MH1402 Algorithms & Computing II

Lecture 0 Introduction

Wu Hongjun

Overview

- **Course Information**
  - **Goals, Syllabus, Textbook, Labs, Exam, Grading, Consultation**
- **Overview of Programming Languages**
- **C++ Programming Language**

Course Information

- Instructor
- Goals
- Syllabus
- Textbook
- Lectures/Labs
- Grading
- Consultation

Instructor

Asst Prof Wu Hongjun

School of Physical & Mathematical Sciences

Division of Mathematical Sciences

Office: SPMS-MAS-05-47

Email: wuhj@ntu.edu.sg

Goals of this course

Learn

- **Basics of C++ programming language**

After taking this course, you should be able to

- **Write C++ programs to solve problems**
- **Read C++ programs**

Syllabus

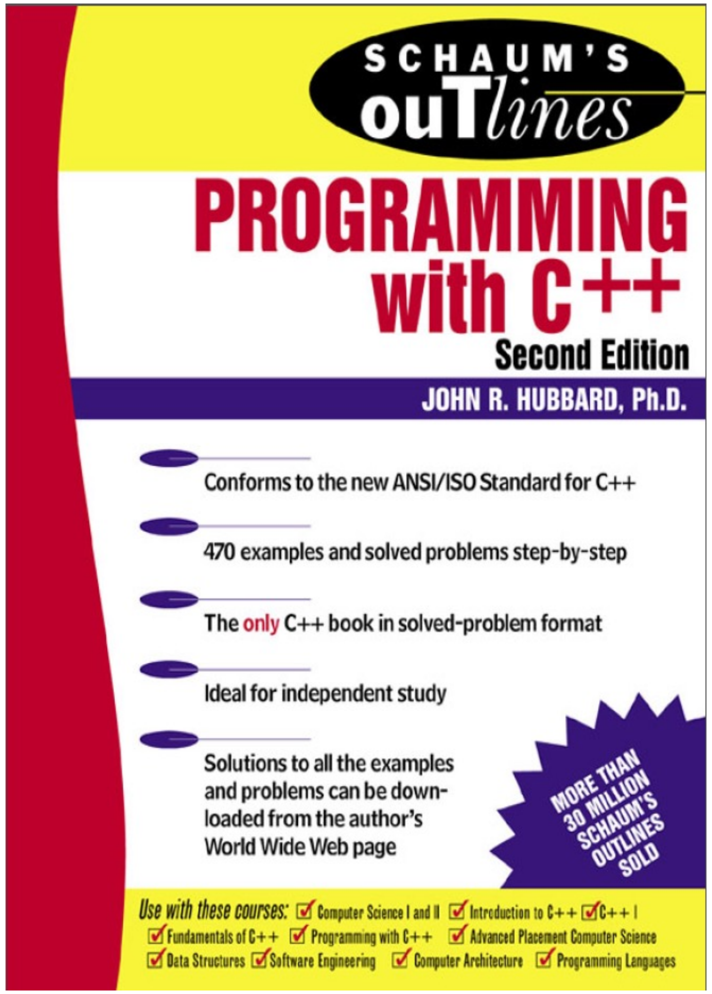
The focus of this course is on the basics of C++ programming language:

1. **Data Types and Operators**
2. **Control Statements: Selection and Loop**
3. **Functions**
4. **Arrays and Vectors**
5. **Pointers and References**
6. **File Input/Output**
6. **Strings**
7. **Classes**

Textbook

- **Schaum's Outlines Programming with C++**
  - Author: J.R. Hubbard
  - Publisher: McGraw-Hill
  - 2nd edition (May 16, 2000)
  - ISBN-13: 978-0071353465
  - Practical, clear and concise
  - Ideal for practicing and clarifying problematic issues
  - Cheap: NTU bookstore sells it at S$20

  The author's homepage:
  http://www.mathcs.richmond.edu/~hubbard/Books.html


SCHAUM'S ouTlines

PROGRAMMING with C++

Second Edition

JOHN R. HUBBARD, Ph.D.

- Conforms to the new ANSI/ISO Standard for C++
- 470 examples and solved problems step-by-step
- The only C++ book in solved-problem format
- Ideal for independent study
- Solutions to all the examples and problems can be downloaded from the author's World Wide Web page

MORE THAN 30 MILLION SCHAUM'S OUTLINES SOLD

Use with these courses: ✓ Computer Science I and II ✓ Introduction to C++ ✓ C++ I ✓ Fundamentals of C++ ✓ Programming with C++ ✓ Advanced Placement Computer Science ✓ Data Structures ✓ Software Engineering ✓ Computer Architecture ✓ Programming Languages

