

# LAB 1

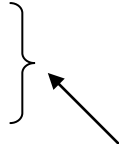
**BACKGROUND:** Handout DEMO1.C++ and Class discussions

**PURPOSE:** Objective of this LAB is to:

- a. Write and execute a simple C++ program.
- b. Get acquainted with the Integrated Development Environment (IDE)
- c. Familiarization with C++ screen output functions.

**Assignment:** Implement the sample code provided. Once you get your sample project to operate, ***modify the program*** to **print to the screen some information about yourself**. Use some imagination in text formatting. Have fun with it. This exercise will provide practice using common C++ output functions such as:

cout()      cin()      }



**Look up these commands in the C++ compiler Help Feature.**

**Note:** Take advantage of the Help utility in the IDE Edit window. Remember to comment your code. Industry demands that certain common coding practices are maintained. As time goes on, you will be required to adhere to a course-coding standard intended to form good coding practices.

**For your information.** In industry, there is much consideration to the structure of a program. In the case of a “Real-Time” application, there is the issue of uninterrupted collection and/or display of data. Programs can be written in a manner that causes all processing to stop while waiting for an event such as a keystroke. Programs can also be written in a manner that allows programs to continue processing data even though one process is waiting for an action. Identify where the sample program may cause all processing to stop.

Introduction:

Begin your program with the program header. This will represent the coding standard for all assignments.

After you write your initial program, use the IDE Editor to make changes to your file LAB1.CPP and then Compile/Link/Run it again. If you make changes to your file or correct “bugs” (and there will be), list these changes in the revision history section of your program header. It is professional and is good programming practice.

**Submit a folder** containing your diskette with the files **LAB1.CPP** and **LAB1.EXE** along with a **printout** of your LAB1.CPP file. All programs must be turned on the due date **before** the start of the lecture. You may also turn work in on-line

# HANDOUT 1

This is the standard comment block to be used for all lab assignments as part of the COMSC9A coding standard. (No Exceptions)

**No space** between comment block and preprocessor directives

## Programming Tip:

Indent your code by one level (3 spaces) within the opening and closing braces. This emphasizes the functional structure of the code. Proper indentation helps to clarify the purpose of a section of code and makes debugging easier.

```
//*****  
//      Copyright © 2012, "Your Company Name"      //  
//      All rights reserved.                        //  
//                                                    //  
// Author:                                           //  
//      "Your Name"                                //  
//                                                    //  
// Department:                                       //  
//      COMSC9A C++ Programming Department          //  
//                                                    //  
// Purpose:                                          //  
//      This program creates a Yuba College demo screen //  
//                                                    //  
// Effects:                                          //  
//      The expected usage is:                      //  
//      1. Introduce selected C++ commands          //  
//      2. Serve as a baseline for Lab Assignment #1 //  
//      3. Introduce the existence of coding standards //  
//                                                    //  
// Revision History                                //  
//      MM/DD/YYYY "Your Name"                     //  
//      - Original Version                          //  
//                                                    //  
//*****  
#include <iostream>  
using namespace std;  
  
int main(void)  
{  
  
    // The following code segment sets writes the user defined    //  
    // output.  
  
    cout << "Welcome to" << endl << endl;  
    cout << "YUBA COLLEGE" << endl << endl;  
    return 0;  
}
```

Place a **single space** between the preprocessor directives and first line of code.