COMP2012H Honors Object-Oriented Programming and Data Structures

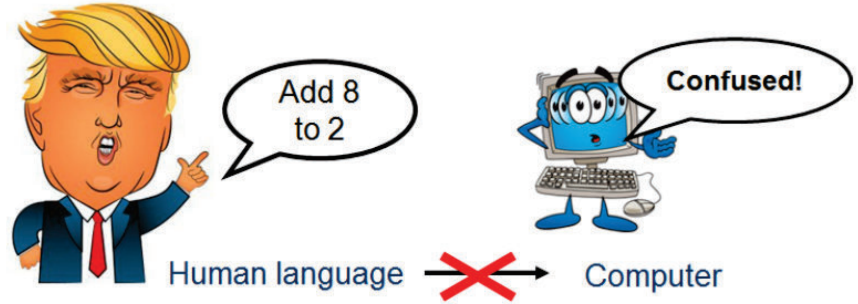
Topic 1: Introduction

Dr. Desmond Tsoi

Department of Computer Science & Engineering
The Hong Kong University of Science and Technology
Hong Kong SAR, China

Computer Programming

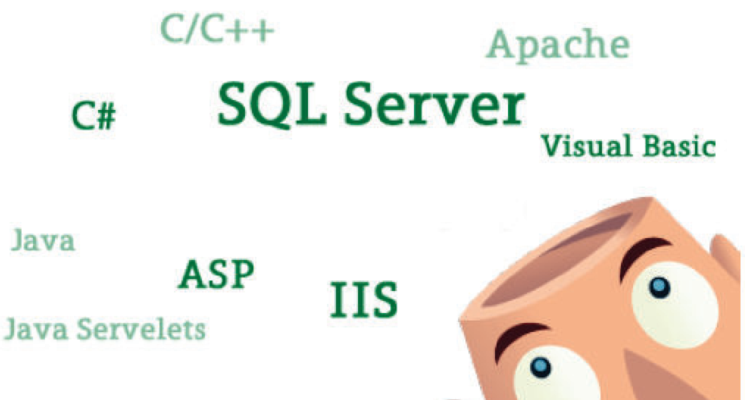
- Learning computer programming is just like learning a natural language such as English, Japanese, Korean, etc.
- Although they are similar, it doesn't mean they are exactly the same.



- Don't worry. Actually this is good. Since computer programming is more systematic and should be much easier to learn, in my opinion. ;) (Good news!)

Programming Languages

- Computer programs are written in programming languages.
- Different to those human languages, a programming language defines A SET OF INSTRUCTIONS in SPECIFIC FORMAT that can be given to a computer.
- Two important issues on writing programs:
  1. Program syntax – Is the grammar of the instructions correct?
  2. Program logic – Is the program able to solve the problem?



Machine Code: Can You Understand This?

```
0000100100101110011001100110100101101100011001010000100100100010011011000110010101100011011101000111
0101011100100110010100110001001011100110001100100010000010100110011101100011011000110011001001011111
0110001101101111011011010111000001101001011011000110010101100100001011100011101000001010001011100111
0011011001010110001101110100011010010110111101101110000010010010001000101110011101000110010101111000
0111010000100010000010100000100100101110011000010110110001101001011001110110111000100000001101000000
1010000010010010111001100111011011000110111101100010011000010110110000100000011011010110000101101001
0110111000001010000010010010111001110100011110010111000001100101000010010010000001101101011000010110
1001011011100010110000100011011001100111010101101110011000110111010001101001011011110110111000001010
0000100100101110011100000111001001101111011000110000100100110000001101000000010100110110101100001011 0
1001011011100011101000001010000010010010000100100011010100000101001001001111010011000100111101000111
0101010101000101001000110010000000110000000010100000100101110011011000010111011001100101001000000010
0101011100110111000000101100001011010011000100110010001110000001011000010010101110011011100000000101 0
0000101000001001001000010010001101010000010100100100111101001100010011110100011101010101010001010010
0011001000000001100010000101000000100101101101011011110111011000100000000110001001011000010010101101111
0011000000000101000000100101110011011101000010000000100101011011110011000000101100010110110010010101 10
0110011100000001011010011001000110000010111010000101000000100101101101011011110111011000100000000110010
0010110000100101011011110011000000001010000001001011100110111010000100000001001010110111100110000001 0
1100010110110010010101100110011100000010110100110010001101000101110100001010000010010110110001100100
0010000000101101100100101011001100111000000101101001100100011000001011101001011000010010101101111 0011
0000000010100000100101101100011001000010000001011011001001010110011001110000000101101001100100011010 0
0101110100101100001001010110111100110001000010100000100101100001011001000110010000100000000100101011 0
1111001100000010110000100101011011110011000100101100001001010110111100110000000010100000100101110011
0111010000100000000100101011011110011000000101100010110110010010101100110011100000001011010011001000 11
1000010111010000101000000100101101101011011110111011000100000000110000000101100001001010110100100110000
0000101000000100101100010001000000001011100100110001001100001100010000101000000100101101110011011110111
0000000010100010111001001100010011000011000100111010000001010000001001011100100110010101110100000001010
0000100101110010011001010111001101110100011011110111001001100101000010100010111001001100010011000110
0110011001010011000100111010000001010000001001001011100111001101101001011110100110010100001001001000000
0110110101100001011010010110111000101100001011100100110001001100011001100110010100110001001011010110
1101011000010110100101101110000001010000001001001011100110100101100100011001010110111001110100000001001
0010001001000111010000110100001100111010001000000001010000100011101001110010101010010100100100000011
001000101110001110000001011100011000100100010000001010
```

