Slot 01:
Introduction to PFC

-Languages and C Compilers
- First Program in C

Objectives

This chapter supplies basic concepts in computer programming. After studying this chapter, you should be able to:

- Define concepts: computer program, computer software, Information, data, fundamental Data Units, data address
- Explain how to make a good software?
- Understand steps to develop a software?
- Explain ways to represent data
- Answer why C is the first language selected
- Understand how a C program can be translated and execute
- Discuss about notable features of C
- Understand a C program structure

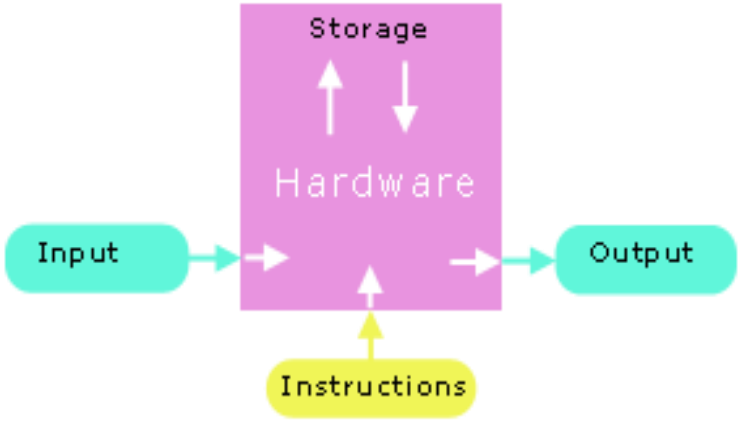
Contents

- Definitions: computer program, computer software
- How to make a good software?
- Steps to develop a software?
- Computer hardware.
- Information, Data, Fundamental Data Units
- Data Representation
- Addressing Information
- Program Instructions
- Languages
- Translate and execute a program
- Why C is the first language selected?
- Some notable features of C
- Structure of a simple C Program.

1.1- Definitions

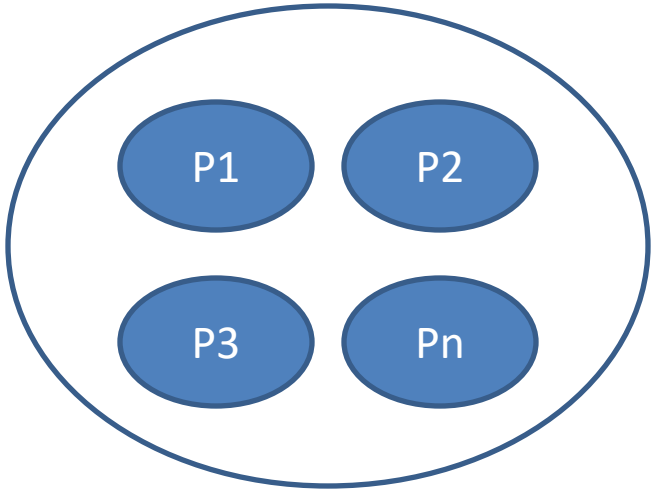
- Computer program:
  - A **simulation** of solution.
  - Is a set of instructions that computer hardware will execute

  ➔Increase **performance** of standard workflow (dòng công việc)



- Computer software:

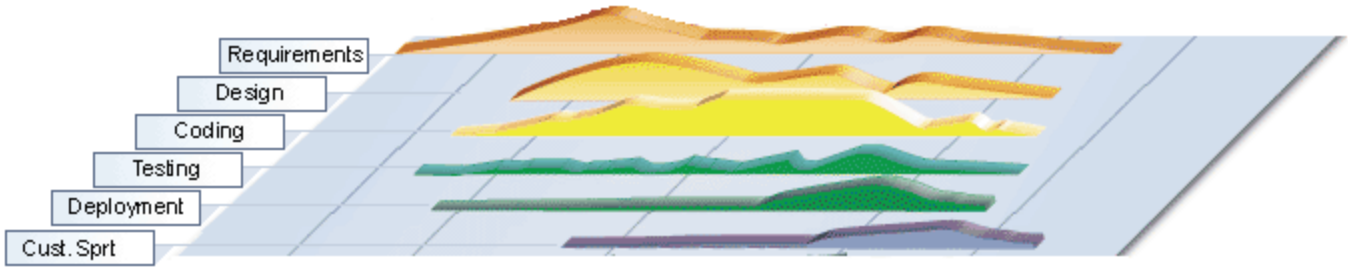
  - A set of related programs



1.2- How to make a good software?

