

IT001 - Nhập môn Lập Trình

BÀI 01: TỔNG QUAN VỀ MÁY TÍNH VÀ PHẦN MỀM MÁY TÍNH



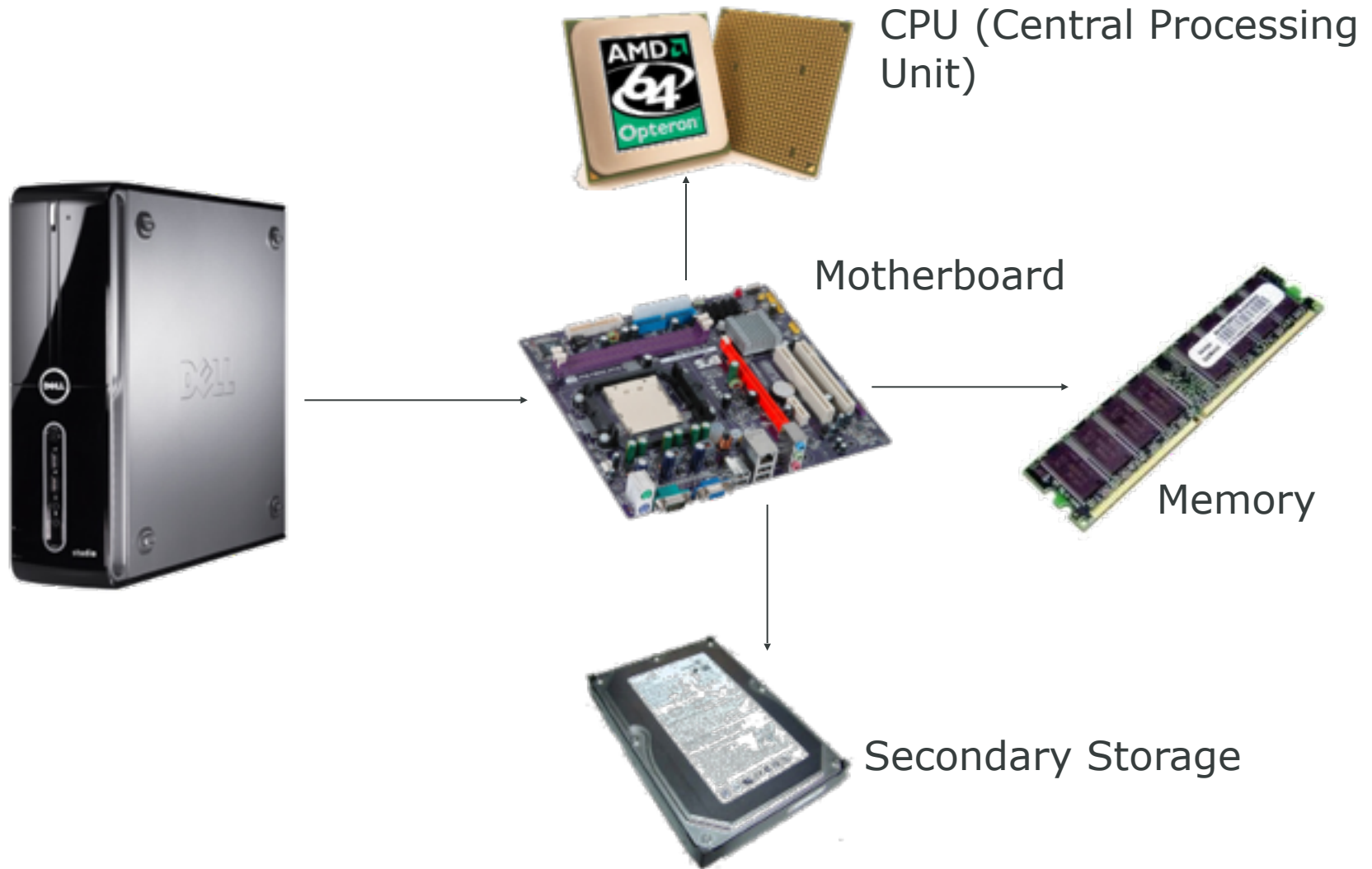
1. Tổng quan về máy tính
2. Các khái niệm cơ bản về lập trình:
3. Các ngôn ngữ lập trình
4. Giới thiệu bước đầu về ngôn ngữ C++, chương trình C++ và công cụ
5. Một số ví dụ minh họa về chương trình C++ và chạy thử.
6. Một số quy tắc cần nhớ khi viết chương trình



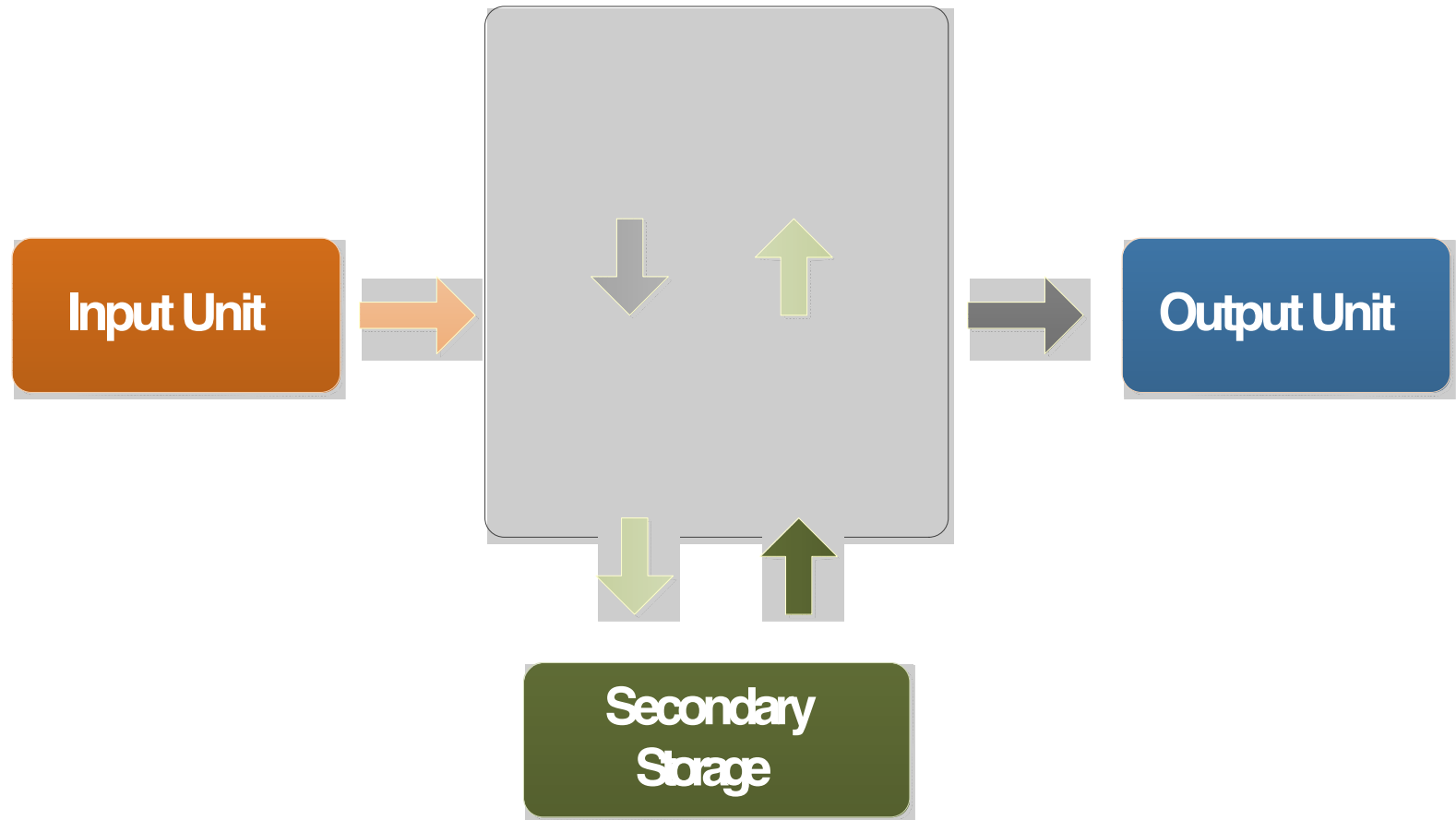
1. Tổng quan về máy tính

- a. Máy tính là gì?
- b. Cấu trúc tổng quan của máy tính và các thiết bị ngoại vi
- c. Phần mềm máy tính
- d. Thông tin được biểu diễn và đo lường như thế nào? Làm sao máy tính xử lý được thông tin? ...

