ICT01

Introduction to C++ will cover some basic C++ syntax. All course materials can be found in the following website:

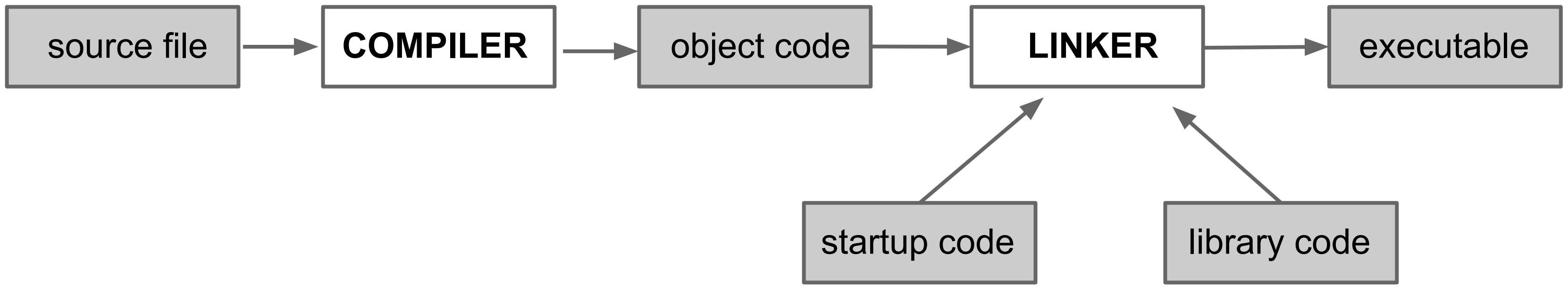
http://stanlry.github.io/ictclass

I strongly recommend you to take a look at http://www.cplusplus.com/. It contains lot of useful references and tutorials.

Introduction to C++

Creating C++ Program

There are steps that converts a text file (source code) input an executable file.



Hello World !

In the tradition, we will start with a hello world program with C++

```
#include <iostream>

int main() {
    std::cout << “Hello World” << “\n”;
    return 0;
}
```

After you type the code input text editor, you can use compiler to create an executable file. The output of the above code will be display “Hello World” in the console screen.

```
#include <iostream>                        // A PreProcessor directive

int main()                                 // Function header
{                                          // Begin of a function
    std::cout << “Hello World” << “\n”;    // Statement
    return 0;                              // Return statement
}                                          // End of a function
```

1. The #include directive tells the compiler to put the code from <iostream> into this program before compiling the program

2. The `int main()` tells the compiler there is a function named main, and the function returns an integer. The open and close brace ({ and }) indicate the begin and end of a function, which is the function body.

3. The std::cout statement tells the compiler to output something to console screen.

$$\underbrace{\texttt{std::}}_{\text{namespace}}\overbrace{\texttt{cout <<}}^{\text{output syntax}}\ \underbrace{\texttt{“Hello World”}}_{\text{a string}}\ \texttt{<<}\ \underbrace{\texttt{“\textbackslash n”}}_{\text{escape sequence}}\ \overbrace{\texttt{;}}^{\text{end of statement}}$$

a. Namespace is a feature in C++ that helps to organize program with pre-existing code (library) from different vendors. In here, it will look for the cout object in std namespace which is the standard library (std)
   There is an alternative way to declare namespace:

   ```
   using namespace std;
   cout << “Hello World”;
   ```

b. cout << is syntax that output text to console screen. We will discuss it later

c. String is a sequence of characters enclosed by quotation mark (“ “).

d. Escape sequence is characters that has special meaning in text representation. The “\n” represent a newline character.

e. Semicolon (;) is added to most end of lines in C++ to declare the end of a statement.

4. At the end of the program, a return 0 tells the operating system that it has finished successfully.

Exercise 1.1

Create your first C++ program. Print your name to the console output.

Variables

Variable is a piece of computer memory that name and type are given to store values.

