Slot 2
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 some concepts related to programming
- Explain how to make a good software
- Understand steps to develop a software
- Explain ways for representing data
- Answer why C is the first language selected
- Understand how a C program can be translated and execute
- Discuss about notable features of the C language
- Understand a C program structure

Contents

- Definitions
- How to make a good software?
- Steps to develop a software?
- Computer hardware.
- 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.

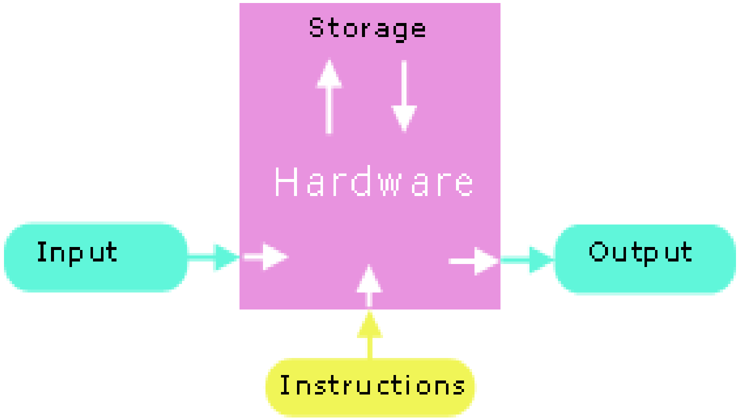
1- Definitions

- Information: Knowledge about something
- Data: Values are used to describe information. So, information can be called as the mean of data
- Problem: A situation in which something is hidden
- Solve a problem: explore the hidden information
- Solution: Value(data) of hidden information
- Algorithm: a way to find out a solution
- Program: A sequence of steps to find out the solution of a problem. A program is a implementation of an algorithm
- Computer program: a program is executed using a computer

Definitions…

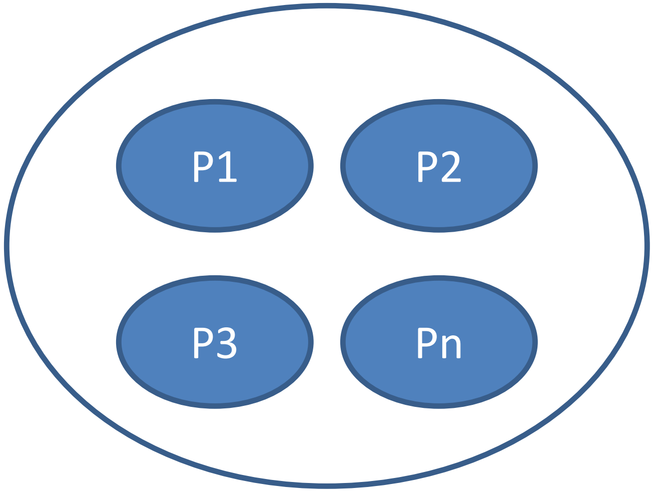
- Computer program = data + instructions
  - A **simulation** of solution.
  - Is a set of instructions that computer hardware will execute

  ➔Increase **performance** of standard workflow



- Computer software:

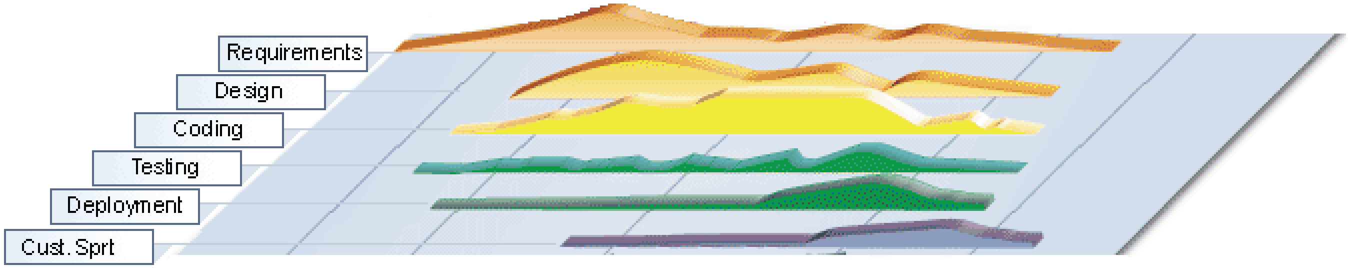
  - A set of related programs



2- How to make a good software?