Lectures / Labs

- **Lectures: 1 hour per week**
  - **Friday 14:30 – 15:30, LT19A**
- **Computer Labs: 2 hours per week**
  - **Lab sessions are in MAS Computer Labs 1,2,3**
  - **Location of labs: SPMS-MAS, level 3**
  - **Labs starts from the second teaching week**
  - **The latest information of your Teaching Assistants can be found from NTULearn ➔ course information ➔ Lectures and Labs**

Lab Schedule

Class Type	Group	Day	Time	Venue	Remark	Teaching Assistant (Tentative)
LAB	LA1	M	11.30-13.30	COMP LAB 1	WK 2-13	Jefferson Choi, Huang Tao, Adamas Adsa Fahreza
LAB	LA2	M	11.30-13.30	COMP LAB 2	WK 2-13	Yu Haiwan, Ivica Nikolic
LAB	LA3	M	11.30-13.30	COMP LAB 3	WK 2-13	Huzhang Guangda, Liu Yucheng
LAB	LA4	T	13.30-15.30	COMP LAB 1	WK 2-13	Jefferson Choi, Adamas Adsa Fahreza, Wang Chenyu
LAB	LA5	T	13.30-15.30	COMP LAB 2	WK 2-13	Yu Haiwan, Dirmanto Jap
LAB	LA6	T	13.30-15.30	COMP LAB 3	WK 2-13	Huzhang Guangda, Liu Yucheng
LAB	LA7	F	10.30-12.30	COMP LAB 1	WK 2-13	Bai Zhongzheng, Huzhang Guangda, Wee Jun Jie
LAB	LA8	F	10.30-12.30	COMP LAB 2	WK 2-13	Wu Hongjun, Wang Chenyu

Contacts of Teaching Assistants

Adamas Aqsa Fahreza		ADAMASAQ001@e.ntu.edu.sg
Bai Zhongzheng		BAIZ0005@e.ntu.edu.sg
Dirmanto Jap	(Dr)	DIRM0002@e.ntu.edu.sg
Ivica Nikolic	(Dr)	inikolic@ntu.edu.sg
Huang Tao	(Dr)	huangtao@ntu.edu.sg
Jefferson Choi		JEFF0010@e.ntu.edu.sg
Huzhang Guangda		GHUZHANG001@e.ntu.edu.sg
Liu Yucheng		LIUY0103@e.ntu.edu.sg
Wang Chenyu		CWANG014@e.ntu.edu.sg
Wee Jun Jie		WEEJ0016@e.ntu.edu.sg
Wu Hongjun	(Dr)	wuhj@ntu.edu.sg
Yu Haiwan		HYU012@e.ntu.edu.sg

Grading

- **Assignment 1 (5%)**
- **Assignment 2 (5%)**
- **Test 1 (30%): Open book on computer,**
  after the recess week (date to be confirmed)
- **Test 2 (60%): Open book on computer,**
  after the last teaching week (date to be confirmed)

- **Note:**
  Different exam papers will be used (not enough computers in the labs).
  The students who use different exam papers will be graded separately (i.e., the students who use different exam papers will not compete with each other. it is to minimize the drawback of using different exam papers)

Grading (cont.)

- **Assignment 1 & 2 are two randomly selected lab submissions**
    - **You should submit all the labs**
    - **You do not know which two lab submissions get selected**
    - **The main purpose is to ensure that you work on every lab**

- **Note that tests are open book**
    - **You do not need to memorize the C++ syntax**
    - **You should practice a lot in order to get familiar with C++**
    - **You should start practicing C++ starting from the first week**
- **Open book:**
    - **You can bring any paper/notes/books to the exam**
    - **You can copy any digital data into your exam computer before the exam.**
    - **No network connection during the exam**
    - **You should not use USB/thumb drive during the exam.**

Grading (cont.)

- **Test 1 is considered as midterm exam**
- **Test 2 is considered as final exam**
- **No makeup tests**
- **What would happen if a student does not take Test 1?**
  - **If the student has valid reason (such as MC), Test 2 of that student would be 90% in the final grade**
  - **If the student does not provide valid reason, zero marks for Test 1.**
- **What would happen if a student does not take Test 2?**
  - **If the student has valid reason (such as MC), then the student does not have grade for this course (Absent with valid reason or Medical Leave)**
  - **If the student does not provide valid reason, the student fails this course**

Grading (cont.)

- **Important: the grading of this course will not follow any curve**
  - **You are not competing with your classmates**
  - **You should learn C++ well in order to get good grade or pass this module**

(We will have two or three exam groups, depending on the number of students taking this course. Each exam group uses different exam papers.

The grades will be adjusted slightly to ensure that different exam groups have roughly the same percentages of grades.)