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

int main() {
    int age;

    age = 12;
    cout << “You age is: ”;
    cout << age << endl;
    return 0;
}
```

The program above declare a variable named age, and assigned a value 12 to it. And use cout to display it on console screen

$$\underbrace{\texttt{int}}_{\text{type of data}}\ \overbrace{\texttt{age}}^{\text{name of the variable}};$$

To declare variable in C++, you need the syntax “<data type>   <name>”. It indicates the type of data to be stored and the name that program will use for storing data. In the above example, the program creates a variable called age which can store an integer.

Examples of variable declaration:

```cpp
int a;
int thisisaverylongvariablename;
int ___omg;

// You can declare multiple variables in single line by using comma
int a, b, c, d, e;
```

In C++ we can declare a variable with number, letters and underscore (_), but the name cannot start with number

In C++ there are some data type that are built-in, we call them primitive type. Data of different type takes different amount of memory to store.

Here are some of the primitive type in C++:

Data Type	Size (byte)	Typical Range	Description
`int`	4	-2147483648 to 2147483647	Integer
`char`	1	-127 to 127 or 0 to 255	Single character or small integer
`float`	4	+/- 3.4e +/- 38 (~7 digits)	Floating point number
`double`	8	+/- 1.7e +/- 308 (~15 digits)	Double precision floating point number

When you declare the variable like this

```
int age;
```

It does not contain any value. You can assign value to variable using assignment operator (=).

```
int age;
age = 12
```

You can also assign value to variable on its declaration.

```
int age = 12;
```

Basic Operators

Operators are symbols that operates on values or a variable. C++ has a wide range of operators:

- arithmetic operators
- assignment operators
- logical operators
- bitwise operators

  …..

In this section we will only discuss some of the operators.

Here are C++'s five basic arithmetic operators:

Operator	Description
+	adds its operands. e.g. 4 + 3
-	subtracts the second operands from the first e.g. 4 - 3
*	multiplies its operands e.g. 4 * 3
/	divides its first operand by the second e.g. 4 / 3
%	finds the modulus of its first operand with respect to the second. This produces the remainder in the division. e.g. 4 % 3 the result would be 1

Assignment Operators:

Operator	Description
=	Assign values from right to left
+=	a += b Same as a = a + b
-=	a -= b Same as a = a - b
*=	a *= b Same as a = a * b
/=	a /= b Same as a = a / b
%=	a %= b Same as a = a % b

Increment and decrement Operators:

Operator	Description
++	a++ Same as a = a + 1
--	a-- Same as a = a -1

Exercise 1.2

Write a program to calculate the value of 2 to the power of 20 divided by 13 minus 11983.

Input and Output

Using C++ standard library, we can retrieve input from console screen using cin object and output to console screen using cout object.

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

int main() {
    int age;

    cout << “What is your age? ”;
    cin >> age;
    cout << “You age is: ”;
    cout << age << endl;
    return 0;
}
```

For input, cin retrieve values from input stream (istream) and convert it into proper data type. For output, cout converts value to string and put it to output stream (ostream).

Exercise 1.3

Write a program that takes two integer as input and output the sum of them.