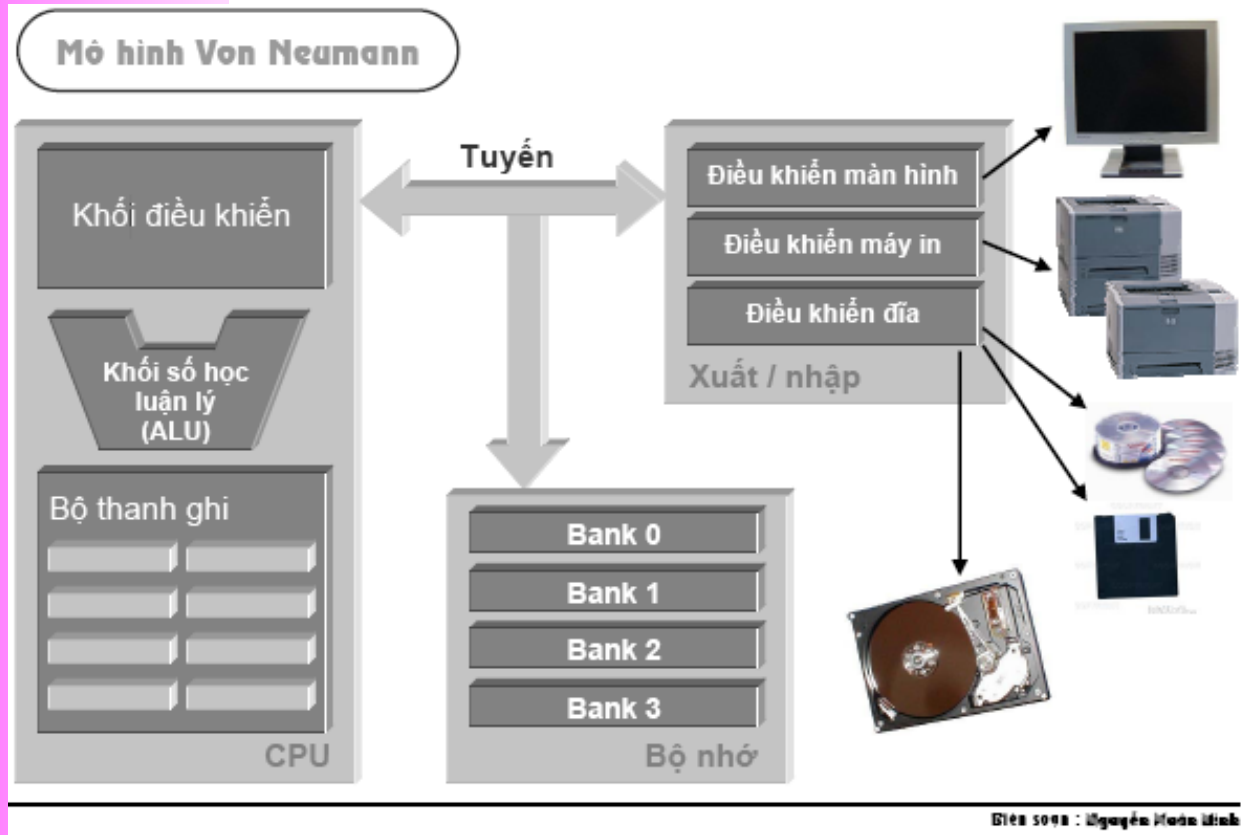
- Issues for a program/software:
  - Usability
    - robust and user-friendly interfaces
  - Correctness
    - comprehensive testing
  - Maintainability
    - Understandability
      - structured programming
      - internal documentation
    - Modifiability
      - standards compliance
  - Portability
    - standards compliance

1.3- Steps to develop a software



- Requirements
- Analysis
- Design
- Coding
- Testing
- Deploying
- Maintenance

1.4- Computer Hardware - Review



3 bước giúp đọc 1 ô nhớ:

(1) CPU áp đặt dữ liệu địa chỉ ô nhớ lên address bus

(2) CPU áp đặt tín hiệu "đọc" lên control bus.

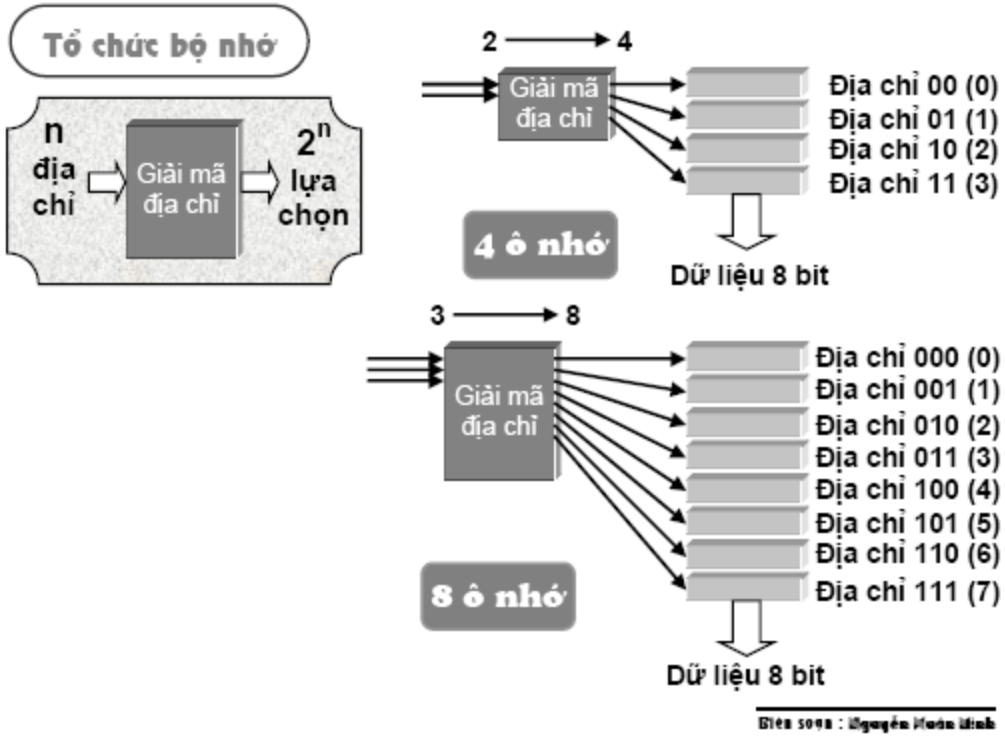
(3) Data trong ô nhớ sẽ theo data bus về 1 thanh ghi của CPU.

