ELEC230 – Laboratory 3 – Basic C++ Examples

Introduction

Today's laboratory is to get you familiar with the programming environment on your Raspberry Pi Zero W using MobaXterm to connect to your Pi over the USB RNDIS/Gadget interface, and allow you to use the Geany IDE to edit, compile and execute programmes in a Unix/Linux environment.

If you haven't yet finished configuring your Pi, please continue with the Lab 2 instructions before starting the examples here.

All students: you were unable to configure the WiFi to connect to Eduroam last week due to some missing instructions. Please now follow the instructions entitled 'Pi-Zero Eduroam configuration' on Canvas before continuing. Then, check the Lab 2 script and complete tasks you were unable to do last week, before continuing.

Examples

These examples are to help you embed and extend what has started to be introduced in lectures, and begin to prepare you for your first assignment. You need to access the "std_lib_facilities.h" headerfile that you'll find on Canvas. It is probably easiest, at this stage, to just open the file in Notepad++ on your Windows PC and then copy the text to a file you have created in Geany. Try and save your files in a structured way on your Raspberry Pi, for example this week you could create a directory "/home/pi/CPPLabs/Week3" and store the programs under there.

Also **note that there are some deliberate errors in many of these examples** so that even if you cut and paste them from this document you will need to correct the errors before the program will run correctly. This should improve your understanding of the C++ syntax[1]. Other sections just contain code fragments that you should include in a program you develop by yourself.

Thirdly, notice how, in the programs below, we'll use the input and output streams **cin** and **cout**, unlike printf etc. which you are used to in C. These (cin and cout) are not just functions; they are called "objects" in C++. They are, in fact, objects of the classes **istream** and **ostream** respectively (these classes are part of the **iostream** library; in the examples below, we are using std_lib_facilities.h instead.) A class is a bit like a user-defined data structure – a 'blueprint' for objects. An object of a particular class is a bunch of data suitable to that class, along with some methods relevant to the data, derived from the class. For example, other objects belonging to the istream class (which deals with streams of input comprising characters, strings or other objects) are get, getline, read, and so on...

Program – Hello world

In Step 14 of the Week 2 lab script, you may have noticed that there is no direct template for a .cpp file in Geany, despite the instruction given. So it would have been better to create a blank new file and add the necessary headers and main function yourself (the "boilerplate" code, if you will) to get the program to work, as shown below.

```
// a first program:
#include <iostream>                 // get standard io streams
int main()                          // main() is where a C++ program starts
{
      cout << "Hello, world!\n"; // output the 13 characters Hello, world!
                                    // followed by a new line
      return 0;                     // return a value indicating success
}
```

[1] 'Syntax' means the way that statements must be structured in a given programming language.

From this point forwards you will need to go back and follow the instructions on Page 1 about the std_lib_facilities.h header file, if you haven't done so already. Note that this file is used for convenience in early examples, to avoid you needing to learn more about header files just yet; it provides to you the most well-used standard headers (in the global namespace).

Program – Input and Output

```
// read first name:
#include "std_lib_facilities.h"
int main()
{
      cout << "Please enter your first name (followed " << "by 'enter'):\n;
      string first_name
      cin >> first_name;
      cout << "Hello, " << first_name << '\n';
}
```

Program - String Input and Concatenation

```
#include "std_lib_facilities.h"
int main()
{
      cout << "please enter your first and second names\n";
      string first;
      string second;
      cin >> first >> second;        // read two strings
      string name = first + ' ' + second; // concatenate strings
                                     // separated by a space
      cout << "Hello, "<< name << '\n';
}
```

Program – Read Integers

```
// read name and age:
int main()
{
      cout << "please enter your first name and age\n";
      string first_name       // string variable
      int age;                // integer variable
      cin >> first_name >> ages;    // read
      cout << "Hello, " << first_name << " age " << age << '\n';
}
```

Program – Simple Arithmetic

```
// do a bit of very simple arithmetic:
#include "std_lib_facilities.h"
int main()
{
      cout << "please enter a floating-point number: "; // prompt for a
                                                        // number
      double n;                                  // floating-point variable
      cin >> n;
      cout << "n == " << n
            << "\nn+1 == " << n+1             // '\n' means "a newline"
            << "\nthree times n == " << 3*n
            << "\ntwice n == " << n+n
            << "\nn squared == " << n*n
            << "half of n == " << n/2
            << "\nsquare root of n == " << sqrt(n)    // library function
            << '\n';
}
```

Program – Simple Computation

```
#include <std_lib_facilities.h>
int main()        // inch to cm conversion
{
      const double cm_per_inch = 2.54; // number of centimeters per inch
      int length = 0;          // length in inches
      while (length != 0);     // length == 0 is used to exit the program
      {                        // a compound statement (a block)
            cout << "Please enter a length in inches: ";
            cin >> length;
            cout << length << "in.  = " << cm_per_inch*length << "cm.\n";
      }
}
```