1.a) Máy tính (Computer) là gì ?



1.b) Cấu trúc tổng quan của máy tính và các thiết bị ngoại vi



1.b) Cấu trúc tổng quan của máy tính và các thiết bị ngoại vi

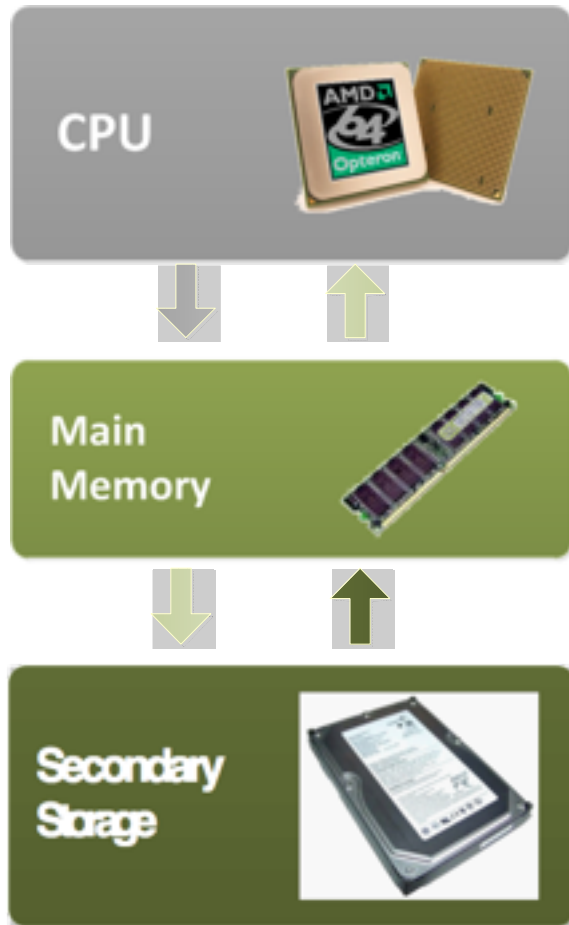


Thiết bị nhập (Input Unit): nhận dữ liệu từ người dùng hoặc từ các chương trình khác



Đơn vị xuất (Output Unit): hiển thị kết quả cho người dùng hoặc cho các chương trình khác.

1.b) Cấu trúc tổng quan của máy tính và các thiết bị ngoại vi



CPU (Central Processing Unit): Đọc các chỉ thị từ bộ nhớ chính và thực thi các chỉ thị.

Main Memory: lưu trữ các chương trình đang thực thi và các dữ liệu liên quan

Secondary Storage: lưu trữ chương trình và các tập tin chứa dữ liệu.

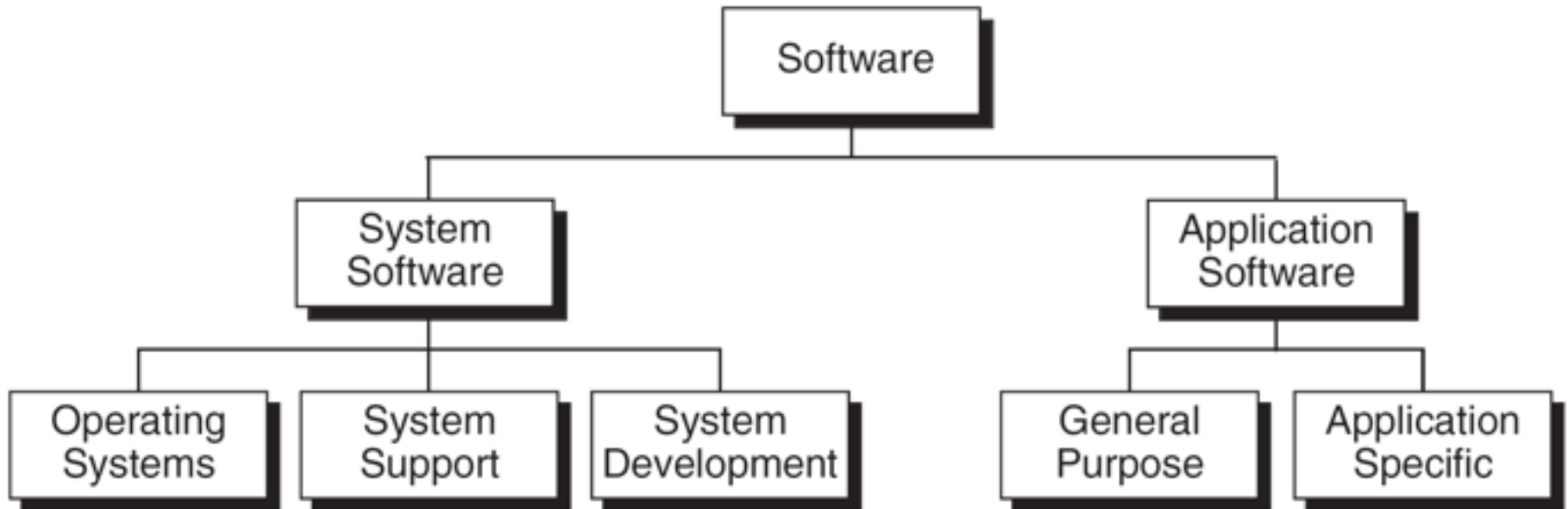
1.c) Phần mềm máy tính



- Phần mềm máy tính (Computer Software): là một tập hợp những câu lệnh hoặc chỉ thị (Instruction) được viết bằng một hoặc nhiều ngôn ngữ lập trình theo một trật tự xác định, và các dữ liệu hay tài liệu liên quan nhằm tự động thực hiện một số nhiệm vụ hay chức năng hoặc giải quyết một vấn đề cụ thể nào đó.
- Phần mềm thực hiện các chức năng của nó bằng cách gửi các chỉ thị trực tiếp đến phần cứng (hay phần cứng máy tính, Computer Hardware) hoặc bằng cách cung cấp dữ liệu để phục vụ các chương trình hay phần mềm khác.
- Phần mềm là một khái niệm trừu tượng, nó khác với phần cứng ở chỗ là "phần mềm không thể sờ hay đụng vào", và nó cần phải có phần cứng mới có thể thực thi được.