Bus	Used to
Address bus	Determine the IO peripherals, position of accessed memory.
Data bus	Transmit data
Control bus	Determine operation on peripherals, read peripheral 's states

1.4- Computer Hardware…

- The most expensive and fastest memory - registers - is reserved for the CPU.
  - CPU transfers information at less than 10 nanoseconds
  - primary memory transfers information at about 60 nanoseconds
  - a hard disk transfers information at about 12,000,000 nanoseconds
- CPU memory is volatile - the contents of the registers are lost as soon as power is turned off.

1.4- Computer Hardware…



- Primary Memory
  - Primary memory holds the information accessed by the CPU.
  - Primary memory is also volatile.
  - The popular term for primary memory is RAM (Random Access Memory).

1.4- Computer Hardware…

- Devices
  - Include I/O devices such as a keyboard, a monitor and a mouse…
  - Storage devices such as a floppy drive, a hard drive and a CD-ROM drive (secondary storage).
  - Each device interfaces with the system buses through a device controller.

1.4- Computer Hardware…

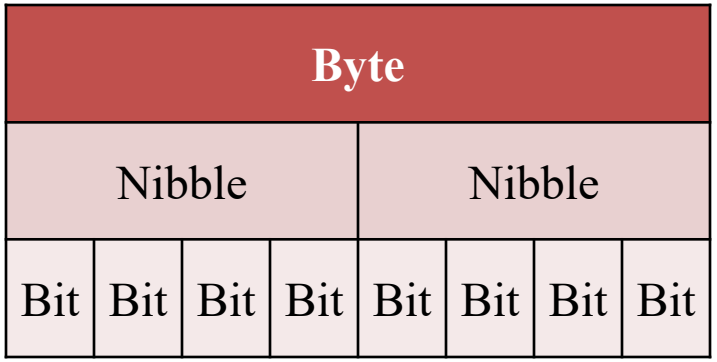
- Devices
  - Include I/O devices such as a keyboard, a monitor and a mouse…
  - Storage devices such as a floppy drive, a hard drive and a CD-ROM drive (secondary storage).
  - Each device interfaces with the system buses through a device controller.

1.5- Information/Data

- ***Data***: specific values that describe something
- ***Information***: Meaning of data
- Program information consists of instructions and data.
  - How is this information stored?
  - What does a program instruction look like?
  - How do we make program instructions readable?

1.6- Data Units

- John von Neumann selected binary (base 2) digits as the EDVAC's fundamental unit.
- The vast majority of modern computers process and store information in binary digits.
- We call a **binary digit** a bit.
- One nibble consists of 4 consecutive bits.
- The fundamental addressable unit of primary memory is the byte → One byte consists of 2 nibbles.



00000000 <- possibility 0
00000001 <- possibility 1
00000010 <- possibility 2
00000011 <- possibility 3
00000100 <- possibility 4
...
00111000 <- possibility 104
...
11111111 <- possibility 255

1.6- Data Units …

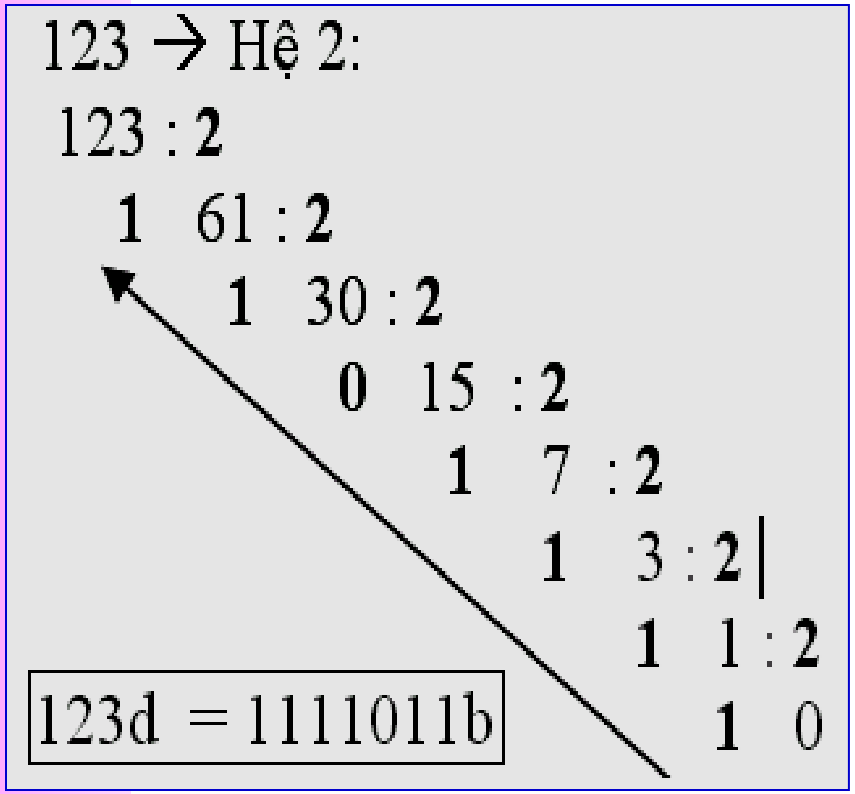
- The natural unit of the CPU is a word. A word is the size of the general registers - the unit of memory within the CPU.