Consultation

- **You will ask questions in NTULearn → Discussions**
  You are encouraged to participate in the online discussion
  - **If you help to answer a question in the Discussion forum, I will give you**
    0.5 or 1 bonus marks if your answer is good
    (capped at 5 bonus marks in total for each student)
    Please do not abuse this award scheme ☺
- **You may visit my office**
  Timing: 4.00pm - 5.00pm, Friday
  Venue: SPMS-MAS-05-47

Why do we learn programming?

Programming is important

- **Computers are very powerful today**
  - **A $1000 computer today can perform several billion operations in one second**
    - **Here one operation means a 64-bit (20-digit) integer addition, subtraction, multiplication;**
    - **One operation may mean a double precision real number (floating point) addition, subtraction, multiplication**
  - **Without using an electronic computing device, suppose that you perform 1 operation per second, and you work 24 hours a day, it takes about 32 years to perform one billion operations**

Programming is important

- **To let the computers work for us, we need to learn computer languages (or called programming languages)**
  - **You learned calculating on paper in primary school**
  - **Later you get familiar with using calculator**
  - **In university, you should get familiar with programming on computers**

Programming is important

- **USA National Research Council identified fundamental skills for mathematical sciences students by 2025:**
  - **The ability to deal with large sets of data**
  - **To think algorithmically and employ computation**
  - **……**

- **For physical sciences, the ubiquity of simulation in physics requires strong algorithm and computing skills**
  - **Example: the most powerful supercomputers in USA are used by physicists**

C++ Programming Language

Overview of Programming Languages

- **Lowest-level programming language: Machine language**
  - **The computer executes the machine code directly**
  - **But it is very difficult for us to read and write**
  - **The code is not portable (Different machine language for different CPUs)**

- **Low-level programming language: Assembly language**
  - **Difficult to write and read**
  - **not portable: different assembly language for different CPUs**

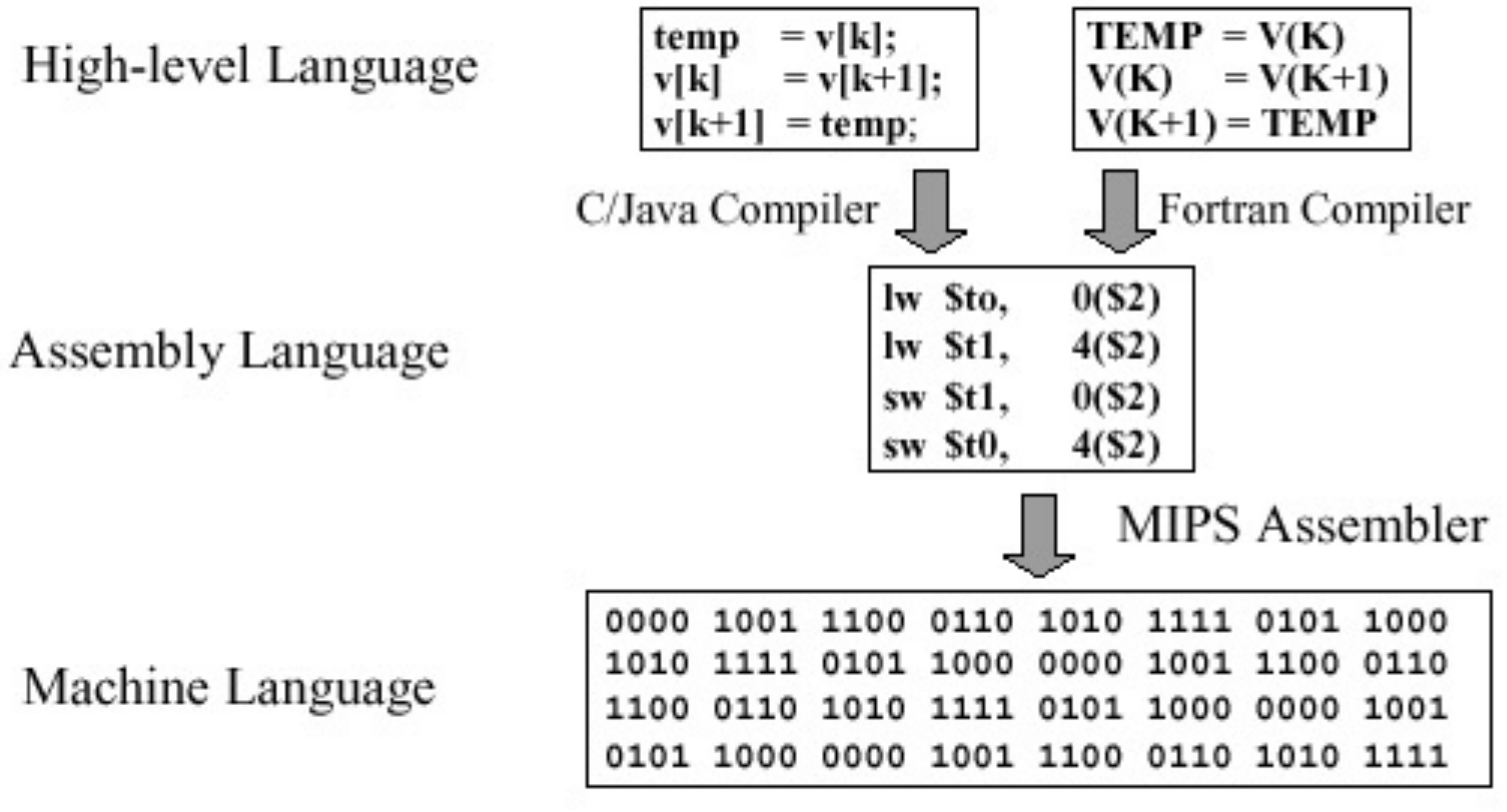
- **High level programming languages (next slide)**