1.c) Phần mềm máy tính

- Các loại phần mềm



1.d) Thông tin



- Trong máy tính, các thông tin được biểu diễn bằng hệ đếm nhị phân còn gọi là bit.
- Các đơn vị đo lường dữ liệu

Table 1: Data Measurement Units

Unit	Abbreviation	Decimal Value	Binary Value	Decimal Size
bit	b	0 or 1	0 or 1	1/8 of a byte
byte	B	8 bits	8 bits	1 byte
kilobyte	KB	1,000 ¹ bytes	1,024 ¹ bytes	1,000 bytes
megabyte	MB	1,000 ² bytes	1,024 ² bytes	1,000,000 bytes
gigabyte	GB	1,000 ³ bytes	1,024 ³ bytes	1,000,000,000 bytes
terabyte	TB	1,000 ⁴ bytes	1,024 ⁴ bytes	1,000,000,000,000 bytes
petabyte	PB	1,000 ⁵ bytes	1,024 ⁵ bytes	1,000,000,000,000,000 bytes
exabyte	EB	1,000 ⁶ bytes	1,024 ⁶ bytes	1,000,000,000,000,000,000 bytes
zettabyte	ZB	1,000 ⁷ bytes	1,024 ⁷ bytes	1,000,000,000,000,000,000,000 bytes
yottabyte	YB	1,000 ⁸ bytes	1,024 ⁸ bytes	1,000,000,000,000,000,000,000,000 bytes

1.d) Thông tin



Quá trình xử lý thông tin trên máy tính

- Nhận thông tin (Receive input): thu nhận thông tin từ thế giới bên ngoài vào máy tính. Thực chất đây là quá trình chuyển đổi các thông tin ở thế giới thực sang dạng biểu diễn thông tin trong máy tính thông qua các thiết bị đầu vào.
- Xử lý thông tin (process information): biến đổi, phân tích, tổng hợp, tra cứu... những thông tin ban đầu để có được những thông tin mong muốn.
- Xuất thông tin (produce output) : đưa các thông tin kết quả (đã qua xử lý) ra trở lại thế giới bên ngoài. Đây là quá trình ngược lại với quá trình ban đầu, máy tính sẽ chuyển đổi các thông tin trong máy tính sang dạng thông tin ở thế giới thực thông qua các thiết bị đầu ra.
- Lưu trữ thông tin (store information): ghi nhớ lại các thông tin đã được ghi nhận để có thể đem ra sử dụng trong những lần xử lý về sau.



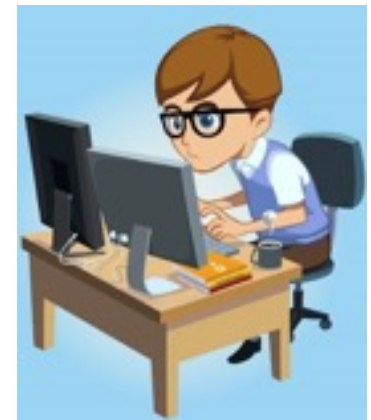
2. Các khái niệm cơ bản về lập trình

- a. Lập trình máy tính, lập trình viên
- b. Chương trình máy tính, mã nguồn, mã máy
- c. Ngôn ngữ lập trình.
- d. Chương trình dịch: Trình biên dịch, trình thông dịch, ...

2.a) Lập trình máy tính, lập trình viên




- Lập trình máy tính (programming)
 - Là kỹ thuật cài đặt một hoặc nhiều thuật toán trừu tượng có liên quan với nhau bằng một hoặc nhiều ngôn ngữ lập trình để tạo ra một chương trình máy tính.
- Lập trình viên (programmer)
 - Lập trình viên (người lập trình hay thảo chương viên điện toán) là người viết ra các chương trình máy tính.



2.a) Lập trình máy tính, lập trình viên

- The first computer programmer



WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item

Not logged in - Talk - Contributions - Create account - Log in

Article **Talk** Read View source View history Search

Ada Lovelace

From Wikipedia, the free encyclopedia


Augusta Ada King-Noel, Countess of Lovelace (née **Byron**; 10 December 1815 – 27 November 1852) was an English mathematician and writer, chiefly known for her work on **Charles Babbage's** early mechanical general-purpose computer, the **Analytical Engine**. Her notes on the engine include what is recognised as the first **algorithm** intended to be carried out by a machine. As a result, she is often regarded as the first computer **programmer**.^{[1][2][3]}

Ada Lovelace was the only legitimate child of the poet **George, Lord Byron** and his wife **Anne Isabella Milbanke** ("Annabella"), Lady Wentworth.^[4] All **Byron's other children** were born out of wedlock to other women.^[5] Byron separated from his wife a month after Ada was born and left England forever four months later, eventually dying of disease in the **Greek War of Independence** when Ada was eight years old. Her mother remained bitter towards Lord Byron and promoted Ada's interest in mathematics and logic in an effort to prevent her from developing what she saw as the insanity seen in her father, but Ada remained interested in him despite this (and was, upon her eventual death, buried next to him at her request). Often ill, she spent most of her childhood sick. Ada married **William King** in 1835. King was made Earl of Lovelace in 1838, and she became Countess of Lovelace.

Her educational and social exploits brought her into contact with scientists such as **Andrew Crosse**, **Sir David Brewster**, **Charles Wheatstone**, **Michael Faraday** and the author **Charles Dickens**, which she used to further her education. Ada described her approach as "poetical science"^[6] and herself as an "Analyst (& Metaphysician)".^[7]

As a teenager, her mathematical talents led her to an ongoing working relationship and friendship with fellow British mathematician **Charles Babbage**, also known as 'the father of computers', and in particular, Babbage's work on the Analytical Engine. Lovelace first met him in June 1833, through their mutual friend, and her private tutor, **Mary Somerville**. Between 1842 and 1843, Ada translated an article by Italian military engineer **Luigi Menabrea** on the engine, which she supplemented with an elaborate set of notes, simply

Ada, Countess of Lovelace



Ada, Countess of Lovelace, 1840

Born	The Hon. Augusta Ada Byron 10 December 1815 London, England
-------------	---

2.b) Chương trình máy tính, mã nguồn, mã máy

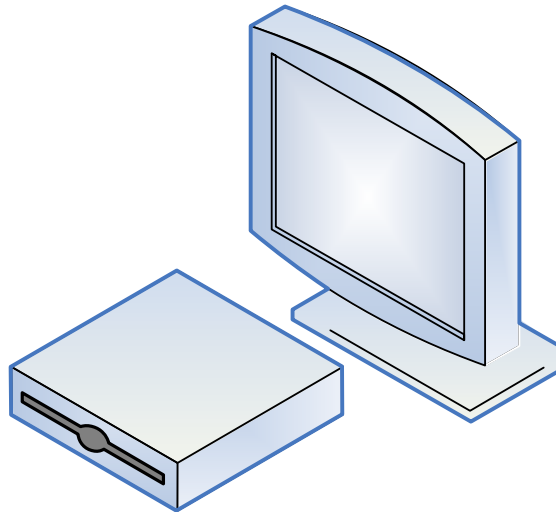


Chương trình máy tính:

- Là một danh sách các câu lệnh hay chỉ thị để máy thực hiện một chức năng nào đó.

Timer Recording

1. Turn on
2. Set Channel to **ch01**
3. Set Date to **1/5/2009**
4. Set Time to **3:00am**
5. Confirm setting



Program X

```
int x=10;  
  
int y=11;  
  
y+=x;  
  
System.out.println(y);  
  
System.out.println(x);
```

2.b) Chương trình máy tính, mã nguồn, mã máy



Chương trình máy tính:

- Là một cách giao tiếp với máy tính.
- Được viết bằng ngôn ngữ lập trình.