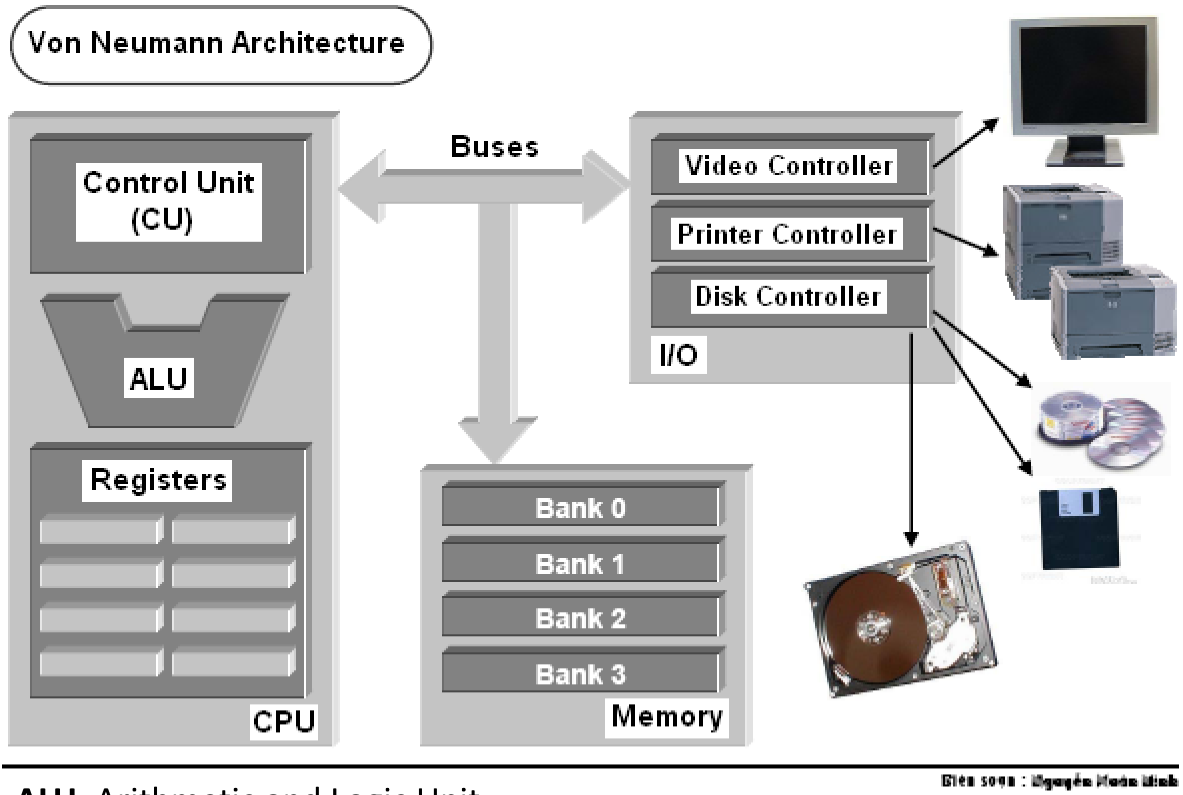
- Issues for a program/software:
  - Usability: Users can use the program to solve the problem
    - robust and user-friendly interfaces
  - Correctness: Solution must be correct
    - comprehensive testing
  - Maintainability: The program can be modified easily
    - Understandability
      - structured programming
      - internal documentation
    - Modifiability
      - standards compliance
  - Portability: The program can run in different platforms
    - standards compliance → Needed modifications are minimum (platform: CPU + operating system running on it)

3- Steps to develop a software



- Requirements → The problem is understood
- Analysis → Data and tasks are identified
- Design → folders, files are organized
- Coding → Implementation
- Testing → Checking whether requirements are satisfied or not
- Deploying → Program is installed to user computers
- Maintenance → Needed modifications, if any, are carried out

4- Computer Hardware - Review



ALU: Arithmetic and Logic Unit

3 steps to read a memory cell:

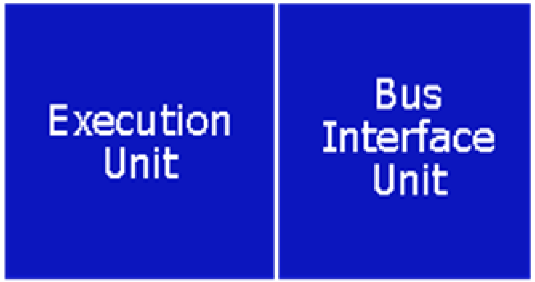
(1) CPU puts the memory address to address bus

(2) CPU puts the read-signal to control bus.

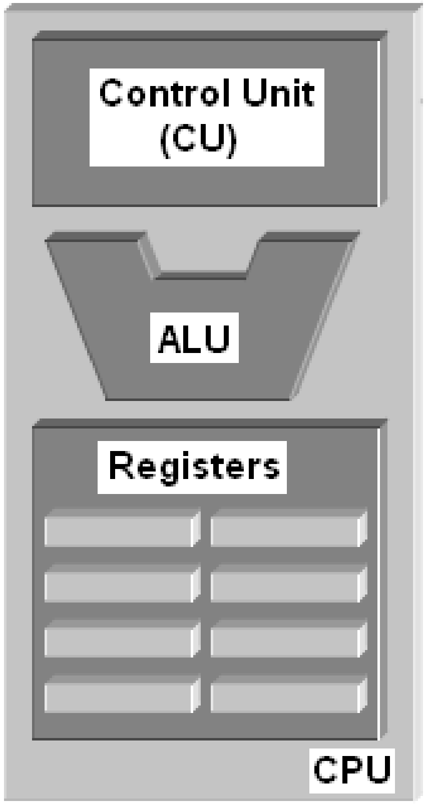
(3) Data in memory cell is transferred to a register in CPU