Overview of Programming Languages

- **High level programming languages**
  - **Easy to write and read**
  - **Portable: the same code can be used on different computers**
  - **More than 2000 high level programming languages**
    - **some of them are listed at Wikipedia: http://en.wikipedia.org/wiki/List_of_programming_languages**
  - **The most widely used programming languages today are:**

    C, C++, C#, Objective-C, SQL, PHP, Java, Javascript, Ruby, Perl, Python, Visual Basic ….

Examples of machine code, assembly code, and high level code



```
temp    = v[k];
v[k]    = v[k+1];
v[k+1]  = temp;
```

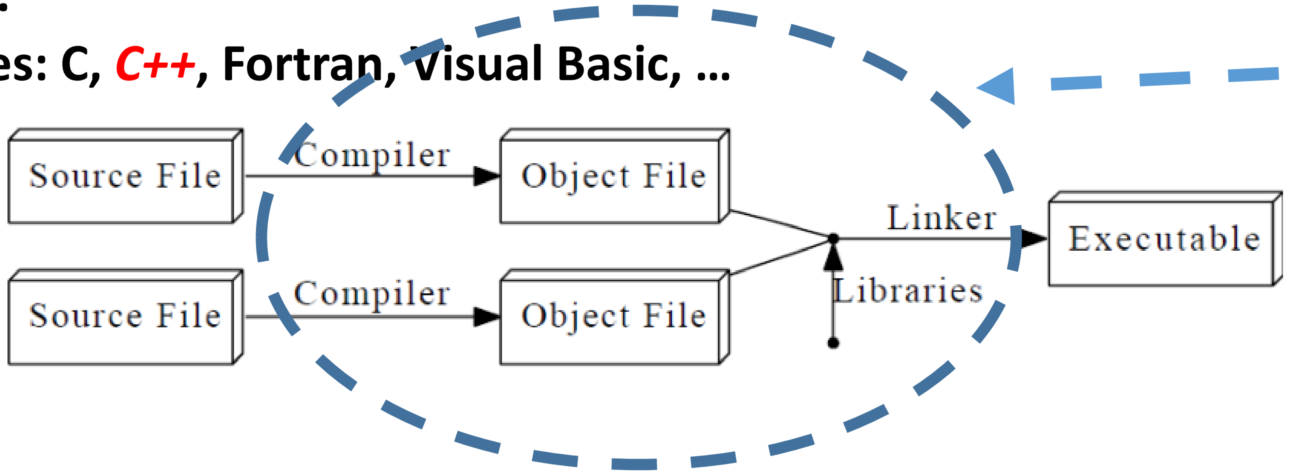
```
TEMP   = V(K)
V(K)   = V(K+1)
V(K+1) = TEMP
```

```
lw  $to,   0($2)
lw  $t1,   4($2)
sw  $t1,   0($2)
sw  $t0,   4($2)
```

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

Overview of Programming Languages

- **Two basic models that a high-level programing code get executed**
  - **Interpreted languages**
    - **Interpreted languages are read and then executed directly. A program called an interpreter reads each program statement, and execute it.**
    - **Examples: *MATLAB*, Mathematica, Perl, PHP, Python, Ruby, …**
  - **Compiled languages**
    - **Compiled languages are transformed from source codes into an executable form before running.**
    - **Examples: C, *C++*, Fortran, Visual Basic, …**



We simply say that the source codes are compiled into executable program

C++ programming language

- **Created by Bjarne Stroustrup in 1979**
  - **High-level programming language**
  - **Compiled programming language**
- **C++ extends the C programming language**
  - **C++ may be viewed as a superset of C**
  - **C++ introduces object oriented programming feature to C (classes in C++)**

Applications of C++

http://www.stroustrup.com/applications.html

http://www.lextrait.com/vincent/implementations.html

C++ is very popular, and is used extensively in developing:

- **Computer operating systems**
  - **Microsoft Windows (95, 98, NT, XP, VISTA, 7, 8, 10)**
  - **Apple MAC OS X, iOS**
  - **Google Chrome OS**
- **Application software**
  - **Microsoft Office (Word, Excel, Powerpoint, Outlook)**
  - **Some websites: Amazon, facebook …**
  - **Almost all the Web browsers (IE, Chrome, Firefox, Opera)**
  - **C/C++ compilers (GCC, Microsoft Visual C++)**
  - **………..**
- **Scientific computation**

C++ vs. MATLAB

We already learned MATLAB, why do we still learn C++?