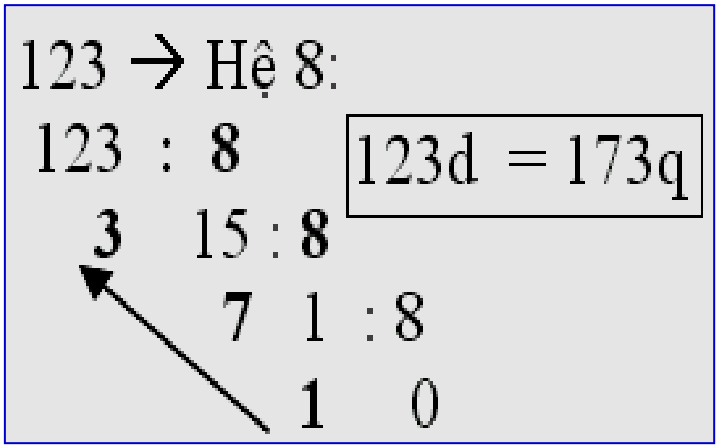
1.7- Data Representations

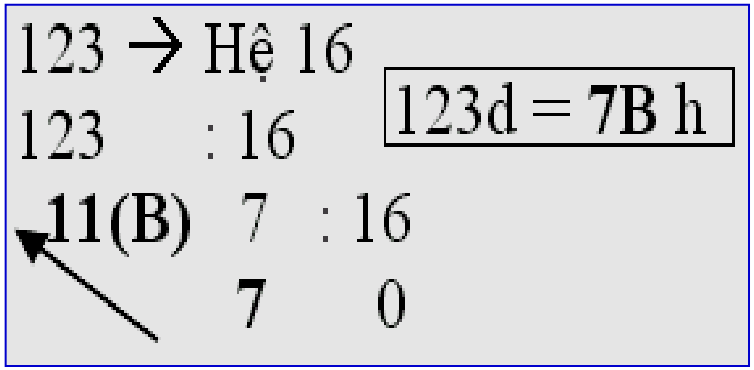
- 3 common number systems:
  - Decimal Representation
  - Hexadecimal Representation
    - Base 16: 0, 1, …, 9, A, B, C, D, E, F
    - Each hexadecimal digit represents 4 bits of information.
    - The 0x prefix identifies the number as a hexadecimal number: 0x5C
  - Octal Representation
    - Base 8: 0, 1, 2, .., 7
    - Set of 3 consecutive bits forms an octal digit
    - The prefix 0 identifies the number as an octal number: 031