Assembly Language: How About This?

```
main:
  !#PROLOGUE# 0
  save %sp,-128,%sp

  !#PROLOGUE# 1
  mov 1,%o0
  st %o0,[%fp-20]
  mov 2,%o0
  st %o0,[%fp-24]
  ld [%fp-20],%o0
  ld [%fp-24],%o1
  add %o0,%o1,%o0
  st %o0,[%fp-28]
  mov 0,%i0
  nop
```

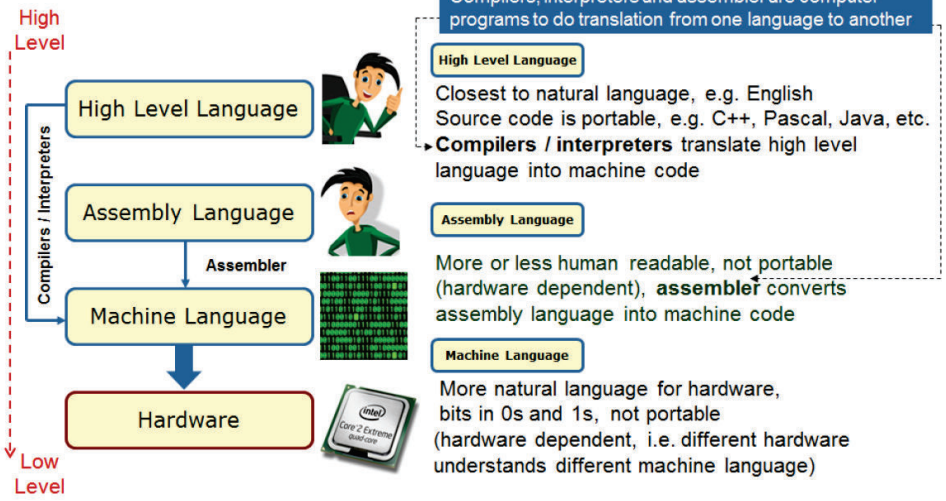
High-Level Language: Is This Better Now?

```
int main( )
{
    int x, y, z;

    x = 1;
    y = 2;
    z = x+y;

    return 0;
}
```