- **MATLAB is useful for scientific computing (easy to use)**
- **C++ is a general-purpose programming language**
  - **C++ is powerful for scientific computing**
    - **C++ code is faster**
    - **A number of free C++ mathematical libraries are available**
      http://en.wikipedia.org/wiki/List_of_numerical_libraries#C.2B.2B
  - **C++ is a powerful programming language for many applications**
    - **C++ program can access low level computer resource (such as memory)**
    - **C++ is suitable for developing professional software**

C++ vs. MATLAB

We already learned MATLAB, why do we still learn C++?

- **The use of MATLAB requires commercial license**
- **It is free to use C++**

Three Simple C++ Programs

C++ Code

- **C++ codes are saved in text format, with file name extension of *.cpp* and *.h***

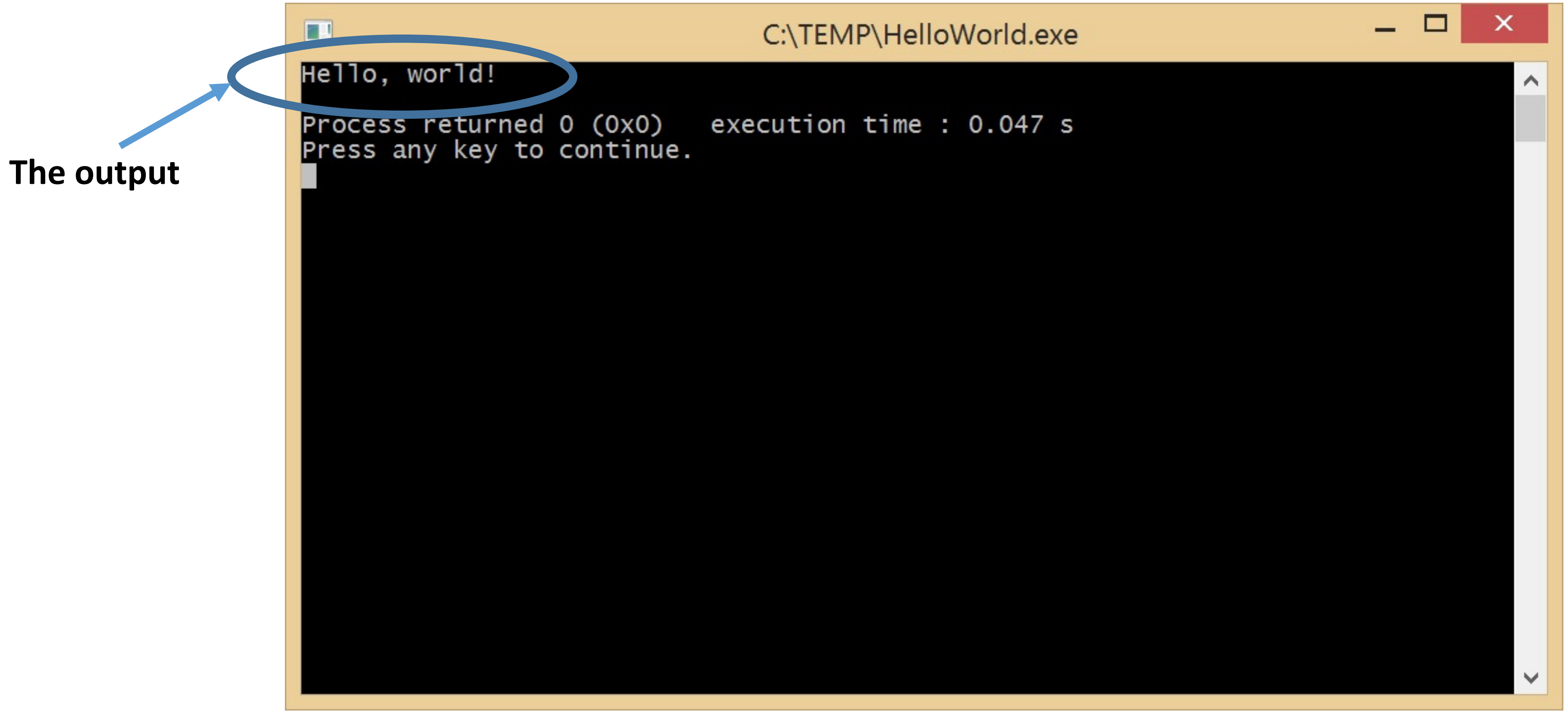
.cpp	(C++ source file)
.h	(C++ header file, we will learn header file later)

- **Any text editor (such as notepad) can be used to write C++ codes**
  - In this course, we use the editor of Code::Blocks
  - The installation of Code::Blocks are illustrated in NTULearn.