Con người giao tiếp với nhau



Con người giao tiếp với máy tính

2.b) Chương trình máy tính, mã nguồn, mã máy



Mã nguồn, mã máy

Source code

```
if (!PREACTION(gm)) {  
    void* mem;  
    size_t nb;  
    if (bytes <= MAX_SMALL_REQUEST)  
        bindex_t idx;  
        binmap_t smallbits;  
        nb = (bytes < MIN_REQUEST)? M  
        idx = small_index(nb);  
        smallbits = gm->smallmap >> i;  
  
        if ((smallbits & 0x3U) != 0)  
            mchunkptr b, p;  
            idx += ~smallbits & 1;  
            b = smallbin_at(gm, idx);  
            p = b->fd;  
            assert(chunksize(p) == smal  
            unlink_first_small_chunk(gm
```



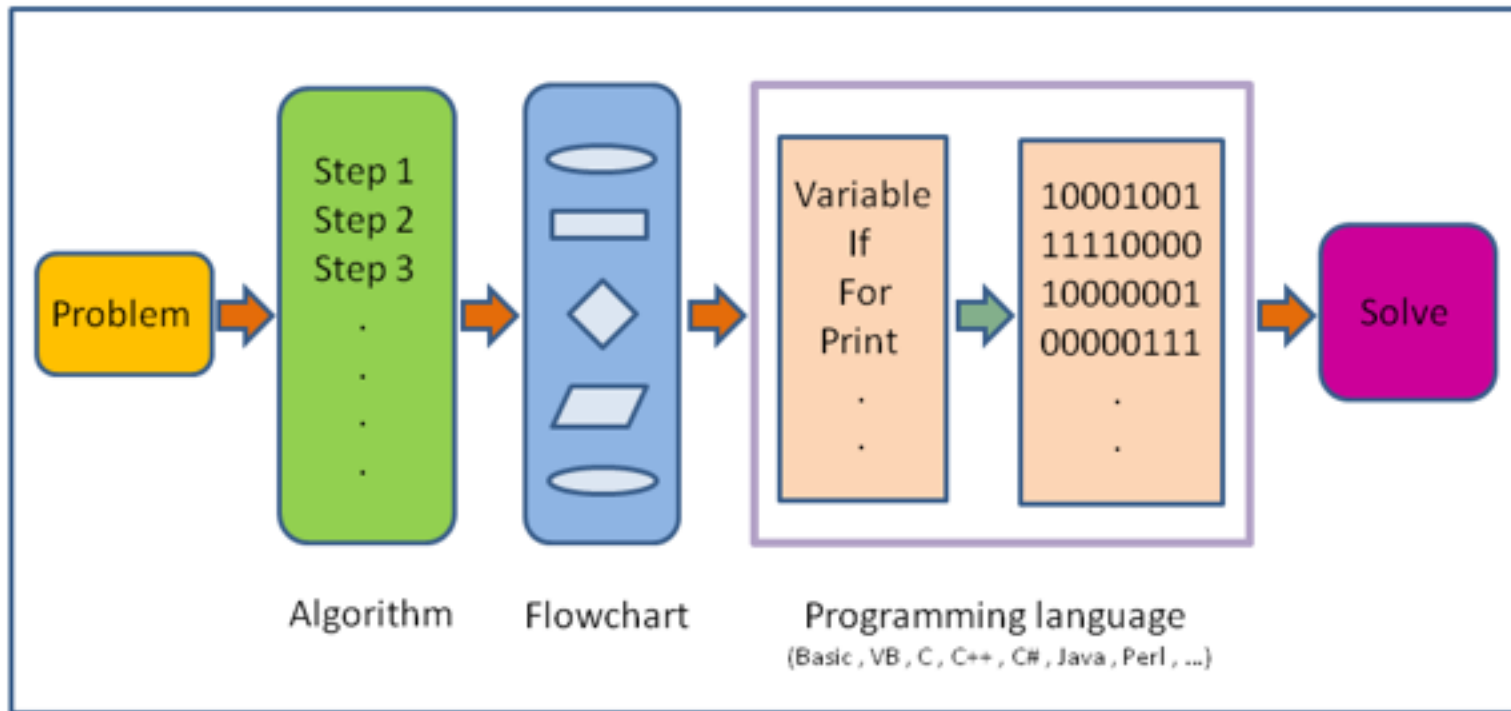
compiler
reordering

Machine code

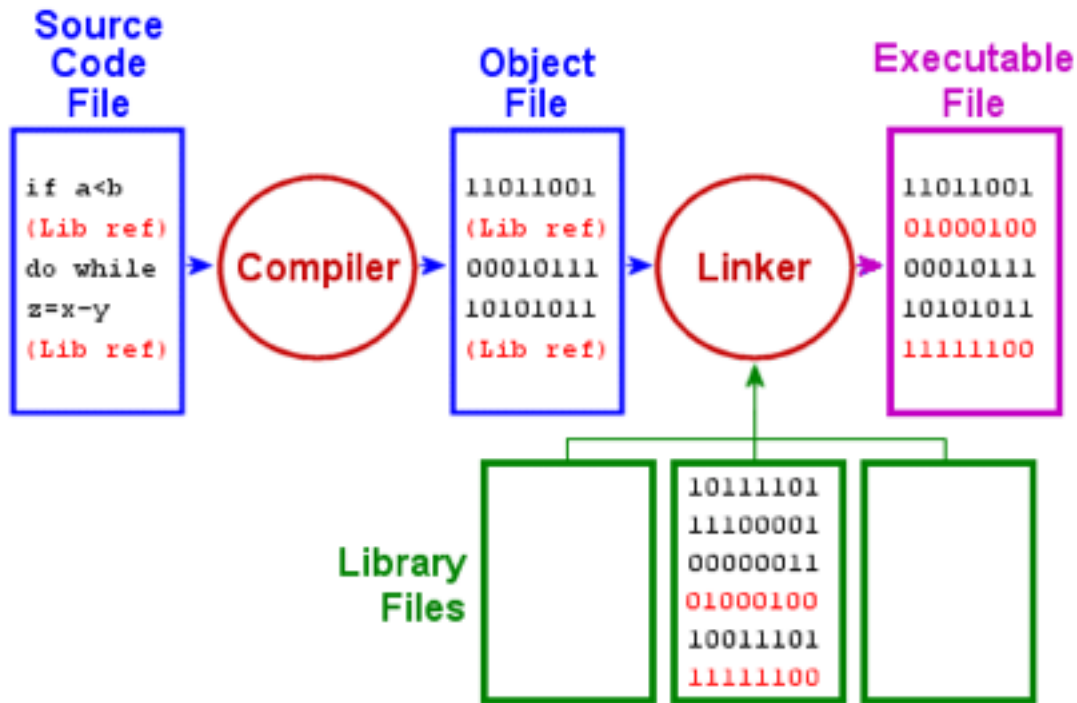
```
cmp        esi,0F4h  
ja         dmalloc+1AEh (777CEh)  
cmp        esi,0Bh  
jae        dmalloc+20h (77640h)  
mov        esi,10h  
jmp        dmalloc+26h (77646h)  
add        esi,0Bh  
and        esi,0FFFFFFF8h  
mov        eax,dword ptr [__fnode+20h]  
mov        edi,esi  
shr        edi,3  
mov        ecx,edi  
shr        eax,cl  
test       al,3  
je         dmalloc+9Ah (776BAh)  
not        eax  
and        eax,1  
add        edi,eax  
mov        esi,dword ptr __fnode+50h
```

2.c) Ngôn ngữ lập trình (Programming Languages)

- Là ngôn ngữ dùng để viết các chương trình cho máy tính. Cũng như các ngôn ngữ thông thường, NNLT cũng có từ vựng, cú pháp và ngữ nghĩa.