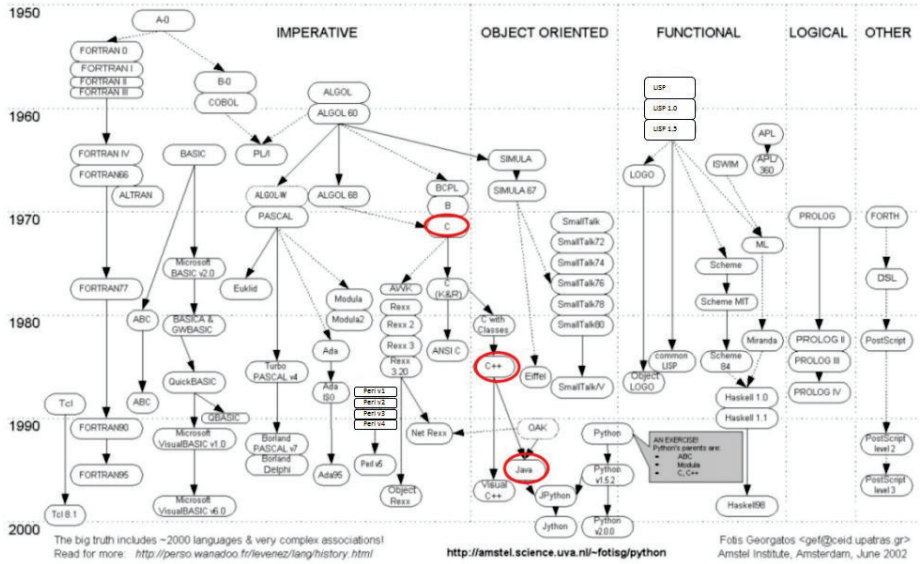
Write a Program to Sum 2 Numbers

- There are 3 integer-value-holding objects: x, y, and z.
- x and y have the value of 1 and 2 respectively.
- z's value is the sum of x's and y's.

Levels of Programming Languages



Chronology of Some High Level Programming Languages

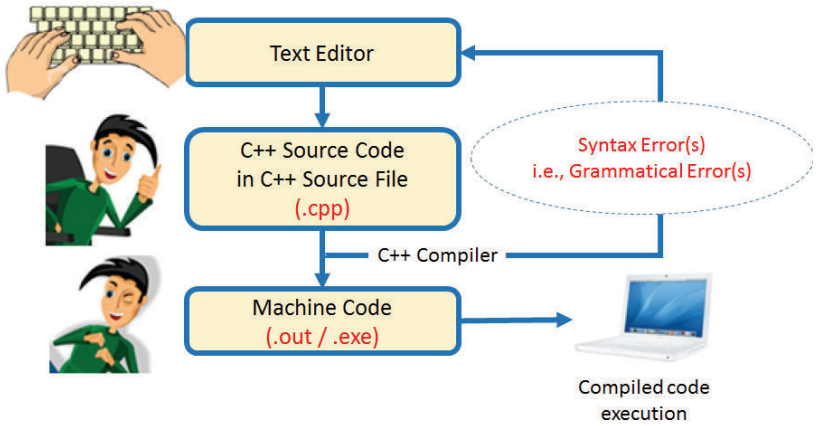


Which Programming Language Are We Going to Use?

We are going to use C++ in this course!

- Why C++?
  Read the FAQ from the designer of C++, Bjarne Stroustrup.
- Which C++?
  - The language has been evolving:
    C++ 1983 ⇒ C++ 1998 ⇒ C++ 2003 ⇒ C++ 2011 ⇒ ···
  - We will learn C++11 (but not all the new features).
- Which compiler?
  GNU gcc/g++. It is free.
  (The compiler you will use is C++11-compliant.)
- Which IDE (integrated development environment) for writing programs?
  VSCode. It is free and supported by many operating systems such as Windows, Mac OS, and Linux.

Development Cycle of a C++ Program



- A compiler translates source programs into machine codes that run directly on the target computer.
- For example, a.cpp ⟶ a.out (or a.exe).
- Some C++ compilers: gcc/g++, VC++.

Example: Hello World!

```
/*
 * File: hello-world.cpp
 * A common program used to demo a new language
 */

#include <iostream>     // Load info of a Standard C++ library
using namespace std;    // Standard C++ namespace

int main()              // Program's entry point
{
    /* Major program codes */
    cout << "Hello World!" << endl;

    return 0;           // A nice ending
}
```

Write, Compile, and Run a Program in a Terminal

STEP 1 : Write the program using an editor.
e.g., VSCode, vi (Unix/Linux), MS Word (Windows)

STEP 2 : Save the program into a file called hello-world.cpp.

STEP 3 : Compile the program using g++ compiler.

```
g++ -o hello-world hello-world.cpp
```

If you don't specify the output filename using the "-o" option, the default is a.out.