The first C++ program: HelloWorld

```
//A simple C++ program
#include <iostream>

using namespace std;

int main( )
{
     cout << "Hello, world!" << endl;
     return 0;
}
```

- **Save the program on the previous slide as a .cpp file (for example, helloworld.cpp)   (Details are given in Lab 0)**
- **Then compile (and link) helloworld.cpp into an executable file.**
- **Run the executable file, it prints out "Hello, world!" to the screen.**



```
Hello, world!

Process returned 0 (0x0)   execution time : 0.047 s
Press any key to continue.
```

The detailed explanation

//A simple C++ program

Explanation:

- **// indicates that everything following it until the end of that line is a comment. A comment is ignored by the compiler**
- **Another way to write a comment in C++ is to put it between /* and */**

  The comment may span several lines, for example

  ```
  /* This is
  a simple
  program */
  ```
- **In MATLAB, % indicates that everything following it until the end of that line is comment**

The detailed explanation

#include <iostream>

Explanation:

- **iostream defines the procedure for input/output**
- **#include <iostream> is to use the content of "iostream" so that our program can print to the screen using the "cout" in "iostream"**
  - **#include <iostream> is needed in most of the C++ programs**
- **If you do not understand it at the moment, it is ok, just copy it to your program**

The detailed explanation

using namespace std;

Explanation:

- **“std” is the standard library of C++. It provides a rich collection of functions for input/output (such as “iostream”), mathematical calculations, string/character manipulations**
- **“using namespace std;” indicates that the “cout” in our program is defined in “std”**
  - **If we do not use “using namespace std;” at the beginning of the program, we should use “std::cout” instead of “cout” in our program**
- **Semilolon “;” must be used at the end to indicate the end of this statement**
- **It does no matter if you do not understand it at the moment, just copy it to your program**

The detailed explanation

```
//A simple C++ program
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello, world!" << endl;
    return 0;
}
```

int main() { … } **defines the code that should execute when the program starts up.**

- **Every C++ program should have one and only one int main() (very different from MATLAB)**
- **More on it when we learn C++ functions**

The curly braces {…} represent grouping of multiple statements into a statement block

The detailed explanation

```
//A simple C++ program
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello, world!" << endl;
    return 0;
}
```

"Hello, world!" , double quotations are used to indicate the string Hello, world!

cout << prints something (here the string) to the screen, note the direction of << (<< is called insertion operator)

endl breaks a line

; must be used to indicate the end of statement (semicolon is known as the statement terminator), different from Matlab

Detailed explanation

```
//A simple C++ program
#include <iostream>
using namespace std;

int main()
{
    cout << "Hello, world!" << endl;
    return 0;
}
```

return 0; means this function (main) returns integer 0 when the program terminates

- A C++ main function must return an integer when it terminates.
- Returning 0 in main function indicates that the program finishes successfully
- Returning a nonzero integer indicates an abnormal termination.
- If "return 0;" is missing in the C++ main function, it is not an error since the program would automatically return 0 at the end of the execution of main function; but it is a good practice to have "return 0;" at the end of the main function.