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

Computer Hardware…



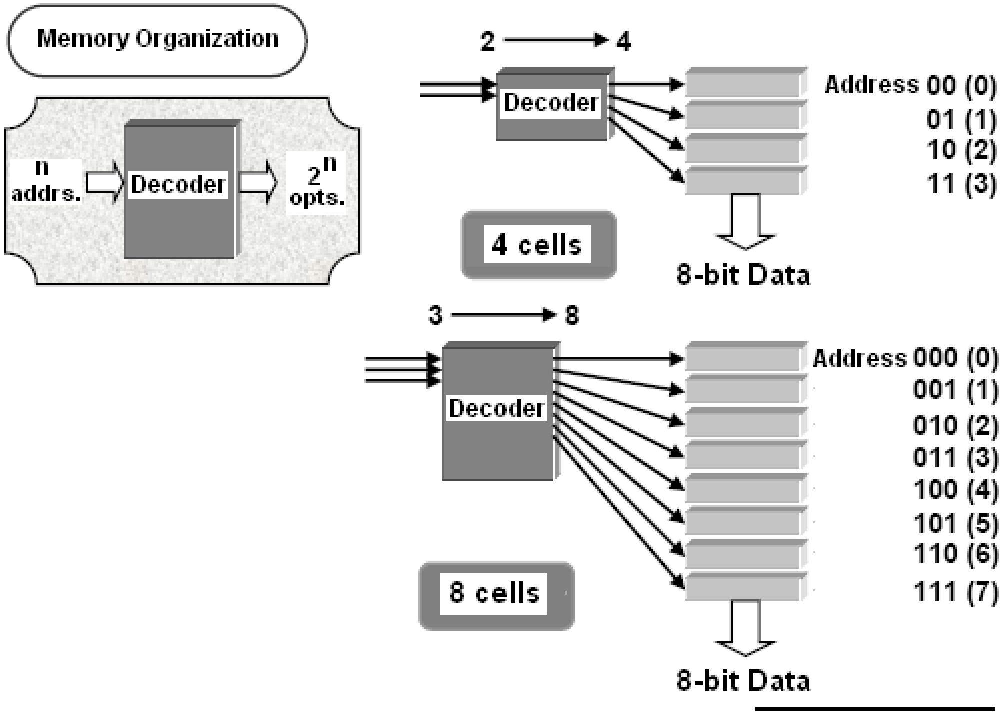
- The most expensive and fastest memory - registers - is reserved for the CPU.
  - CPU transfers information at less than 10 nanoseconds
  - primary memory transfers information at about 60 nanoseconds
  - a hard disk transfers information at about 12,000,000 nanoseconds



Biên soạn : Nguyễn Hoàn Minh

- CPU memory is volatile - the contents of the registers are lost as soon as power is turned off.

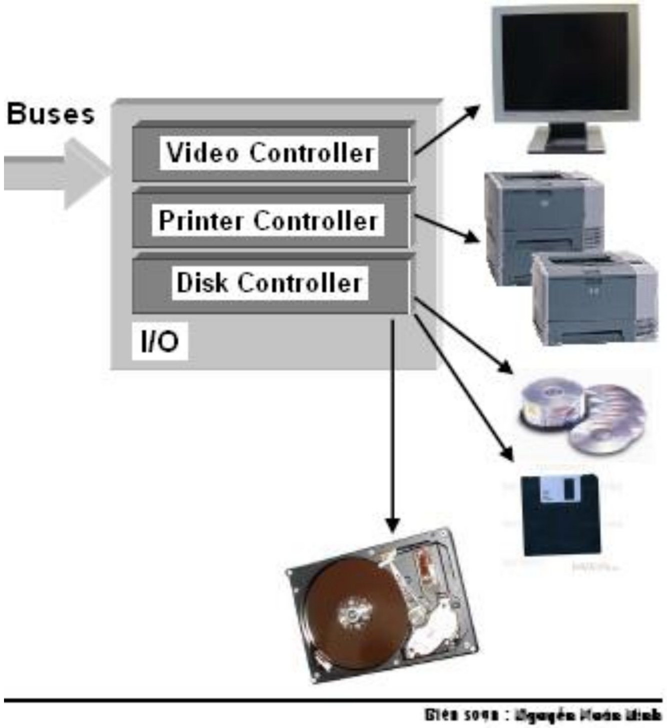
Computer Hardware…



- 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).
  - A specific memory cell is identified uniquely by a decoder. Decoder has n inputs and 2^n outputs. With a specific input, only one output is chosen (value=1), others having the value 0

Computer Hardware…

- Devices
  - Include basic 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).
  - All device interfaces connect to the system buses through a central controller.



5- Data Units

- Transistor is the basic physical unit for storing data → Binary format
- 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 **bi**nary digi**t** as a bit.
- Nibble = 4 consecutive bits.
- Byte = 8 consecutive bits
  = 2 nibbles
- Unit of memory is BYTE

Byte							
Nibble				Nibble			
Bit	Bit	Bit	Bit	Bit	Bit	Bit	Bit

00000000 <- possibility 0

00000001 <- possibility 1

00000010 <- possibility 2

00000011 <- possibility 3

00000100 <- possibility 4

…

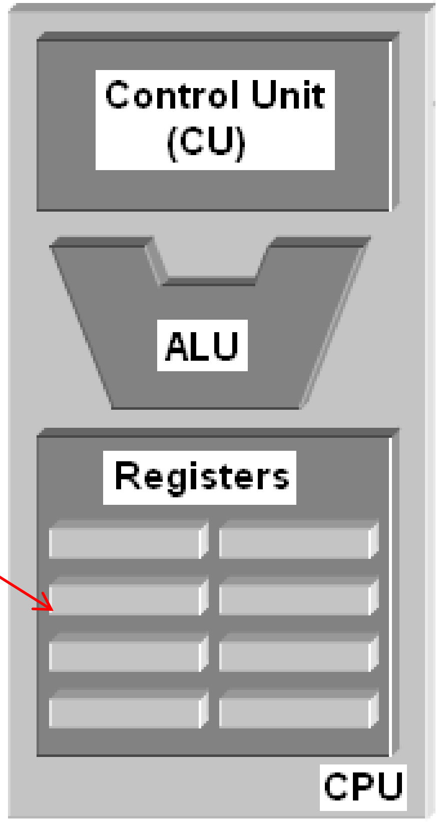
00111000 <- possibility 104

…

11111111 <- possibility 255

Data Units …

- The natural unit of the CPU is a word.
- The word length is number of bits of a general register within CPU(CPU memory).
- Word length can be 8, 16 (old CPUs), 32, 64 (current CPUs)