2.d) Chương trình dịch



Nguồn: <http://www.aboutdebian.com/compile.htm>



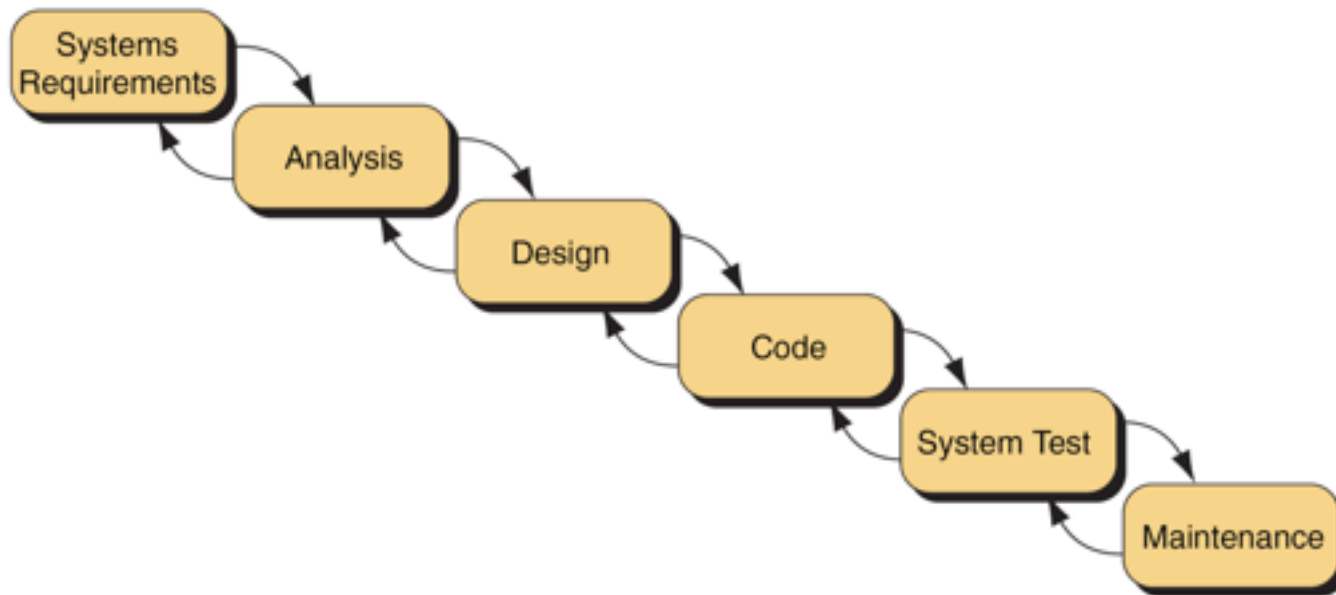
3. Các ngôn ngữ lập trình

- a. Vai trò của NNLT đối với công nghệ lập trình.
- b. Ngôn ngữ lập trình cấp thấp
- c. Ngôn ngữ lập trình cấp cao
- d. Một vài ngôn ngữ lập trình thông dụng



3.a) Vai trò của ngôn ngữ lập trình

- Mô hình phát triển phần mềm water fall



NNLT đóng vai trò là một **công cụ** giúp con người thực hiện bước **cài đặt (code)** chương trình.

3.b) Ngôn ngữ cấp thấp



- Ngôn ngữ máy (machine language) là các chỉ thị dưới dạng nhị phân, can thiệp trực tiếp vào trong các mạch điện tử.
- Chương trình được viết bằng ngôn ngữ máy thì có thể được thực hiện ngay không cần qua bước trung gian nào.



Machine Language

Language directly
understood by the
computer

binary code



3.b) Ngôn ngữ cấp thấp

• VD:

1		00000000	00000100	0000000000000000
2	01011110	00001100	11000010	0000000000000010
3		11101111	00010110	00000000000000101
4		11101111	10011110	00000000000001011
5	11111000	10101101	11011111	00000000000010010
6		01100010	11011111	00000000000010101
7	11101111	00000010	11111011	00000000000010111
8	11110100	10101101	11011111	00000000000011110
9	00000011	10100010	11011111	00000000000100001
10	11101111	00000010	11111011	00000000000100100
11	01111110	11110100	10101101	
12	11111000	10101110	11000101	00000000000101011
13	00000110	10100010	11111011	00000000000110001
14	11101111	00000010	11111011	00000000000110100
15		01010000	11010100	00000000000111011
16			00000100	00000000000111101

Tuy nhiên chương trình viết bằng ngôn ngữ máy dễ sai sót, cồng kềnh và khó đọc, khó hiểu vì toàn những con số 0 và 1.

3.b) Ngôn ngữ cấp thấp



- Hợp ngữ (assembly language) được thiết kế để máy tính trở nên thân thiện hơn với người sử dụng.
- Các câu lệnh bao gồm hai phần: phần mã lệnh (viết tựa tiếng Anh) chỉ phép toán cần thực hiện và phần tên biến chỉ địa chỉ chứa toán hạng của phép toán đó.



Machine Language

Language directly understood by the computer

binary code

Symbolic Language

English-like abbreviations representing elementary computer operations

assembly language

3.b) Ngôn ngữ cấp thấp



- VD

1	entry	main, ^m<r2>
2	subl2	#12, sp
3	jsb	C\$MAIN_ARGS
4	movab	\$CHAR_STRING_CON
5		
6	pushal	-8(fp)
7	pushal	(r2)
8	calls	#2, SCANF
9	pushal	-12(fp)
10	pushal	3(r2)
11	calls	#2, SCANF
12	mull3	-8(fp), -12(fp), -
13	pusha	6(r2)
14	calls	#2, PRINTF
15	clrl	r0
16	ret	

Để máy thực hiện được một chương trình viết bằng hợp ngữ thì chương trình đó phải được dịch sang ngôn ngữ máy. Công cụ thực hiện việc dịch đó được gọi là Assembler

3.c) Ngôn ngữ cấp cao



- Ngôn ngữ cấp cao (High level language): là ngôn ngữ được tạo ra và phát triển nhằm phản ánh cách thức người lập trình nghĩ và làm.
- Ngôn ngữ cấp cao rất gần với ngôn ngữ con người (Anh ngữ) nhưng chính xác như ngôn ngữ toán học.



Machine Language

Language directly understood by the computer

binary code

Symbolic Language

English-like abbreviations representing elementary computer operations

assembly language

High-level Language

Close to human language.