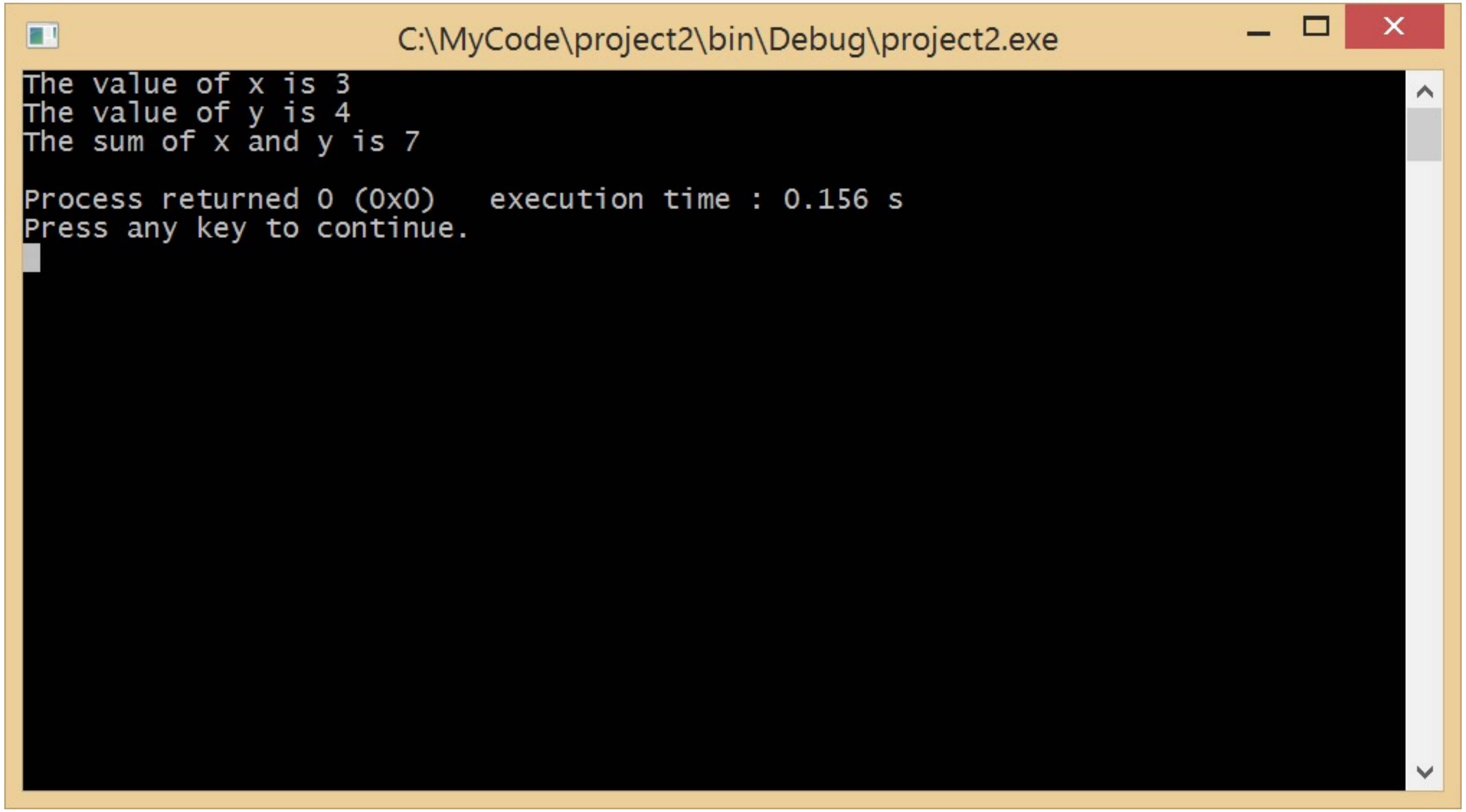
The second C++ Program: Simple Computing

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

int main( )
{
     int x, y, z;
     x = 3;
     y = 4;
     z = x+y;
     cout << "The value of x is " << x << endl;
     cout << "The value of y is " << y << endl;
     cout << "The sum of x and y is " << z << endl;
     return 0;
}
```

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

int main( )
{
     int x, y, z;    //  Declare three integer variables x, y and z
                    //   Every variable in C++ must be declared before being used
                    //   (different from MATLAB)
     x = 3;         //   Assign 3 to the variable x
     y = 4;
     z = x+y;       //  compute the sum of x and y,
                    //   then assign the value of the sum to variable z
     cout << "The value of x is " << x << endl;
                    //   print the string "The value of x is " to the screen
                    //   then print the value of x to the screen
     cout << "The value of y is " << y << endl;
     cout << "The sum of x and y is " << z << endl;
     return 0;
}
```

Save the program on the previous slide into a file with .cpp extension.

Compile and execute it, we get the output:



```
The value of x is 3
The value of y is 4
The sum of x and y is 7

Process returned 0 (0x0)   execution time : 0.156 s
Press any key to continue.
```