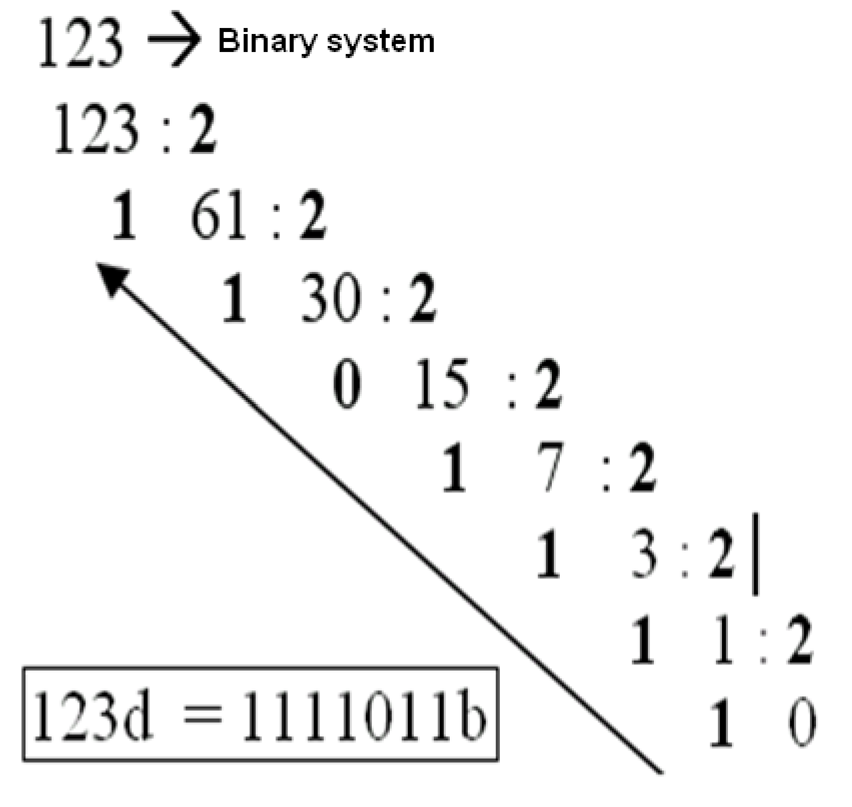
Biên soạn : Nguyễn Hoàng Minh

6- Data Representations

- Data in computer are binary values → They can be treated as numbers.
- 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
  - We can convert a number in one system to another (introduced in the subject Introduction to Computing) → Next 12 slides will be read by yourself. Use your notebook for doing exercises.

Read by yourself

Data Representations: Conversion- A review



123 → Binary system

123 : 2

1 61 : 2

1 30 : 2

0 15 : 2

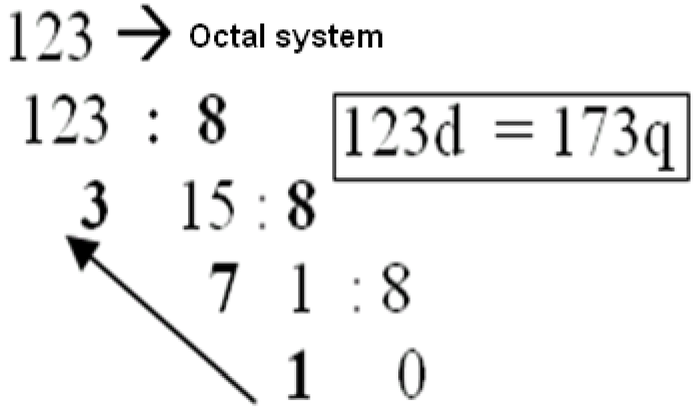
1 7 : 2

1 3 : 2

1 1 : 2

1 0

123d = 1111011b



123 → Octal system

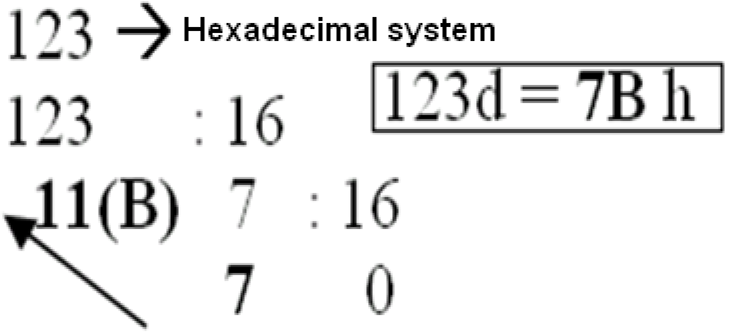
123 : 8

3 15 : 8

7 1 : 8

1 0

123d = 173q



123 → Hexadecimal system

123 : 16

11(B) 7 : 16

7 0

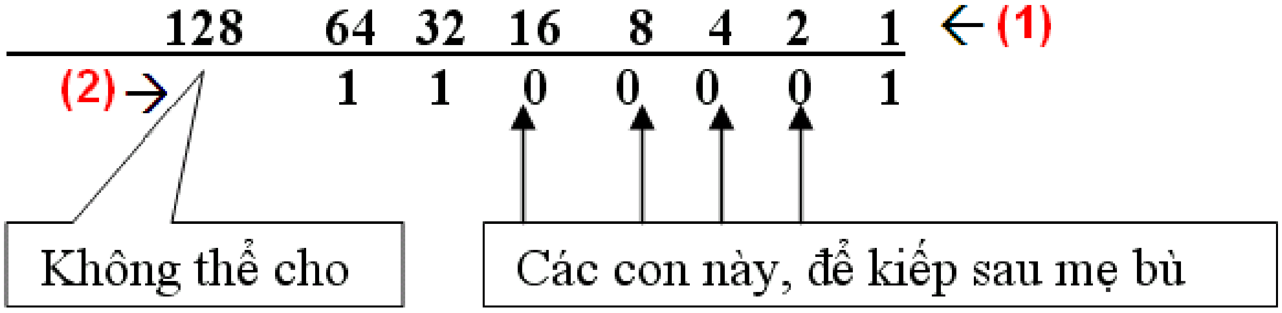
123d = 7B h

Read by yourself

Data Representations: Conversion: A review

97 → Binary system

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)