Read by yourself

Data Representations: Conversion- A review



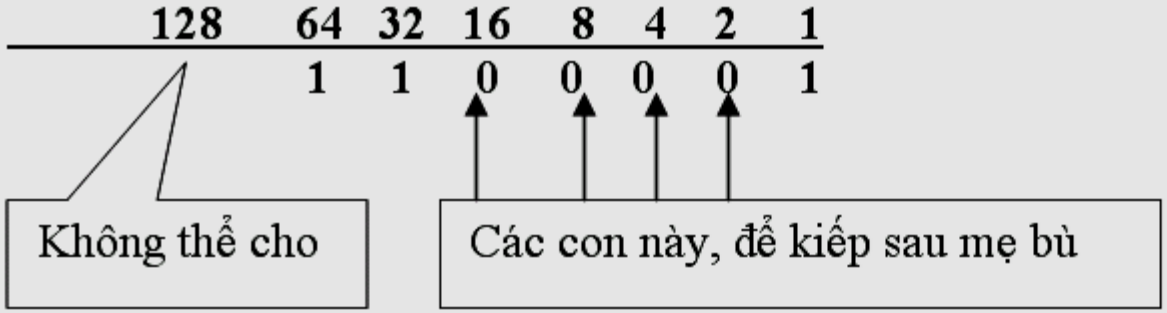




Read by yourself

Data Representations: Conversion: A review

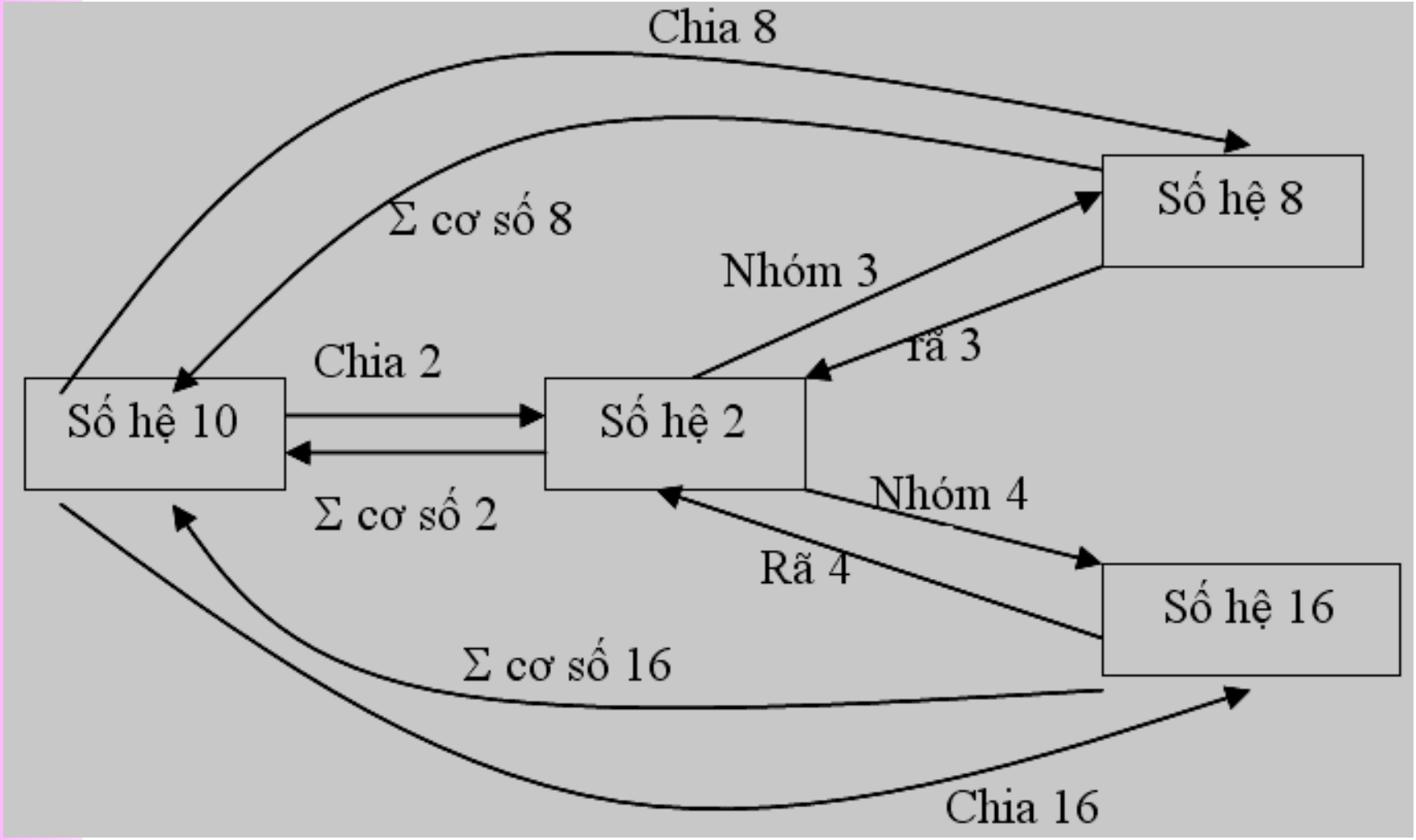
Mẹ có 97 cây vàng chia cho các con, cách phân phối: 1,2,4,8,16,... cây cho mỗi lần cho. Người con xin nhiều thì cho trước (vì anh ta cần để mở công ty), người xin ít cho sau (vì chỉ để nhậu). Khi cho được: Viết 1, khi không cho được: viết 0 (để kiếp sau mẹ bù)



Vậy : 97 = 1100001 b

Read by yourself

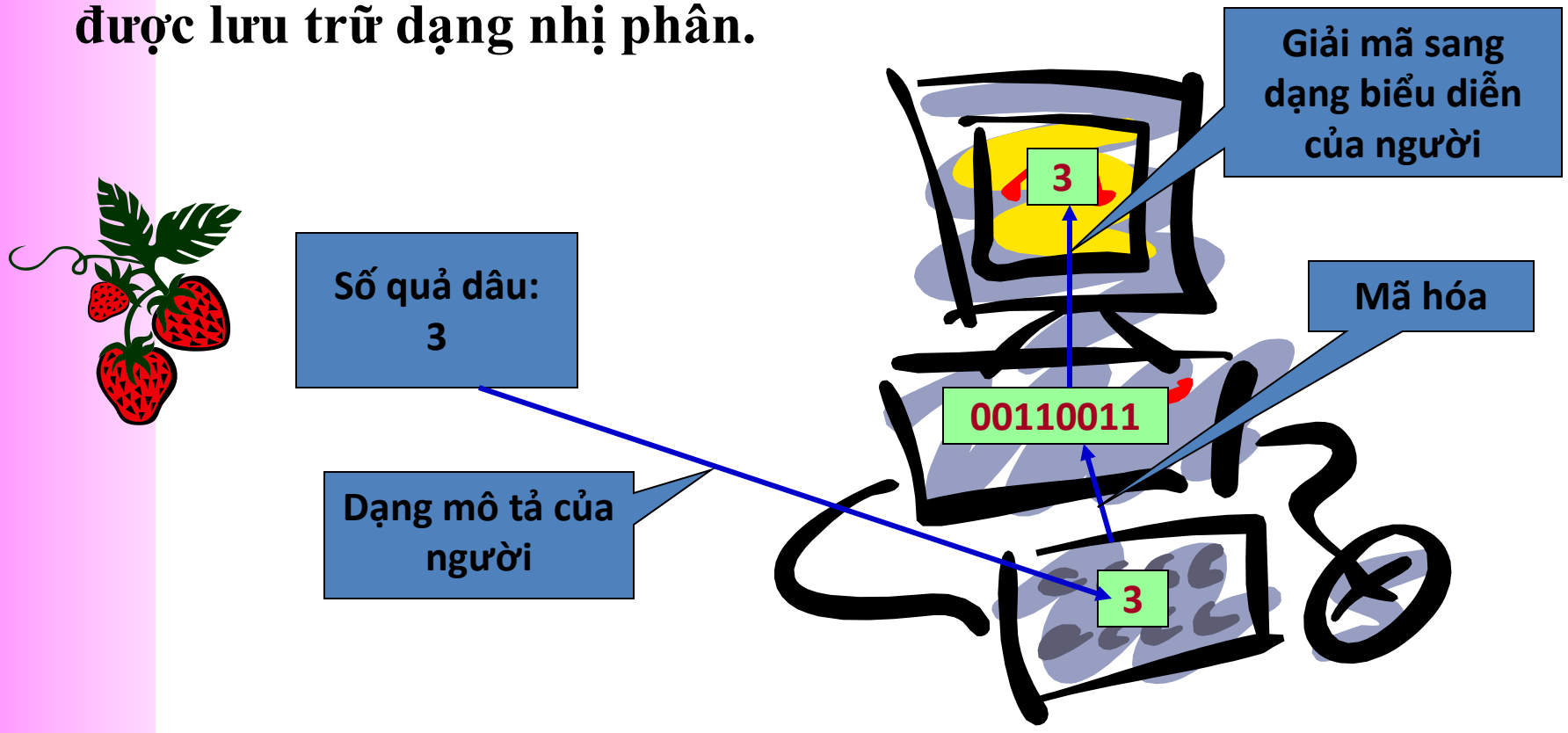
Data Representations: Conversion: A review



