Laboratory Exercise 1

You may obtain a (free) copy of MS Visual C++ Compiler for use on your home computer.

Please use this tutorial to acquaint yourself with the MS Visual C++ compiler.

To get started:

1. Log in
   Enter your Username – *Login*
   From the bottom row of buttons select **gnome -> Windows XP**
   (Only now) Enter your Password – *Login*
   Create a folder (preferably on a memory stick) called “Lab1”

2. Enter the IDE
   Select **Visual Studio 2010**
   File -> New -> Project
   Visual C++ -> Win32 -> Win32 Console Application
   Enter **Name** as “tut1”
   Select **Location** as your Lab1 folder
   Select **Solution name** as “tut1”

3. Enter your program (Note that key-words are automatically **bold)**.

4. Compile, link and execute your program.
   Select **Debug -> Start Debugging**
   If there are any errors, correct them and repeat

It is recommended that you use a USB memory device to transfer files between home and University.

1. Type and save this code through the IDE. Then compile it and execute it.

```
// Tutorial 1 - A simple program which utilises
// <iostream>.
// We will discuss this in more detail next week.

#include "stdafx.h"
#include <iostream>
using namespace std;

int main(int argc, char *argv[])
{
   char name[50];
   cout << "\n**********************\n";
   cout << "Hello world!" << endl;
   cout << "Please enter your name: ";
   cin.getline(name, 50);
   cout << "\n**********************" << endl;
   cout << endl;
   cout << "Hello " << name << " ! " << endl;
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

2. Modify the program so that it prints Hello World! three times.
   Do this with a for loop and then repeat the enhancement with the use of a while loop.
3. Enhance the program so that it displays a menu of options to perform different functions.
   Use a switch statement to evaluate which functions should be executed.

These <iostream> functions may prove useful.

cout << *string* sends strings and numeric values to standard output

cout << endl sends a newline character to standard output and flushes the output buffer

cin >> *string* accepts strings and values from standard input, ignoring leading whitespace, and terminating when a whitespace character is encountered .
(whitespace: any number of spaces, tabs or newlines)

cin.get(ch) extracts a character from the input stream into a character variable.

You may still use <stdio.h> but it is also worthwhile learning <iostream.h> capabilities.