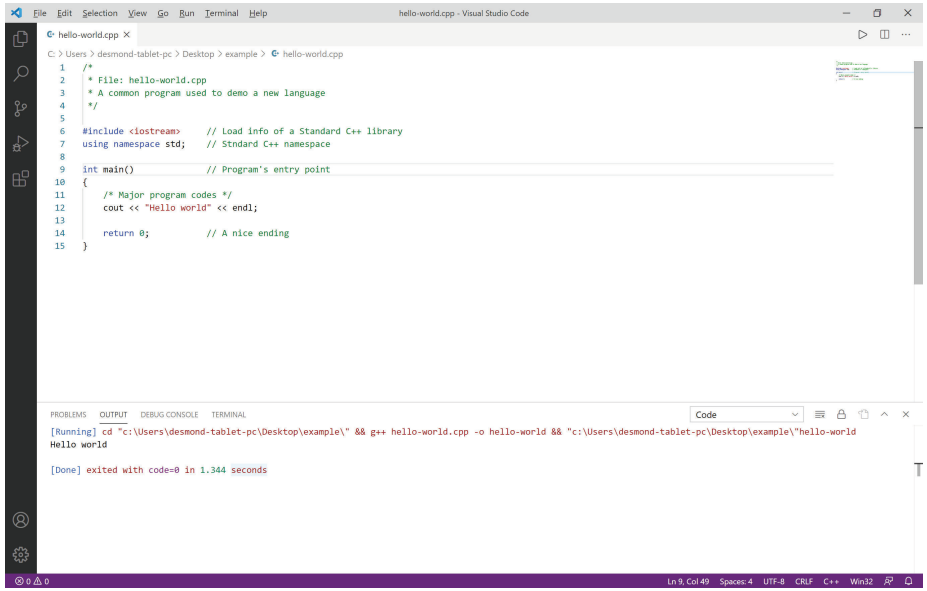
```
g++ hello-world.cpp
```

STEP 4 : Run the program in a terminal (command window):

```
linux:: hello-world
Hello World!
```

VSCode IDE for C/C++

In the lab, you will use VSCode (similar to MS Visual Studio).

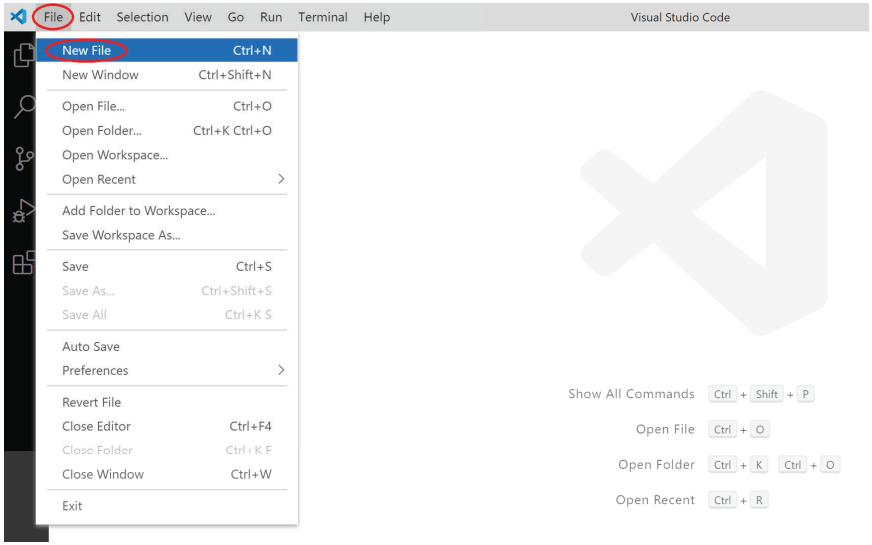


Program Development using VSCode

Step 1	Create a new file	Ctrl-N (or Click File → New File)
Step 2	Write program	VSCode built-in editor
Step 3	Save program	Ctrl-S (or Click File → Save)
Step 4	Compile and run program	F1 → "Run code"

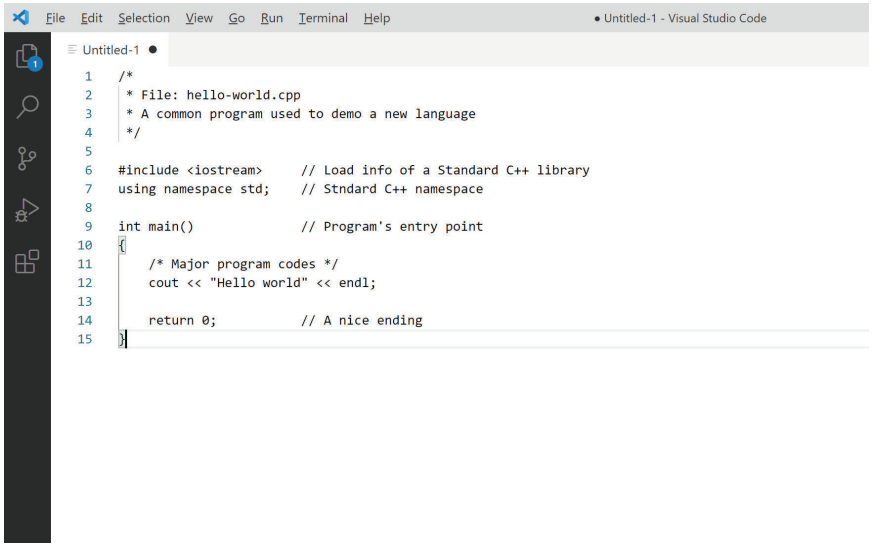
Step 1a: Create a new file

- Ctrl-N (or Click "File" → "New File")