Data Representations: Conversion…

Read by yourself

- Máy tính là thiết bị nhị phân nên tất cả mọi dữ liệu đều được lưu trữ dạng nhị phân.



Data Representations: Conversion…

Fill the corresponding binary expansions of the following decimal number:

Decimal	4-bit Binary	Decimal	8-bit Binary	Decimal	16-bit Binary
9	1001	7	0000 0111	255	0000 0000 1111 1111
7		34		192	
2		125		188	
15		157		312	
12		162		517	
11		37		264	
6		66		543	
5		77		819	
8		88		1027	
13		99		2055	
14		109		63	

Read by and do yourself

Data Representations: Conversion…

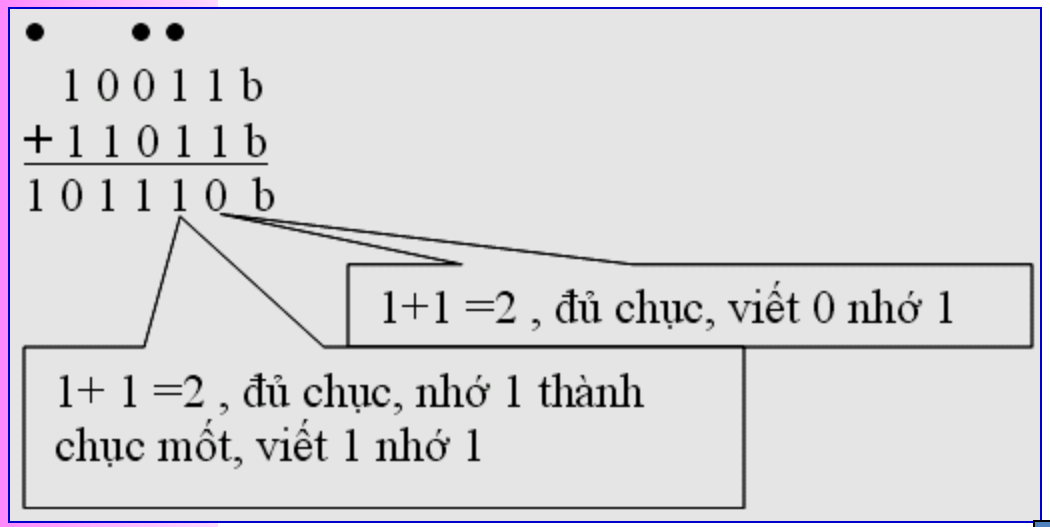
Read by yourself

Fill the blank cells

Decimal	Binary	Hexa.	Decimal	16-bit Binary	Hexadecimal
9	1001	9	255	0000 0000 1111 1111	00FF
127	0111 1111	9F	192		
125			188		
157			312		
162			517		
37			264		
66			543		
77			819		
88			1027		
99			2055		
109			63		

Data Representations: Operations

Read by yourself



Do yourself:

3245q + 247q

1A7Bh + 26FE7h

+

101101111 b
100111011 b
110110001 b
110001101b

Data Representations: Operations …

	1	0	0	1	1	b
-		1	1	0	1	b
Trả	1	1 → 0				
Kết qủa	0	0	1	1	0	b

Cộng bên dưới: 1+1=2, đủ chục, 0 nhớ 1

0 không trừ được 1, mượn 1 chục là 2, 2 trừ 1 = 1. Trả 1 chục

1-0=1

1-1=0

Read by yourself

Cùng cách này ta cộng trừ được các số trong hệ 8, 16

Do yourself

1101101101b - 10110111b 3654q – 337q 3AB7h – 1FAh

36Ah – 576q = ? h 64AEh – 1001101b= ? q

Data Representations: Operations …

Nhân là hội cách phép cộng

```
    1 0 0 1 b
  x 1 0 1 b

    1 0 0 1
+ 0 0 0 0
  1 0 0 1
  1 0 1 1 0 1 b
```