	128	64	32	16	8	4	2	1	← (1)
(2) →		1	1	0	0	0	0	1	

Không thể cho

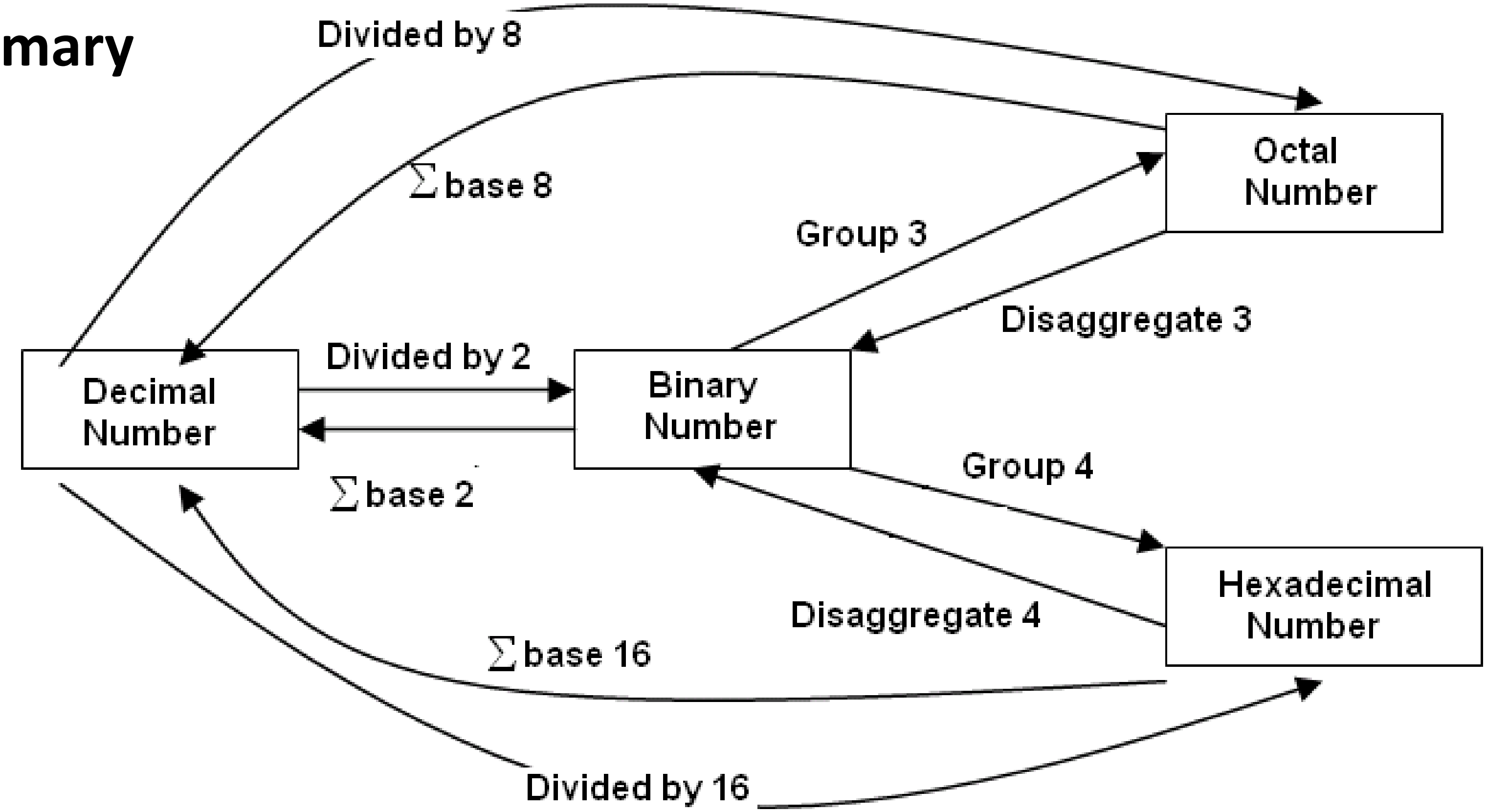
Các con này, để kiếp sau mẹ bù

97d = 110 0001b

Read by yourself

Data Representations: Conversion: A review

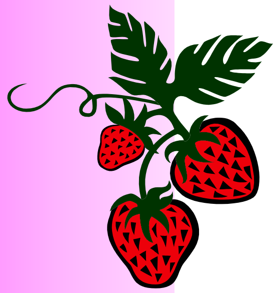
Summary

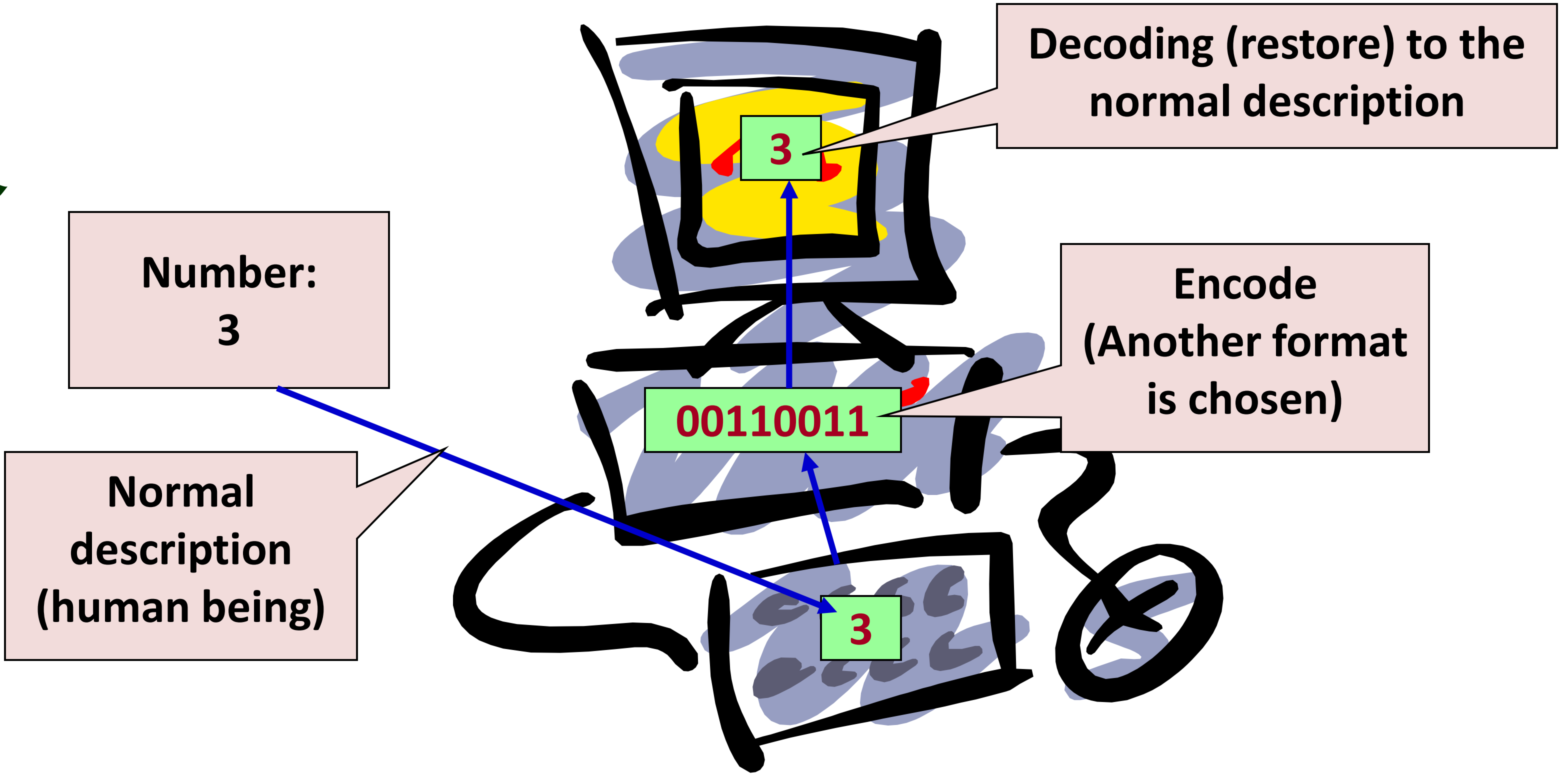


Data Representations: Conversion…

Read by yourself

- Computer is a binary device → All data are stored in binary format




Number:
3
Normal description (human being)