Example: $a = a + b$

[add values of a and b , and store the result in a , replacing the previous value]

C, C++, Java, Basic

3.d) Một vài ngôn ngữ lập trình thông dụng



ActionScript Ada **ASP.NET** Assembler Basic
c **C++** C# Cobol Cobra CODE ColdFusion
Delphi Eiffel Fortran FoxPro GPSS **HTML** J#
J++ **Java** JavaScript **JSP** LISP Logo LUA
MEL Modula-2 Miranda Objective-C **Perl** **PHP**
Prolog **Python** **SQL** Visual Basic Visual
Basic.NET VBA Visual-FoxPro



4. Giới thiệu sơ lược C++

- a. Giới thiệu tổng quan về ngôn ngữ C++
- b. Giới thiệu sơ lược về cấu trúc chương trình
- c. Giới thiệu môi trường, công cụ hỗ trợ việc lập trình Visual Studio C++
- d. Quy trình tổng quát viết, dịch, chạy thử chương trình



4.a) Giới thiệu tổng quan về ngôn ngữ C++

- C++ là một ngôn ngữ lập trình hỗ trợ lập trình hướng đối tượng, lập trình thủ tục.
- C++ được coi như là ngôn ngữ bậc trung (middle-level), khi nó kết hợp các đặc điểm và tính năng của ngôn ngữ bậc cao và bậc thấp.
- C++ được phát triển bởi Bjarne Stroustrup năm 1979 tại Bell Labs ở Murray Hill, New Jersey, như là một bản nâng cao của ngôn ngữ C và với tên gọi đầu tiên là “C với các Lớp”, nhưng sau đó được đổi tên thành C++ vào năm 1983.
- C++ là một Superset của C, và bất kỳ chương trình C nào cũng là một chương trình C++.



4.a) Giới thiệu tổng quan về ngôn ngữ C++

- Các từ khóa

<code>asm</code>	<code>auto</code>	<code>bool</code>	<code>break</code>
<code>case</code>	<code>catch</code>	<code>char</code>	<code>class</code>
<code>const</code>	<code>const_cast</code>	<code>continue</code>	<code>default</code>
<code>delete</code>	<code>else</code>	<code>extern</code>	<code>do</code>
<code>enum</code>	<code>false</code>	<code>double</code>	<code>explicit</code>
<code>float</code>	<code>dynamic_cast</code>	<code>export</code>	<code>for</code>
<code>friend</code>	<code>goto</code>	<code>if</code>	<code>inline</code>
<code>int</code>	<code>long</code>	<code>mutable</code>	<code>namespace</code>
<code>new</code>	<code>operator</code>	<code>private</code>	<code>protected</code>
<code>public</code>	<code>register</code>	<code>reinterpret_cast</code>	<code>return</code>
<code>short</code>	<code>signed</code>	<code>sizeof</code>	<code>static</code>
<code>static_cast</code>	<code>struct</code>	<code>switch</code>	<code>template</code>
<code>this</code>	<code>throw</code>	<code>true</code>	<code>try</code>
<code>typedef</code>	<code>typeid</code>	<code>typename</code>	<code>union</code>
<code>unsigned</code>	<code>using</code>	<code>virtual</code>	<code>void</code>
<code>volatile</code>	<code>wchar_t</code>	<code>while</code>	



4.a) Giới thiệu tổng quan về ngôn ngữ C++

- Các kiểu dữ liệu

Data Types		
Type	Length	Range
unsigned char	8 bits	0 to 255
char	8 bits	-128 to 127
enum	16 bits	-32,768 to 32,767
unsigned int	16 bits	0 to 65,535
short int	16 bits	-32,768 to 32,767
int	16 bits	-32,768 to 32,767
unsigned long	32 bits	0 to 4,294,967,295
long	32 bits	-2,147,483,648 to 2,147,483,647
float	32 bits	$3.4 * (10^{** -38})$ to $3.4 * (10^{** +38})$
double	64 bits	$1.7 * (10^{** -308})$ to $1.7 * (10^{** +308})$
long double	80 bits	$3.4 * (10^{** -4932})$ to $1.1 * (10^{** +4932})$



4.a) Giới thiệu tổng quan về ngôn ngữ C++

- Các toán tử

[]	()	.	->	++	--	&
*	+	-	~	!	sizeof	/
%	<<	>>	<	>	<=	>=
==	!=	^	!	&&	!!	?:
=	*=	/=	%=	+=	-=	<<=
>>=	&=	^=	!=	,	#	##

The following operators are specific to C++:

:: .* ->*

The operators # and ## are used only by the preprocessor.

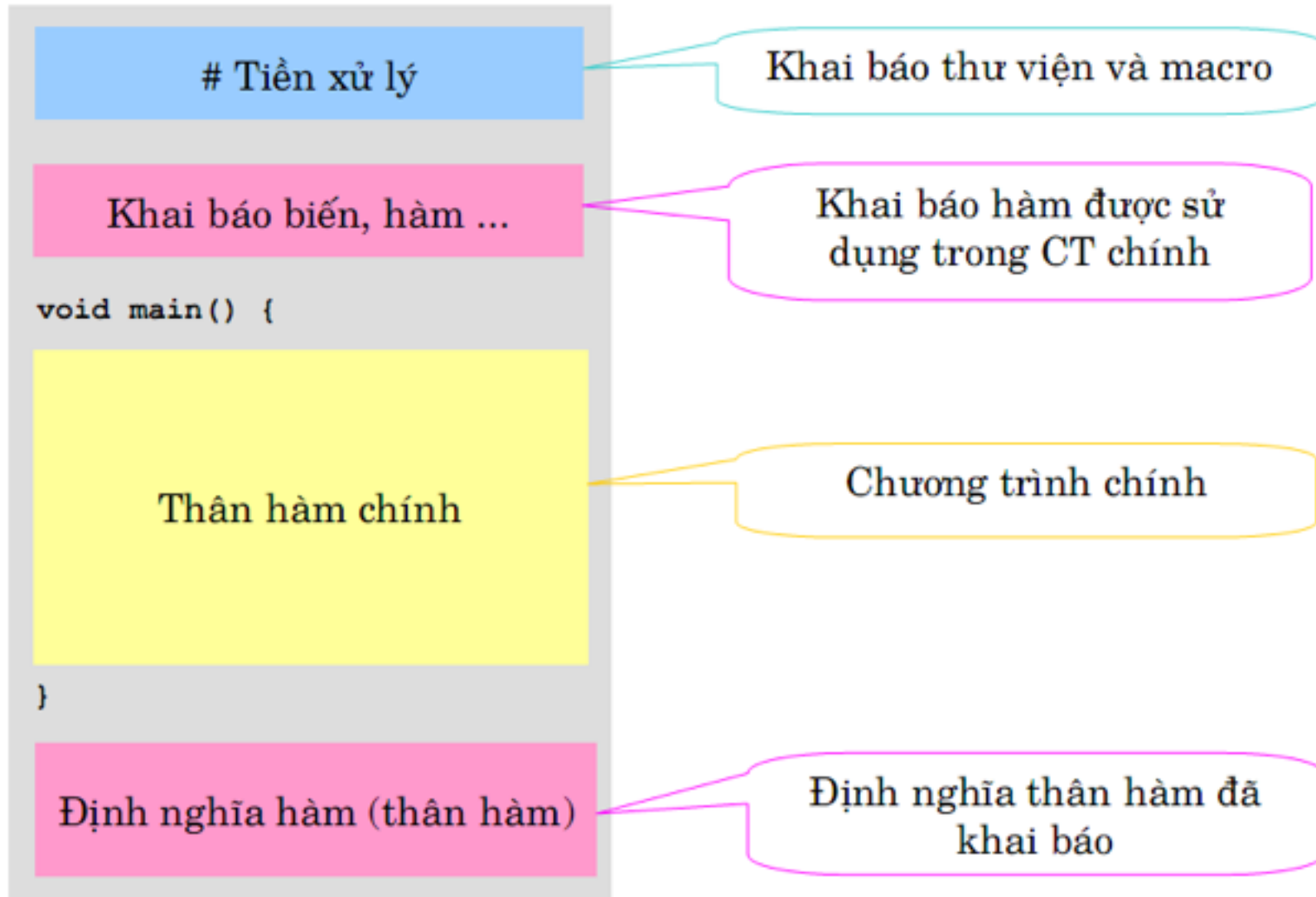
Depending on context, the same operator can have more than one meaning. For example, the ampersand (&) can be interpreted as

- a bitwise AND (A & B)
- an address operator (&A)
- in C++, a reference modifier

In the first case, the & is a **unary operator**; in the second, the & is a **binary operator**.