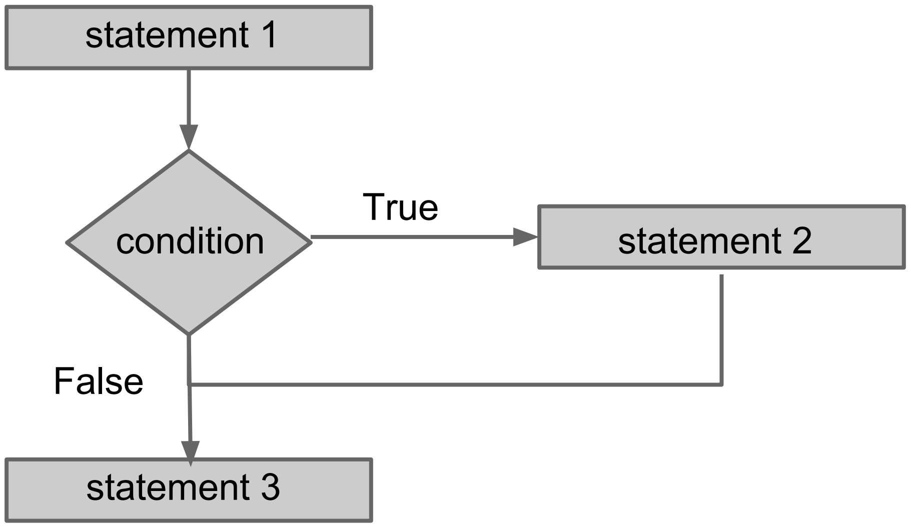
Flow Control

Up to this moment, our program execute statements from the first line to the last line sequentially. Our program won't be much useful if run the same statement sequence every time.

The `if` Statement

When a C++ program must choose whether to take a particular action, we must implement the choice with an `if` statement.

```
statement 1
if ( condition ) {
    statement 2
}
statement 3
```



Condition

Condition is an expression whose value is being tested and resolve to value of true (1) or false (0).

Relational Operators:

Operator	Description
==	Equals
!=	Not Equals
>=	Greater than or Equals
<=	Less than or Equals
>	Greater than
<	Less than

e.g.
12 > 3 resolve to true
12 >= 3 resolve to true
12 >= 12 resolve to true

e.g.
12 < 3 resolve to false
12 <= 3 resolve to false
12 <= 12 resolve to true

Lets look at an example of `if` statement in C++

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

int main() {
    int age = 12;

    if ( age == 12 ) {
        cout << “I am Young\n”;
    }
    cout << “I am ” << age << “ years old\n”;
    return 0;
}
```

Logical operators usually use to combine logical expression to form complex condition expression

Logical Operators:

Operator	Description
||	Logical OR
&&	Logical AND
!	NOT

A	B	A || B
true	true	true
true	false	true
false	true	true
false	false	false

A	B	A && B
true	true	true
true	false	false
false	true	false
false	false	false

A	!A
true	false
false	true

Example using logical operators:

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

int main() {
    int age;

    cin >> age;
    if ( age < 18 && age > 13  ) {
        cout << “You are a teenager\n”;
    }

    if ( age >= 18 ) {
        cout << “You are an adult\n”;
    }
    return 0;
}
```

Exercise 1.4

Read the following expression, determine whether it resolves to true or false.

Assume a = 0, b = 1.

- `12 > 11 && 11 > 12`
- `0 || 12 <=11 || !1`
- `!a || a == 0`
- `( 12 + b ) == 13`
- `b = ( a + b )`
- `!( a + b == 0 )`

`if else` and `else if` statement

`if else` statement contains two execution block, when `if` condition satisfied, statements under corresponding block will be executed, otherwise, statements under `else` block will be executed.

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

int main() {
    int age;

    cin >> age;
    if ( age < 18 && age > 13  ) {
        cout << “You are a teenager\n”;
    } else {
        cout << “You are not a teenager\n”;
    }
    return 0;
}
```

`else if` statement contains multiple execution block, and multiple condition.

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

int main() {
    int age;

    cin >> age;
    if ( age < 18 && age > 13  ) {
        cout << “You are a teenager\n”;
    } else if ( age <= 13 ) {
        cout << “You are not a child\n”;
    } else {
        cout << “You are an adult\n”;
    }
    return 0;
}
```

Exercise 1.5

A leap year (閏年) is a year containing one additional day. Every year divisible by 4 is a leap year, if it is not a century. Every 4th century is a leap year and no other century is a leap year.

Examples:

1. Each of the years 1948, 2004, 1676 etc. is a leap year.
2. Each of the years 400, 800, 1200, 1600, 2000 etc. is a leap year.
3. None of the years 2001, 2002, 2003, 2005, 1800, 2100 is a leap year.

Write a program that takes one input as year number and calculate whether it is a leap year.