Chia là hội các phép trừ

```
      1 0 1 1'0'1' b :  1 0 0 1 b
-     1 0 0 1           1 0 1 b
      0 0 1 0 0
-       0 0 0 0
        1 0 0 1
-       1 0 0 1
        0 0 0 0
```

thương số

hạ 0

hạ 1

dư số

Read by yourself

Bài tập : Làm các phép tính

1011010 b* 1011b

1101000b + 2AB h + 345 q = ? h = ? q

3AFh / 1Ch =? b = ?d

3ACh – 562q = ?b = ? d

3FFA h / 327q = ?b = ? d

Data Representations: Operations …

Read by yourself

Phép AND (và)

And 2 bit cho trị 1 khi cả 2 bit cùng mang tri 1

1 AND 1 = 1
1 AND 0 = 0
0 AND 0 = 0

```
  100101 b
AND 001101 b
  000101 b
```

Phép OR (hay)

Or 2 bit cho trị 1 nếu có bit tóan hạng mang trị 1

1 OR 1 = 1
1 OR 0 = 1
0 OR 0 = 0

```
  100101 b
OR  001101 b
  101101 b
```

Phép XOR – Không đồng thời giống trị nhau (không đội trời chung)

1 XOR 0 = 1
1 XOR 1 = 0
0 XOR 0=0

```
   1001101 b
XOR 0011110 b
   1010011 b
```

Phép NOT - Đảo bít

Not (100100b) → 011011b

Not (011011b) → 100100b

Data Representations: Signed Integers

Dùng bit trái (leftmost bit) để mô tả dấu:

0:positive, 1:negative

Biểu diễn số âm

Ta nghĩ rằng : 01000011 mô tả cho +67 thì 11000011 sẽ mô tả cho –67

Sai vì:

```
   01000011 b
+  11000011 b
  100000110 b không phải 0      (đòi hỏi : +67 –67 = 0)
```

Giải pháp số bù 2 Two-complement number

Biểu diễn bù 2 là kết qủa của phép đảo (Not) rồi +1

```
(+ 67)   01000011 b
(bù 1)   10111100 b ( đảo / Not / One-Complement)
+               1 b
  10111101 b ( bù 2) Đây là biểu diễn của –67
```

Read by yourself

Kiểm lại:

```
    (+ 67)  01000011 b
+   (-67)   10111101 b
        1 00000000 b  ( bít trái tràn, bỏ đi, ta có kết qủa 0, hợp lý )
```

Tóm lại : Biểu diễn số dương → bù 2 → Biểu diễn số âm
Biểu diễn số âm → bù 2 → Biểu diễn số dương

Data Representations: Signed Integers

Read by yourself

Tìm biểu diễn của –35 với 1 byte

Cách làm: Ta tìm biểu diễn của +35 rồi tìm bù 2 của nó ta được biểu diễn số âm

Tìm trị của biểu diễn số nguyên có dấu 1 byte 11111100 b

Bít trái : 1 → Biểu diễn số âm. Nếu ta tìm được biểu diễn dương tương ứng, tìm trị , thêm dấu trừ ta được trị kết qủa.

Bài tập

Tìm biểu diễn của số nguyên không dấu 1 byte: 251 , 163, 117

Tìm biểu diễn của số nguyên không dấu 2 byte: 551 , 160, 443

Tìm biểu diễn của số có dấu 1 byte: -51 , -163, -117, 320

Tìm trị của biểu diễn số nguyên có dấu 1 byte :

01100011 b , 10001111 b , 11001010 b , 01001100 b

1.8- Addressing Information

- Each byte of primary memory has a unique address, start from zero
  - Kilobyte = 1024 bytes
  - Kilo K= 1024 (2^{10})
  - Mega or M (=1024k)
  - Giga or G (=1024M)
  - Tera or T (=1024G)
  - Peta or P (=1024T)
  - Exa or E (=1024P)

B: byte
b: bit

- Addressible Memory
  - The maximum size of addressable primary memory depends upon the size of the address registers

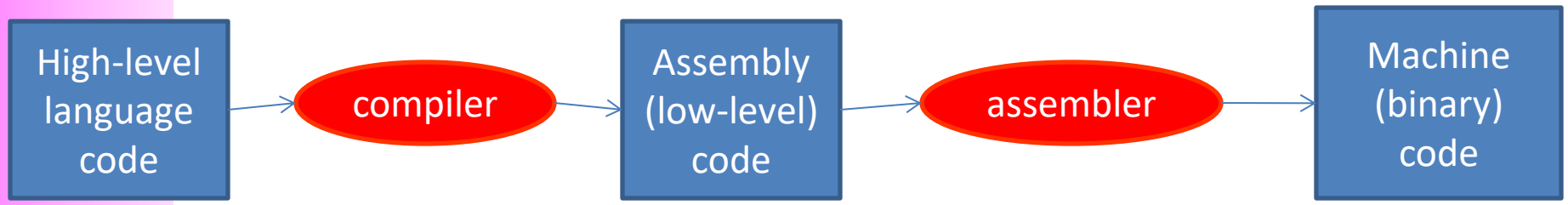
	MEMORY
…	…….
5	1010 1010
4	0011 1100
3	0101 0100
2	1001 0000
1	1100 1011
0	0100 0001

1.9- Program Instructions

01001011	100110110110	011011010111
Opcode	Operand 1	Operand 2

- Each program instruction consists of an operation and operands
- The CPU performs the operation on the values stored as operands or on the values stored in the operand addresses.
- Operands: Constants, registers, primary memory addresses