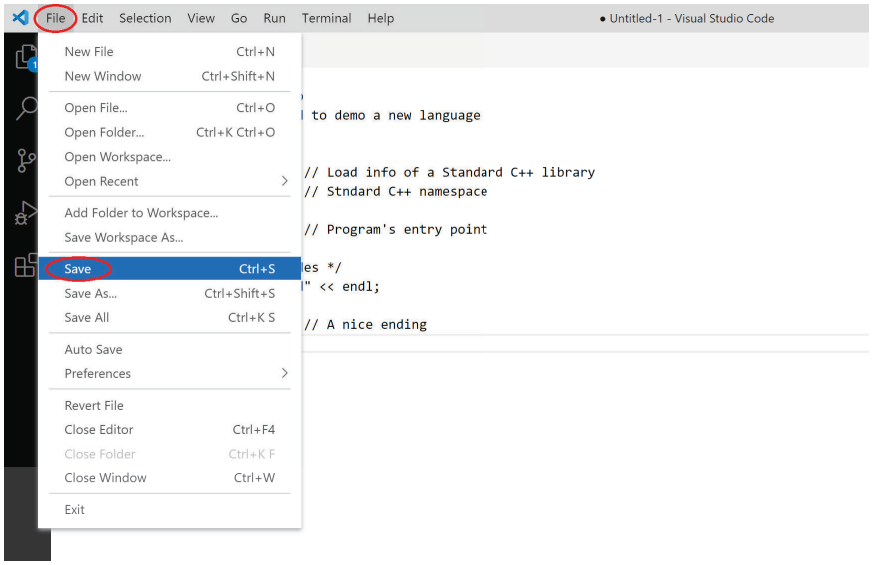
Step 2: Write program

- Write your program



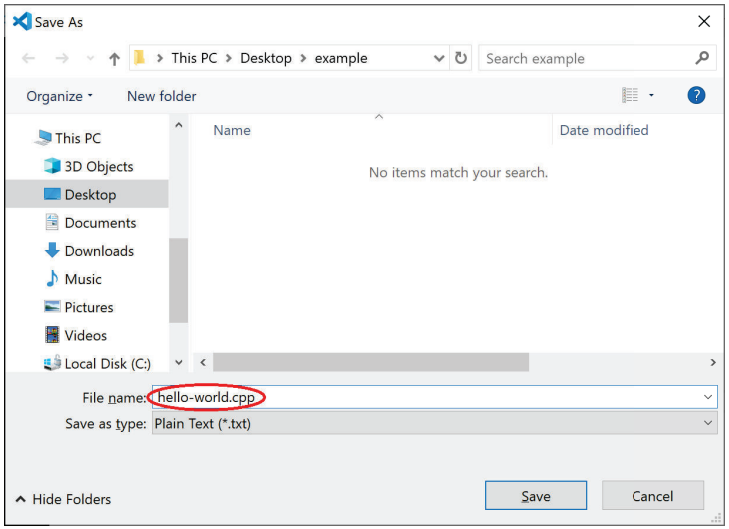
Step 3a: Save program

- Ctrl-S (or File → Save)