Program – Type Safety Violation

```
#include "std_lib_facilities.h"
int main()
{
      int a = 20000;
      char c = a;
      int b = c;
      if (a != b)        //  != means "not equal"
            cout << "oops!: " << a << "!=" << b << '\n'
      else
      cout << "Wow! We have large characters\n";
}
```

Program – Uninitialized Variables

```
#include "std_lib_facilities.h"
int main()
{
      int x;             // x gets a "random" initial value
      char c;            // c gets a "random" initial value
      double d;          // d gets a "random" initial value
                  // - not every bit pattern is a valid floating-point value
      double dd = d;     // potential error: some implementations
                         // can't copy invalid floating-point values
      cout << " x: " << x << " c: " << c << " d: " << d << '\n';
}
```

Program – Simple Computation II

```
#include "std_lib_facilities.h"
int main()
{
      const double cm_per_inch = 2.54;
      int val;
      char unit
      while (cin >> val >> unit) {  // keep reading
            if (unit == 'i')        // 'i' for inch
                  cout << val << "in == " << val*cm_per_inch << "cm\n";
            else if (unit == 'c')   // 'c' for cm
                  cout << val << "cm == " << val/cm_per_inch << "in\n";

            else
                  return 0;  // terminate on a "bad unit", e.g. 'q'
      }
}
```

Program - Square

Write a program that calls a function “int square(int x)” for the first 100 positive integers. The main routine should print two columns of numbers the value and its square.

Upload your program to Canvas using the link provided. Although this is not assessed, it will give you the chance to get some feedback, and us the chance to evaluate the skills of the group so far. Don’t forget that Assignment 1, based around C++, will soon be released, so this is a good chance to get some early feedback.

Program – Vectors

```
// compute mean (average) and median temperatures:

#include “std_lib_facilities.h”
int main()
{
      vector<double> temps;    // temperatures in Fahrenheit, e.g. 64.6
      double temp;
      while (cin>>temp)  temps.push_back(temp); // read and put into vector
      double sum = 0;
      for (int i = 0; i< temps.size(); ++i) sum += temps[i];
                                                  // sums temperatures
      cout << "Mean temperature: " << sum/temps.size() << '\n';
      sort(temps.begin(), temps.end());
      cout << "Median temperature: " << temps[temps.size()/2] << '\n';
}
```

Program – Word List

Note that in this example you will need to add the main function and header file(s).

```
// “boilerplate” left out
      vector<string> words;
      for (string s; cin>>s && s != "quit"; )    // && means AND
            words.push_back(s);
      sort(words.begin(),words.end());           // sort the words we read
      for (string s : words)
            cout << s << '\n';


      /*
          read a bunch of strings into a vector of strings, sort
          them into lexicographical order (alphabetical order),
          and print the strings from the vector to see what we have.
      */
```

Program – Eliminate Duplicates

Note in this example you will need to add the main function and header file(s).

```
// Eliminate the duplicate words by copying only unique words:
      vector<string> words;
      for (string s; cin>>s && s!= "quit"; )
            words.push_back(s);
      sort(words.begin(), words.end());
      vector<string>w2;
      if (0<words.size()) {               // note style { }
            w2.push_back(words[0]);
            for (int i=1; i<words.size(); ++i)    // note: not a range-for
                  if(words[i-1] != words[i])
                        w2.push_back(words[i]);
      }
      cout<< "found " << words.size()-w2.size() << " duplicates\n";
      for (string s : w2)
             cout << s << "\n";
```

Program – Out of Range

Note in this example you will need to add the main function and header file(s).

```
vector<int> v(10);      // a vector of 10 ints,
                        // each initialized to the default value, 0,
                        // referred to as v[0] .. v[9]
for (int i = 0; i<v.size(); ++i) v[i] = i;      // set values
for (int i = 0; i<=10; ++i)               // print 10 values (???)
      cout << "v[" << i << "] == " << v[i] << endl;
```

Program – Try and catch

```
int main()
{
  try
  {
    throw 20;
  }
  catch (int e)
  {
    cout << "An exception occurred. Exception Nr. " << e << '\n';
  }
  return 0;
}
```

Program – Exceptions

```
int main()
      try
      {
              // …
      }
      catch (out_of_range&) { // out_of_range exceptions
              cerr << "oops - some vector index out of range\n";
      }
      catch (…) {       // all other exceptions
              cerr << "oops - some exception\n";
      }
```

Version history

Updated:	Dr D.G. McIntosh & Dr D.G. McGuiness	October 2021
Original version:	Prof. J.S. Smith	Autumn 2019