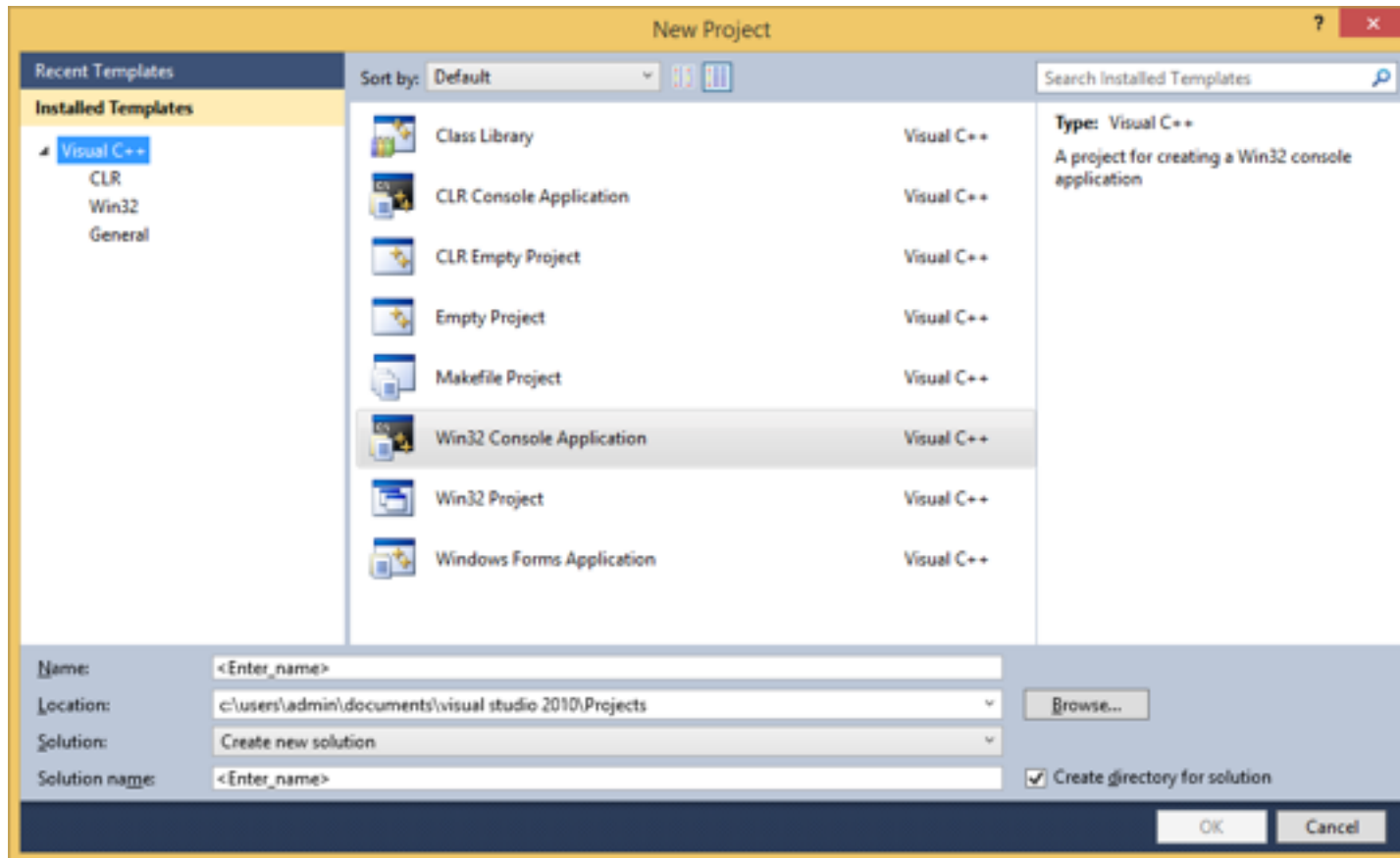
4.b) Giới thiệu sơ lược về cấu trúc chương trình





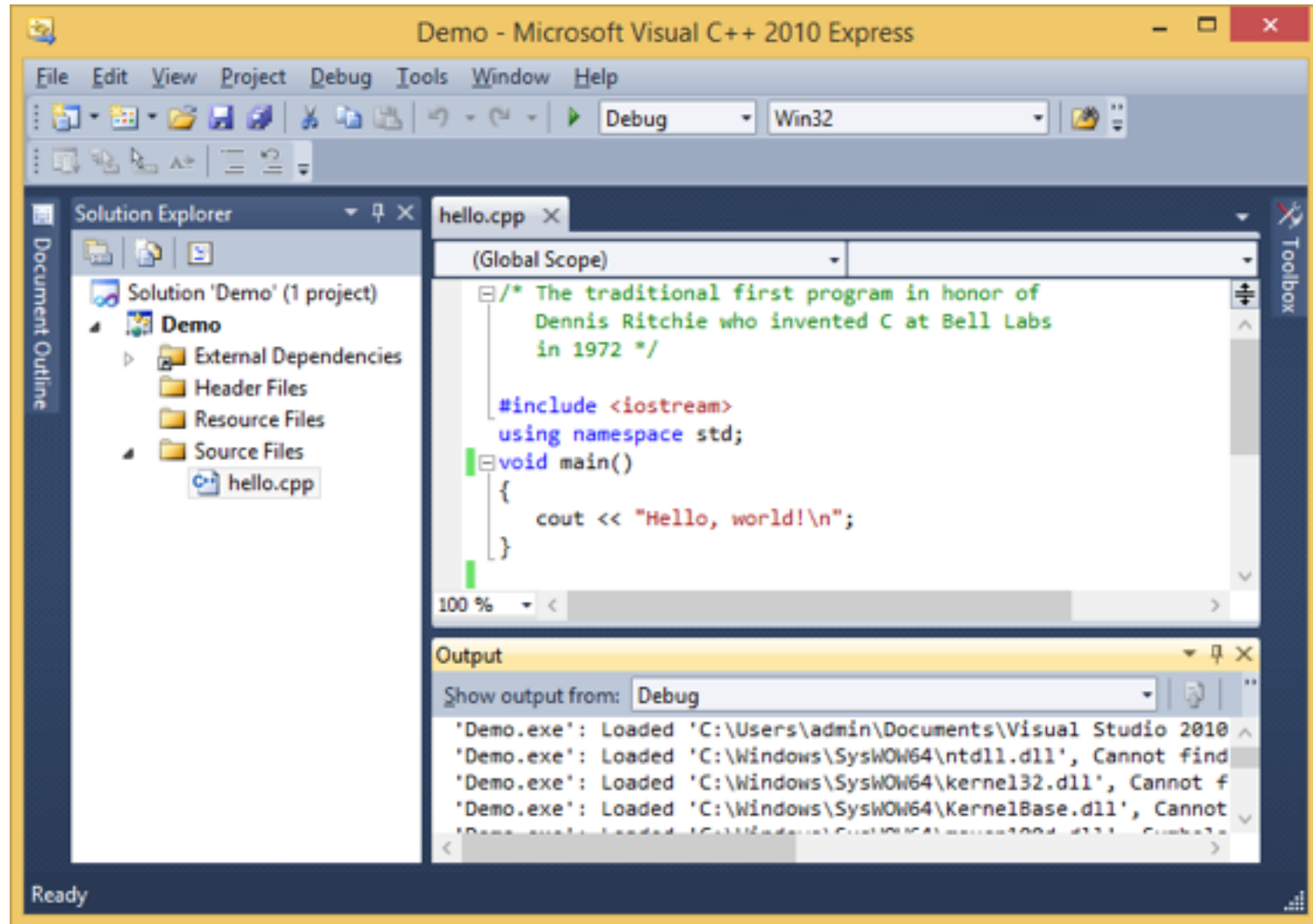
4.c) Giới thiệu sơ lược về Visual C++ 2010