1.9- Program Instructions…

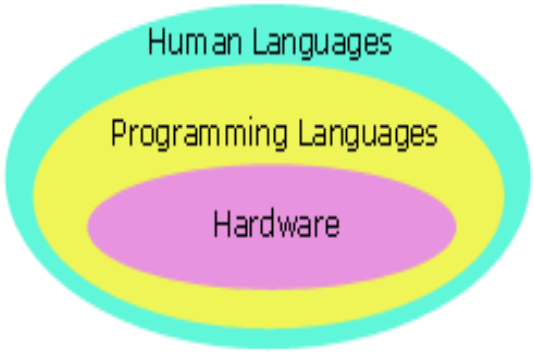



```
swap(int v[], int k)
{int temp;
  temp = v[k];
  v[k] = v[k+1];
  v[k+1] = temp;
}
```

```
swap:
    muli $2, $5,4
    add  $2, $4,$2
    lw   $15, 0($2)
    lw   $16, 4($2)
    sw   $16, 0($2)
    sw   $15, 4($2)
    jr   $31
```

```
00000000101000010000000000011000
00000000100011110000110000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

1.10- Programming Languages



- Programs that perform relatively simple tasks and are written in assembly language contain a large number of statements.
- Machine Language → Assembly language → High-level languages,
- To make our programs shorter, we use higher-level languages.

1.10- Programming Languages…

- 5 Generations of Programming Languages:
  (1) Machine languages.
  (2) Assembly languages.
  (3) Third-generation languages. These are languages with instructions that describe how a result is to be obtained (C, Pascal, C++, Java…).
  (4) Fourth-generation languages. These are languages with instructions that describe what is to be done without specifying how it is to be done (SQL).
  (5) Fifth-generation languages are the closest to human languages. They are used for artificial intelligence, fuzzy sets, and neural networks (Prolog, Matlab)

1.10- Programming Languages…

- The higher the level, the closer to the human languages and the further from native machine languages
  - Each third generation language statement ~ 5-10 machine language statements.
  - Each fourth generation language ~ 30-40 machine language statements.

1.11- Translate and Execute a Program

- When we code a program in a high level language, we write source code. We translate this code into machine language.
- 2 ways of translations:
  - Interpreting: Translate and execute statements one-by-one → **Interpreter**
  - Compiling: Translate all the program then execute all the program → **Compiler**
- **C translator is a compiler**

1.12- Why C is the 1st Language?

- C is one of the most popular languages in use globally

- Some **reasons** for learning programming using the C language include:

Language	*Time to Run*
Assembly	0.18 seconds
C	2.7 seconds
Basic	10 seconds

Comparative times for a Sieve of Eratosthenes test

  - C is English-like,
  - C is quite compact - has a small number of keywords,
  - A large number of C programs need to be maintained,
  - C is the lowest in level of the high-level languages,
  - C is faster and more powerful than other high-level languages,
  - The UNIX, Linux and Windows operating systems are written in C and C++.

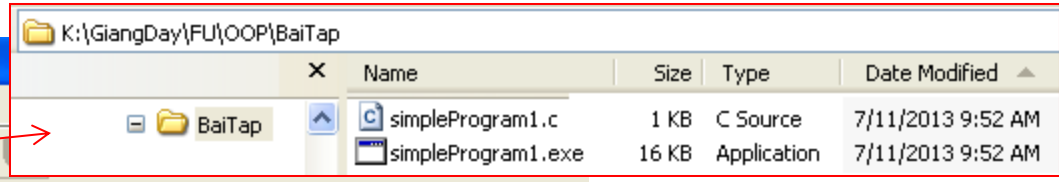
1.13- Some Notable C Features

- Comments

  /* */

  – We use comments to document our programs and to enhance their readability. C compilers ignore all comments.
- Whitespace
  – We use whitespace to improve program readability and to display the structure of our program's logic. C compilers ignore all whitespace
- Case Sensitivity
  – C language is case sensitive.
  – C compilers treat the character 'A' as different from the character 'a' .

1.14- Structure of a Simple C Program



Dev-C++ 4.9.9.2

File Edit Search View Project Execute Debug Tools CVS Window Help

New Insert Toggle Goto

simpleProgram1.c

```
1  /* simpleProgram1.c
2     Demo. for a simple C program
3     Author: SuTV
4     Date: 2013/7/11
5  */
6  #include <stdio.h>
7  int main()
8  { /* Print out a greeting to the screen */
9    printf("Hello everyone from C");
10   /* wait user pressing the ENTER key */
11   getchar();
12   /* Quit the program */
13   return 0;
14 }
```

Comment for program description

Declaration for library using

Entry point of C-program

Statements + comments

Exit point of C-program

Structure…: C program Entry Points

Entry point: the point that a program begins.

Entry points of C-programs:

```
[int] main( [void] )
{  <statements>
   [ return number; ]
}
```

Common form

```
[int] main( int argCount, char* args[])
{  <statements>
   [ return number; ]
}
```

Demo.
In the module H
(Files)

Summary

- Definitions: computer program, computer software
- How to make a good software?
- Steps to develop a software?
- Computer hardware.
- Information, Data, Fundamental Data Units
- Data Representation
- Program Instructions
- Languages
- C Compilers
- Why C is the first language selected?
- Some notable features of C
- Structure of a simple C Program.