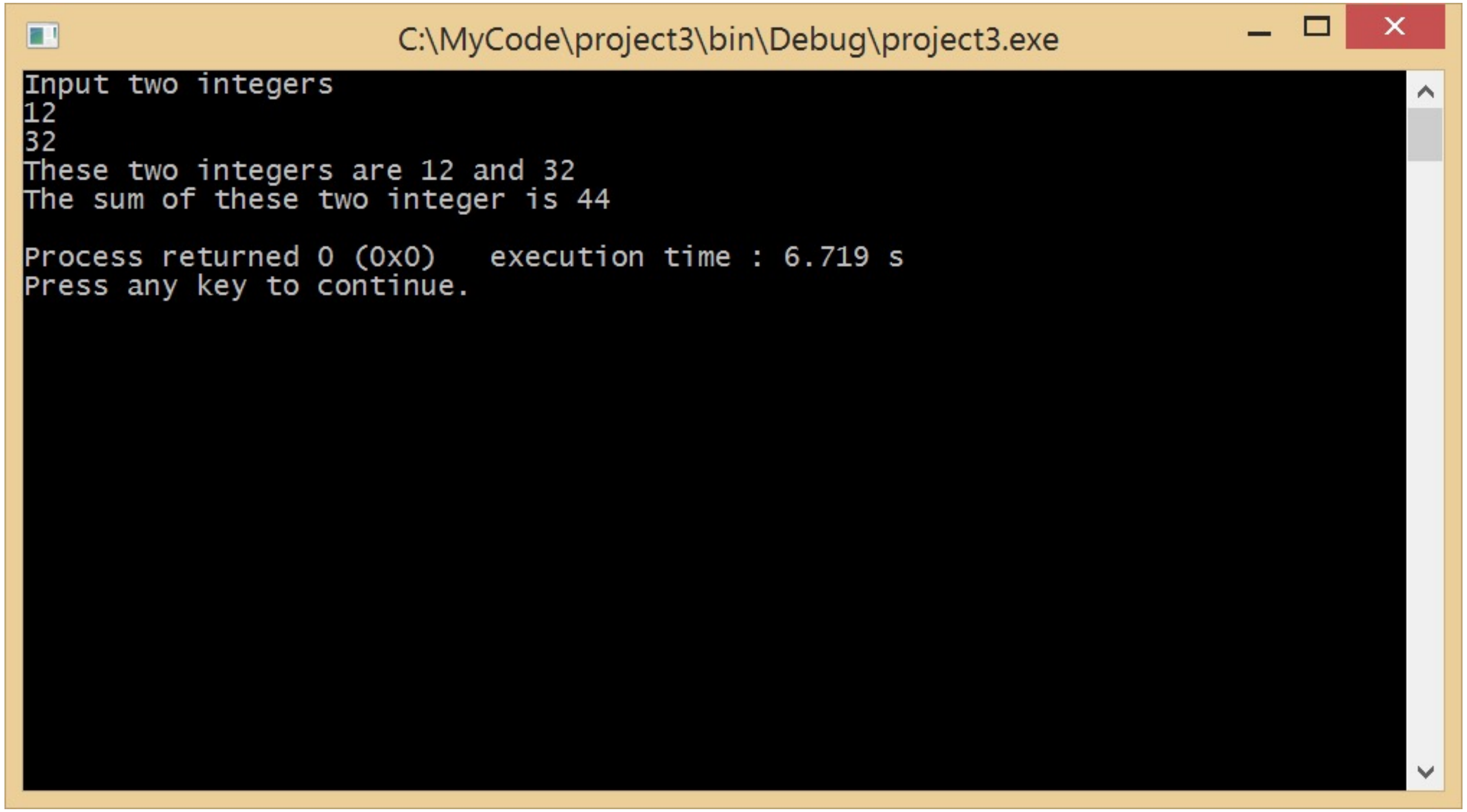
The third C++ program: input data

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

int main( )
{
     int x, y, z;
     cout << "Input two integers "<< endl;
     cin >> x >> y;
     z = x+y;
     cout << "These two integers are " << x << " and " << y << endl;
     cout << "The sum of these two integer is " << z << endl;
     return 0;
}
```

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

int main( )
{
     int x, y, z;
     cout << "Input two integers "<< endl;
     cin >> x >> y;    // use cin to input two integers, store them in x and y.
                       //  note the direction of >>  (extraction operator)
                      //  press “enter” after typing each integer
                      //  or press “enter” after typing two integers
                     //  (with space between those two integers)
     z = x+y;
     cout << "These two integers are " << x << " and " << y << endl;
     cout << "The sum of these two integer is " << z << endl;
     return 0;
}
```

Save the program on the previous slide into a file with .cpp extension.

Compile and execute it, we get the output:



```
Input two integers
12
32
These two integers are 12 and 32
The sum of these two integer is 44

Process returned 0 (0x0)   execution time : 6.719 s
Press any key to continue.
```

Advices

- **Practice is extremely important for learning programming**
  - **Use all 3 sources for practicing: labs, lectures, textbook**
  - **Practicing means creating working programs**
  - **Test (or even modify) all programs from lectures and labs in Code::Blocks**
    - **Code::Blocks is used in this course to edit, compile and debug C++ codes**
    - **Code::Blocks is installed on all the computers in the labs**
    - **You can install Code::Blocks on your computer following the instructions given in NTULearn**

Advices

- **Don’t be afraid of writing program**
    - **Everyone makes a lot of programming errors**
        - **We should get familiar with correcting those errors (debugging)**
- **Don’t worry, your C++ programs will not physically destroy your computer**