3
00110011
3
Decoding (restore) to the normal description
Encode (Another format is chosen)

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	

Do by yourself

Data Representations: Conversion…

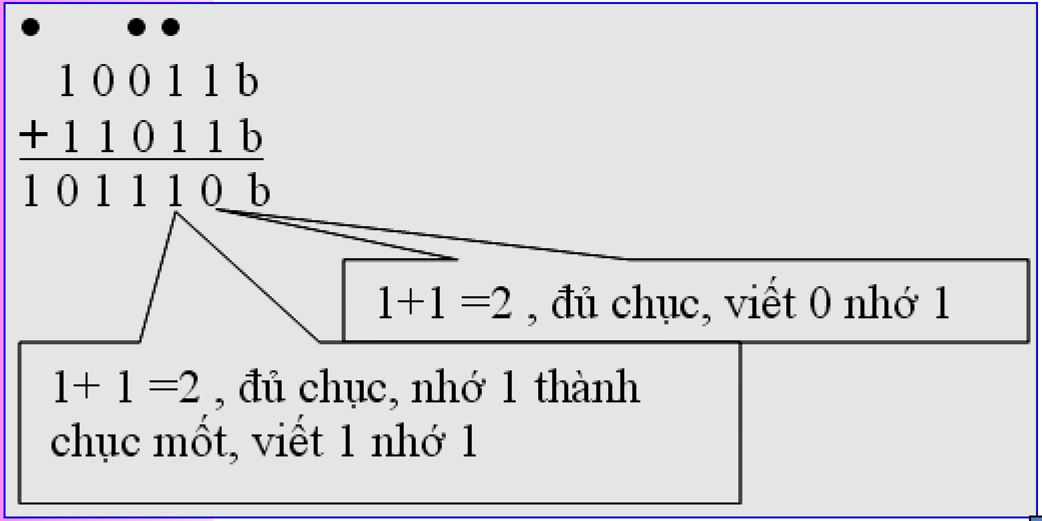
Do by yourself

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



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

1+1 =2 , đủ chục, viết 0 nhớ 1

1+ 1 =2 , đủ chục, nhớ 1 thành chục mốt, viết 1 nhớ 1

Do yourself:

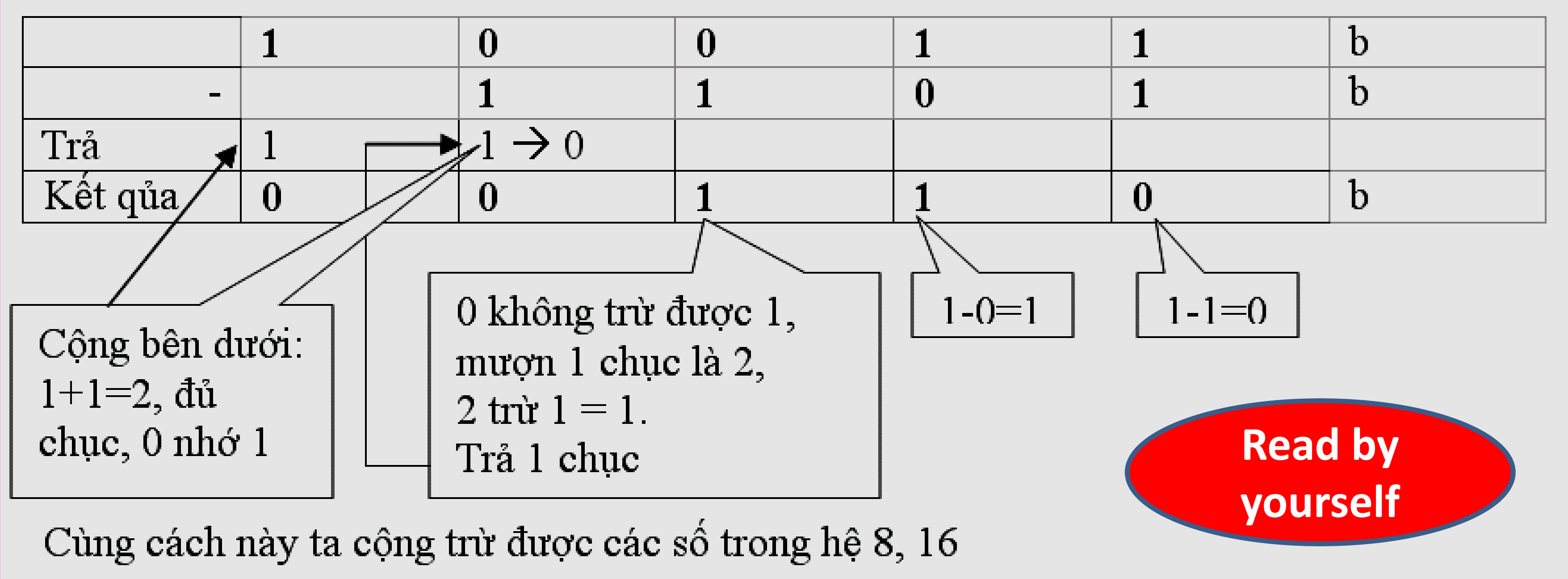
3245q + 247q

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

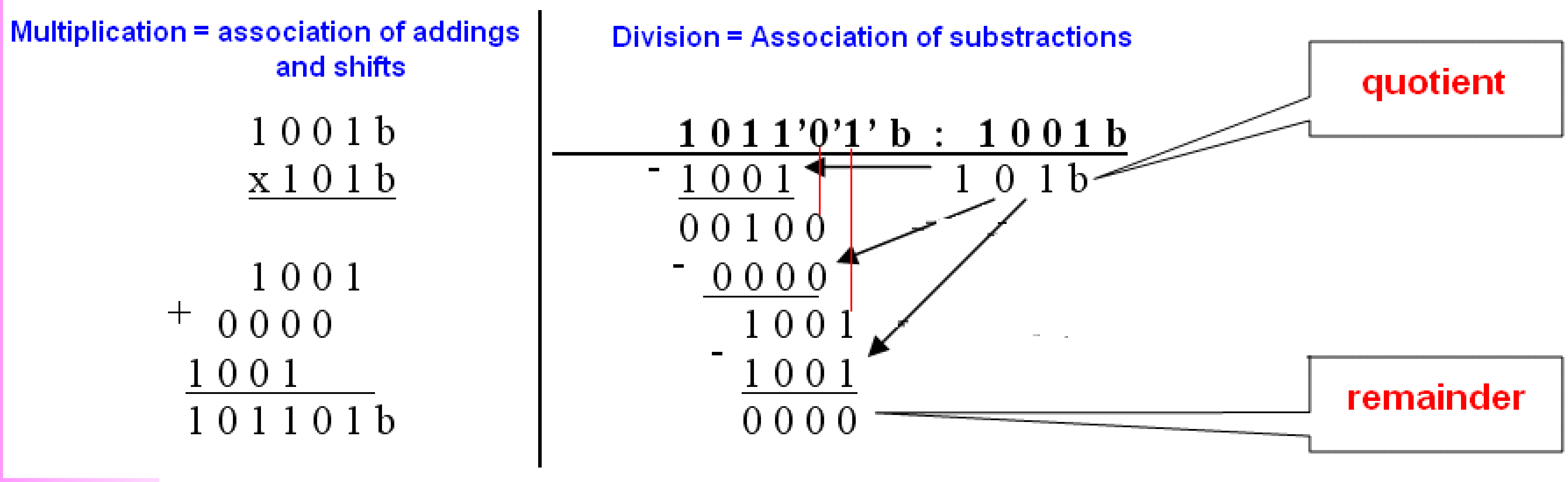
1-1=0

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

Read by yourself

Do yourself
1101101101b - 10110111b 3654q – 337q 3AB7h – 1FAh
36Ah – 576q = ? h 64AEh – 1001101b= ? q

Data Representations: Operations …



Multiplication = association of addings and shifts

```
       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
```

Division = Association of substractions

```
    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
```

quotient

remainder

Read by yourself

<u>Exercises</u> :