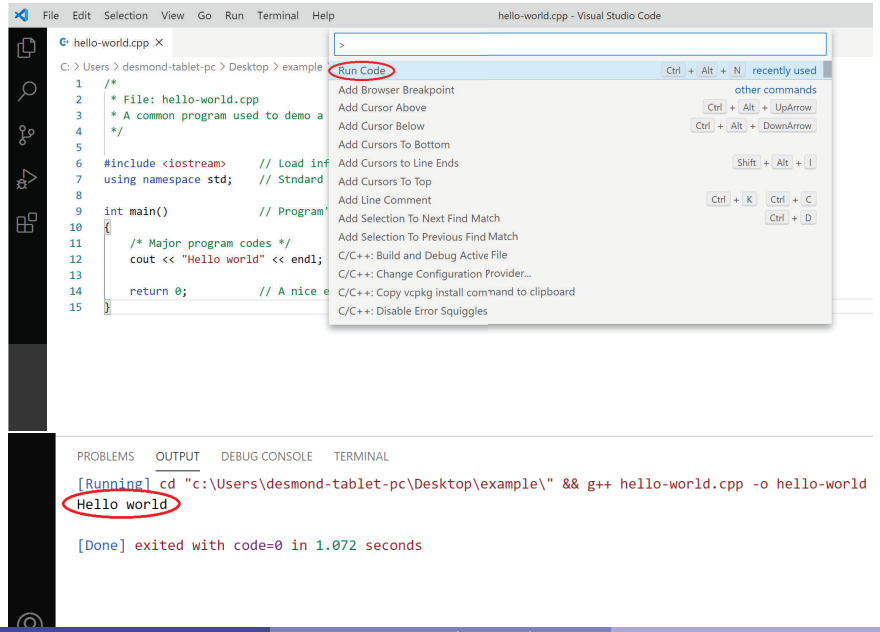
Step 3b: Save program

- Choose a location, give it a proper name, and click "Save".
  Note: make sure the name is something which ends with .cpp to indicate that is a C++ source file (e.g., hello-world.cpp).



Step 4: Compile and run program

- F1 → click "Run code"



Main: the Entry Point

- Every program must have exactly one and only one main( ) function.

Simple Form of the main Function

```
int main ( ) { ··· }
```

General Form of the main Function

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

(We'll talk about argc and argv later.)

- Between the braces "{" and "}" are the program codes consisting of zero or more program statements.
- Each simple C++ statement ends in a semicolon ";".

C++ Comments

- Use `/* ··· */` for multiple-line comments.

```
/*
 * A common program used to demo a new language
 */
```

- Single-line comments start with `//`.

```
// Program's entry point
```

- Comments are just for human to read.
- They will not be translated by the compiler into machine codes.

#include and Standard C++ Libraries

- #include will include information of a library — a collection of sub-programs. e.g. `#include <iostream>` gets the information of the standard C++ library called iostream that deals with I/O:
  - cin: an object to read, e.g., from the keyboard or file
  - cout: an object to print out, e.g., to the screen or file
  - cerr: an object to print error message, e.g., to the screen or file

Examples

```
// endl means "end of a line"
cout << "Einstein: God does not play dice." << endl;

// You may also break down the message in several lines
cerr << "Error: "
     << "There is no stress and tension in HKUST!"
     << endl;
```

- These library information files are called header files.

#include and User-defined Libraries

- You may also define your *own* library.
- Again you need to use #include to include its information into your sub-programs.
- Example: `#include "drawing.h"` gets the information of a user-defined C++ library about drawing.
- By convention, the header file of a user-defined library ends in ".h" or ".hpp", while Standard C++ library header files have no file suffix.
- Also by convention, the header file of a user-defined library is delimited using double-quotes "..." , while Standard C++ library header files use < ... >.

C++ is a Free Format Language

- Extra blanks, tabs, lines are ignored.
- Thus, codes may be indented in any way to enhance readability.
- More than one statement can be on one line.
- Here is the same Hello World program:

```
#include <iostream> /* File: hello-world-too.cpp */
using       namespace std; int main   (int argc,
   char** argv) { cout<<"Hello World!"<<endl;return   0;}
```

- On the other hand, a single statement may be spread over several lines.

```
cout << "Hello World!"
     << endl;
```

Good Programming Style

- Place each statement on a line by itself.
- For long statements
  - if possible, break it down into several shorter statements.
  - wrap it around with proper indentation (since extra space doesn't matter!)
- Use blank lines to separate sections of related codes that together perform some action.
- Indent consistently. Use the same indentation for the same block of codes.

Programming as Problem Solving

- Understand and define the problem clearly.
  - What are the input(s) and output(s)?
  - Any constraints?
  - Which information is essential?
- Develop a solution.
  - Construct an algorithm.
- Translate the algorithm into a C++ program.
- Compile the program.
- Test the program.
- Debug the program.
- Document the program as you write the program.
- Maintain the program
  - modify the codes when conditions change.
  - enhance the codes to improve the solution.

What Makes a Good Program?

- Correctness
  - Meets the problem requirements
  - Produces correct results
- Easy to read and understand
- Easy to modify
- Easy to debug
- Efficient
  - Fast
  - Requires less memory

That's all!

Any questions?