COMP6651 – Programming Environment cheat sheet.

Introduction

This document is meant to show you some simple snippets of code that will be very useful for your labs and homework, especially for the IO part. If you are familiar to C/C++ this should be enough to do all your homework (Exercise and Problems).

However, this is not meant to be a C/C++ manual so if you are not familiar with C/C++ it, please purchase a C/C++ book or look at online resources.

The content of this document will be presented to you in detail in the first lab and in class if needed.

Programming environment

For this course, we will use a very simple environment using the Linux operating system. You use it using a command line terminal (i.e. xterm).

You can use various editors with syntax highlighting such as vi, vim, emacs, etc.

You can compile and run using the command line commands shown below.

Compiling and running

Most of your programs will consist of one file (i.e. main.cpp). You can compile it:

```
g++ –o run main.cpp
```

This will compile the program located in main.cpp and output an executable called run. You can run it by typing:

```
./run
```

For more complex problem we will use the make tool that will be explained in the class/lab. The rules of compiling your project will be encoded in a file called *Makefile* and you can compile it typing the command *make*.

Header wrappers

The header file (.h) should have the following statements:

```
#ifndef SOMEUNIQUENAME
#define SOMEUNIQUENAME


// all your code

#endif
```

This is not necessary for the cpp files.

Main function

The entry to your program is the main function that has the following signature:

```
int main(int argc, char* argv[]){}
```

argc – is the number of command line arguments, including the name of the file.
argv[i] – is the string of the ith, parameters (0 is the actual file name)

For instance, if your call is:

```
./run file1 file2
```

argc = 3, argv[0]=run, argv[1]=file1, argv[1]= file2

Numerical data types in C/C++

Most numerical typs you will use are: int, float and double. Int is integer, float and double are floating point numbers.

String data type in C/C++

You should use the std::string class. To do that you need to add the statement

```
#include<string>
```

at the top of your program.

An important detail is that the argv parameters (see section 4) uses a C style type. You can convert it as follows:

```
std::string s = argv[i];
```

Important STL data types

C++ comes standard with a number of useful data types encapsulated as Standard Tenplate Library (STL). Here are some important types:

Vector type

std::vector<type> v;
Important methods:
Adding an element at the end: v.push_back(type e);
Resizing the data: v.resize(int);
R/W access: v[index] =
Clearing the data: v.clear. (note the size of the vector will be 0)

File and console IO.

You can output message to the console (very useful for debugging) using:

```
std::cout<<" TEXT"<<number<<std::endl;

std::endl – is an end of line character
```

To use this you need to add the following statement:

```
#include <iostream>
```

To open a file for reading/writing:

```
// at the top
#include <fstream>

std::ifstream fi(name);
if(!fi){
    std::cout<<"Unable to open the file "<< name<<" for reading!"<<std::endl;
    return -1;
}

std::ofstream fo(name);
if(!fo){
    std::cout<<"Unable to open the file "<< name<<" for writing!"<<std::endl;
```

```
    return -1;
}

int i = 5;
float f = 0.1;
double d = 0.02;;
std::string name "BLAH";


// writing

fo<<i<<" "<<f<<" "<<d<<std::endl
fo<<name<<std::endl;

The resulting file will be:

5 0.1 0.01
BLAH


// when reading you need to add the spaces and/or the end of line character, oterwise when
reading the program will not be able to figure out how to separate the elements.

// reading

// you can use the >> operator for any numerical type

fi>>i;
fi>>f;
fi>>d;
fi>>name;

If the file read is the one above, then the values of your variable will be:
i=5
f=0.1
d=0.02
name=BLAH
```