- Tạo project



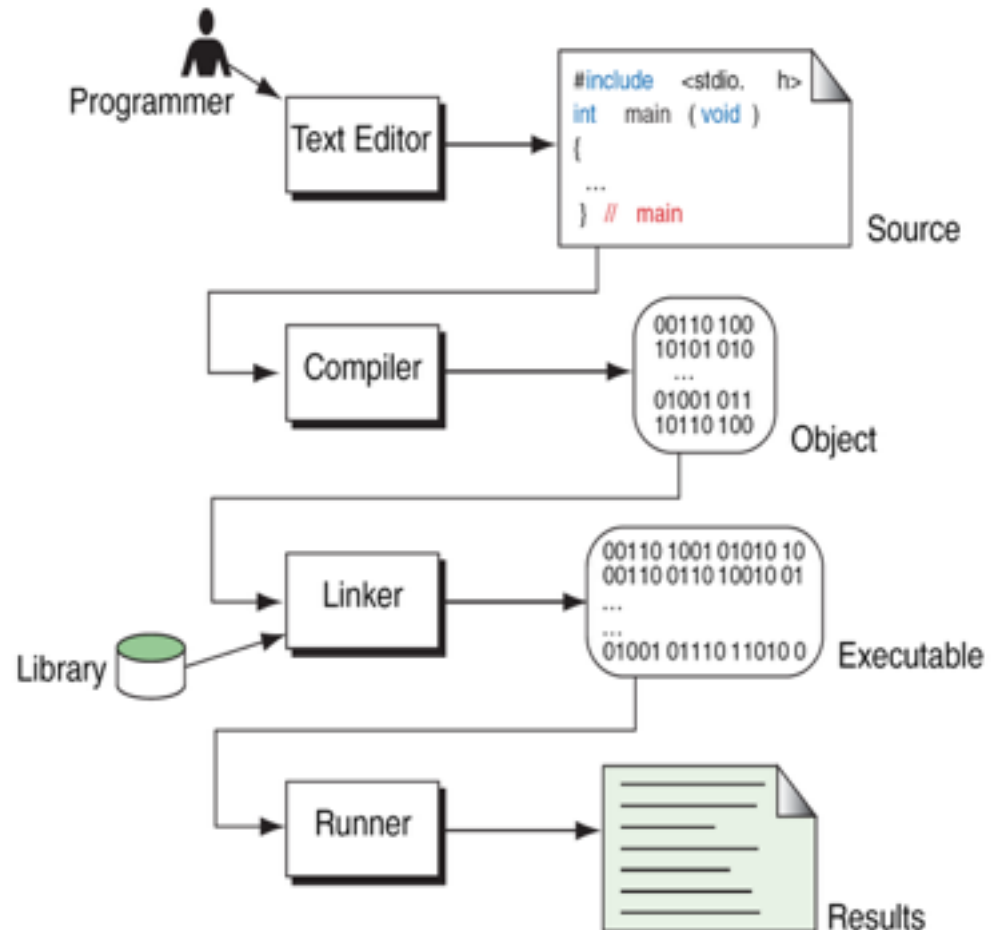
4.c) Giới thiệu sơ lược về Visual C++ 2010

- Cấu trúc tổ chức project



4.d) Quy trình tổng quát viết, dịch, chạy thử chương trình

- **Writing** source code as an C++ file.
 - e.g. **"hello.cpp"** file
- **Preprocessing**
 - **Processes** the source code for compilation.
- **Compilation**
 - Checks the **grammatical rules** (syntax).
 - Source code is converted to **object code** in machine language (e.g. **"hello.obj"** file)
- **Linking**
 - Combines object code and libraries to create an **executable** (e.g. **"hello.exe"** file).
 - Library: common functions (input, output, math, etc).



5. Ví dụ minh họa về chương trình C++ và chạy thử.

- a. Ví dụ 1: Nhập xuất đơn giản như “Hello World” (code: hello.cpp)
- b. Ví dụ 2: Chương trình có nhập xuất dữ liệu và tính toán xử lý đơn giản như “Nhập độ dài 2 cạnh của hình chữ nhật, xuất diện tích của hình” (code: tinhdientich.cpp)
- c. Ví dụ 3: Chương trình phức tạp hơn, có sử dụng vòng lặp: kiểm tra một số nguyên n có phải là số nguyên tố không (code: kiemtrasnt.cpp)

5.a) Ví dụ 1



Xuất đơn giản như “Hello World”

```
/* The traditional first program in honor
of
    Dennis Ritchie who invented C at Bell
    Labs
    in 1972 */

#include <iostream>
using namespace std;
void main()
{
    cout << "Hello, world!\n";
}
```

5.b) Ví dụ 2



Chương trình có nhập xuất dữ liệu và tính toán xử lý đơn giản như “Nhập độ dài 2 cạnh của hình chữ nhật, xuất diện tích của hình”

```
/* Minh hoa chương trình tính diện tích hình chu nhật
*/

#include <iostream>
using namespace std;
void main()
{
    int chieu_dai,chieu_rong;
    cout<< "Nhap chieu dai = ";
    cin>>chieu_dai;
    cout<< "Nhap chieu rong = ";
    cin>>chieu_rong;
    // Tính diện tích hình chu nhật
    int dien_tich = chieu_dai*chieu_rong;
    // In kết quả ra màn hình
    cout << "Diện tích HCN = "<<dien_tich;
}
```

5.c) Ví dụ 3



Chương trình phức tạp hơn, có sử dụng vòng lặp: kiểm tra một số nguyên n có phải là số nguyên tố không

```
/* Chương trình minh họa kiểm tra số nguyên tố */

#include <iostream>
using namespace std;
// Hàm kiểm tra số nguyên n có phải là số nguyên tố
// (true) hay không (false).
bool kiemtralasonguyento(int n)
{
    if( n<2) return false;
    for(int i=2; i<n;i++)
        if(n%i==0) return false;

    return true;
}
void main()
{
    int n;
    cout<< "Nhập vào số nguyên n = ";
    cin>>n;
    if (kiemtralasonguyento(n)== true)
        cout<< "Số "<<n<<" là số nguyên tố !";
    else
        cout<< "Số "<<n<<" không phải là số nguyên tố !";
}
```


6. Một số quy tắc cần nhớ khi viết chương trình



- Chương trình nên được tách thành nhiều đơn thể (mô-đun), mỗi đơn thể thực hiện một công việc và càng độc lập với nhau.
- Cách trình bày chương trình càng nhất quán sẽ càng dễ đọc và dễ hiểu (định hướng về phong cách lập trình).
- Mỗi câu lệnh có thể viết trên một hay nhiều dòng nhưng phải được kết thúc bằng dấu ;
- Quy tắc viết lời giải thích, lời giải thích không có tác dụng với sự làm việc của chương trình trên máy tính, chỉ có tác dụng với người đọc
- Sử dụng các hàm chuẩn: sử dụng `#include`
- Hàm chính main

