ command and compile the program. However, the executable file `a.exe` file will be called `a.out`.

- Once you have an executable, run it a few times by using the command `./a.exe (or ./a.out on the terminal of Mac machine)` over and over.

Try different inputs each time; see what happens if you enter letters or punctuation instead of numbers.

Hint: You don't need to type '`./a.exe` or `./a.out` over and over! You can use the up arrow on your keyboard to retrieve previously entered commands.

Exercise B: A C++ Program with User-Defined Functions (8 marks)

Read This First

In physics, assuming a flat Earth and no air resistance, a projectile launched with specific initial conditions will have a predictable range (maximum distance) and a predictable travel time.

The range or maximum horizontal distance travelled by the projectile can be approximately calculated by:

$$d = \frac{v^2}{g}\sin(2\theta)$$

Where: *g is gravitation acceleration (*9.81 m/s^2)
θ: the angle at which the projectile is launched in degrees
v: the velocity at which the projectile is launched d: the total horizontal distance travelled by the projectile

To calculate the projectile travel time (t) when it reaches the maximum horizontal distance, the following formula can be used:

$$t = \frac{2v\sin\theta}{g}$$

In this exercise, you will complete a given C++ source file called `lab1_exeB.cpp` that prompts the user to enter a projectile's initial launch velocity (v) and displays the table of maximum horizontal distance and travel time for the trajectory angles of 0 to 90 degrees.

What to Do:

First, download the file `lab1_exeB.cpp` from D2L. This file contains the definition of function main and prototypes for four other functions. Your job is to complete the definition of missing functions as follows:

- **Function create_table:** which is called by the main function, receives the projectile's initial velocity and displays a table of the projectile's maximum travel distance (d) and time (t) for trajectory angles of 0 to 90 (degrees), with increments of 5 degrees. Here is a sample of the required table:

```
Angle          t            d
  (deg)        (sec)         (m)
  0.000000     0.000000     0.000000
  5.000000     1.778689     177.192018
  10.000000    3.543840     349.000146
```

You don't have to worry about the format or the number of digits after the decimal point. The default format is acceptable.

- **Function projectile_travel_time:** receives two double arguments, the trajectory angle (θ), and the initial velocity (θ) and returns projectile travel time (t).

- **Function projectile_travel_distance:** receives two double arguments, the trajectory angle (θ), and the initial velocity (θ) and returns the projectile maximum horizontal distance (d).

- **Function degree_to_radian:** receives an angle in degrees and converts to radian. This function is needed, because C++ library function `sin` needs its argument value to be in radian.

Notes

- To use library function `sin`, you need to include header file `cmath`.
- Please pay attention to constant values of π, and gravitation acceleration, `g`. The following lines are already included in the given file:

```
const double PI 3.141592654 const
double G 9.8
```

- While running your program, try a few times to enter a negative value or invalid input (for example, instead of a number, enter xyz), for velocity, and observe how the program reacts.

What to Submit:

Submit a copy of your program (your code and the program output) as part of your lab report in PDF format. You don't need to upload your actual source code. Only it must be copied and pasted into your lab report along with the program's output.

Exercise C – Introduction to Pointers

This exercise will not be marked, and you shouldn't submit anything.

Read This First

This is an important exercise in ENSF 614. If you don't become comfortable with pointers, you will not be able to work with C++. So, spend as much time on this exercise as necessary to understand what is happening at every step in the program.

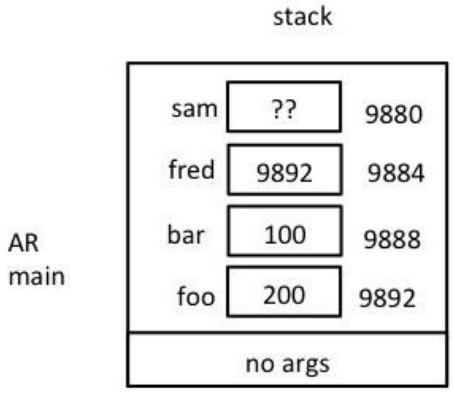
What to do

Download the file `lab1_exeC.cpp`. Trace through the execution of the program to determine what the output will be.

Now assume the following addresses for variables:

```
sam      9880
fred     9884
bar      9888
foo    9892
```

Draw a set of AR diagrams for points two through five and **use the given address numbers as values of the pointers (don't use arrow notation in this exercise)**. To understand how to draw these diagrams the solution for point one is given in the following figure:



After you complete the exercise, compile lab1_exeC.cpp and check your predicted output against the actual program output.

Exercise D: Pointers as function arguments

What to do – Part I

First, copy the file `lab1_exeD1.cpp` from D2L. Then, carefully make diagrams for point one in this program, using "**Arrow Notation**" as discussed during lectures (you don't need to use made-up addresses, as we did in the previous exercise). Then compare your solution with the posted solution on the D2L.

There is nothing to submit for this part.

What to do – Part II

Now download the file `lab1_exeD2.cpp` from D2L and draw AR diagram for point one in this file.

Submit the AR diagram for part II as part of your lab report.

Exercise E: Using pointers to get a function to change variables

Read This First

Here is an important note about terminology:

- *Don't* say, ``Pointers can be used to make a function return more than one value.'' A function can never have more than one return value. A return value is transmitted by a return statement, not by a pointer.
- *Do* say, ``Pointer arguments can be used to simulate call by reference,'' or ``Functions with pointer arguments can have the side effect of modifying the variables that the pointer arguments point to.''

What To Do

Make a copy of the file `lab1_exeE.cpp` from D2L. If you try to compile and run this program, it will give you some warning and display meaningless output because the definition of function `time_convert` is missing.

Write the function definition for `time_convert` and add code to the main function to call `time_convert` before it prints answers.

Submit a copy of your source code and the screenshots of the program output as part of your lab report.

ACKNOWLEDGEMENTS

This lab is based on the documents made available by *M. Moussavi, PhD, PEng,* for teaching ENSF614.