1011010 b* 1011b

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

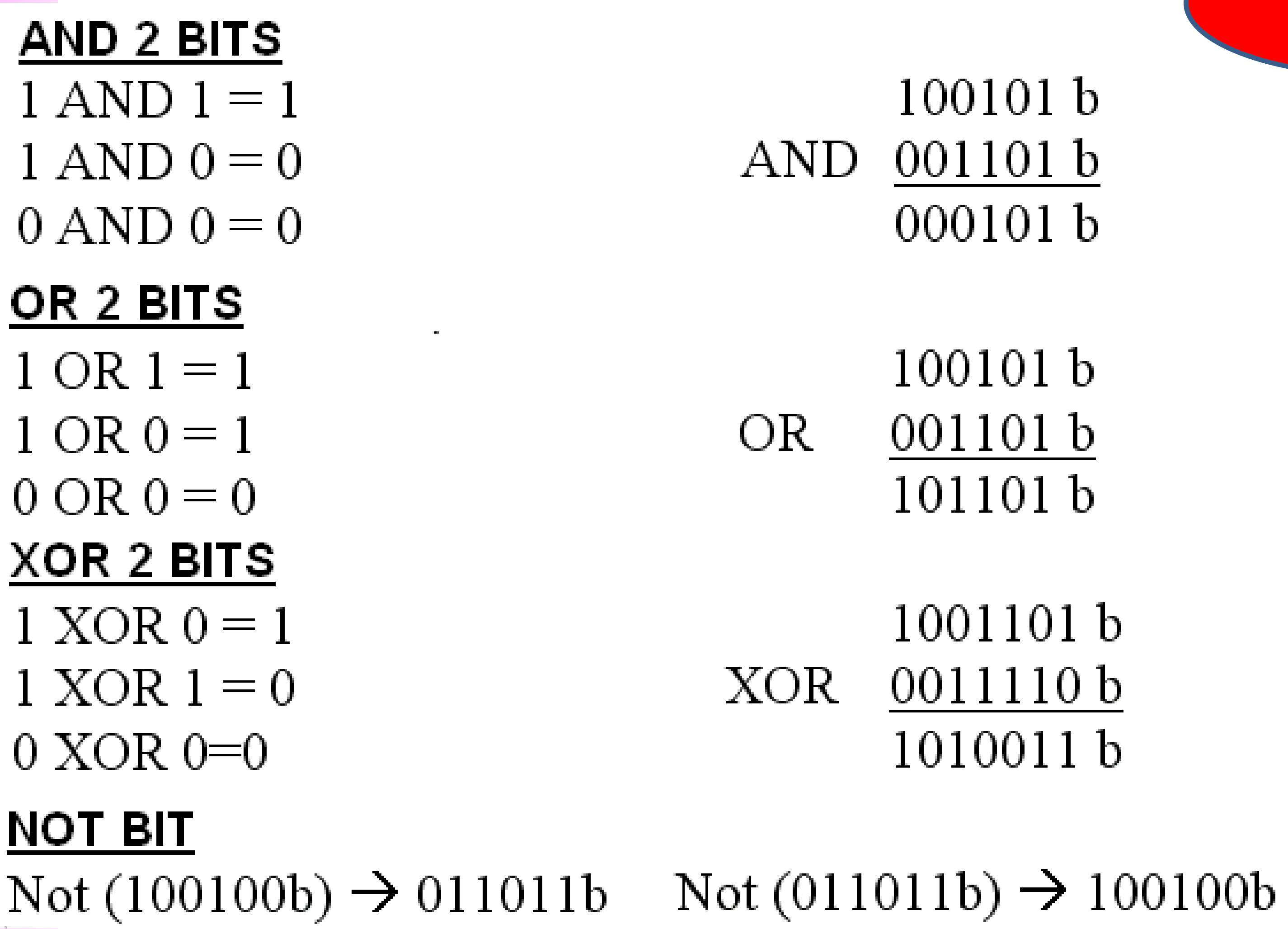
3AFh / 1Ch =? b = ?d

3ACh – 562q = ?b = ? d

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

Data Representations: Operations …

Read by yourself



AND 2 BITS

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

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

OR 2 BITS

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

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

XOR 2 BITS

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

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

NOT BIT

Not (100100b) → 011011b   Not (011011b) → 100100b

Data Representations: Signed Integers

The leftmost bit is the sign bit.

0:positive, 1:negative

Read by yourself

Representing negative integer

67d , 1 byte → 01000011

→-67d → 11000011

Check: 67 + (-67) =0

```
       0100 0011
  +    1100 0011
      10000 0110 → False
```

Solution: Use 2-complement format

(+67) → 0100 0011

1011 1100 (1-complement/reverse bits/ Not operator)

```
            +          1
(-67)        1011 1101 (2-complement)
```

Check:

```
(67)          0100 0011
(-67)         1011 1101
+          0  0000 0000
```

Positive representation ← 2-complement → negative representation

Data Representations: Signed Integers

Read by yourself

Give binary representation of –35 using 1 byte

Solution: +35 → binary representation → 2-complement → Binary representation of -35

Give the decimal of the binary presentation of a signed one-byte integer 11111100 b

Leftmost bit is 1 → This is a binary representation of a negative integer. 1111 1100 →2-complement format→ positive number → Decimal number n → -n is the value of this representation

Exercises

Show binary formats of 1-byte unsigned numbers:

251 , 163, 117

Show binary formats of 2-byte unsigned numbers:

551 , 160, 443

Show binary formats of 1-byte signed numbers:

-51 , -163, -117, 320

Show the decimal values of 1-byte unsigned representations: :

01100011 b , 10001111 b , 11001010 b , 01001100 b

7- Addressing Information

- Each byte of primary memory has a unique address (order number), starting 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)
- Addressible Memory
  - The maximum size of addressable primary memory depends upon the size of the address registers

B: byte
b: bit

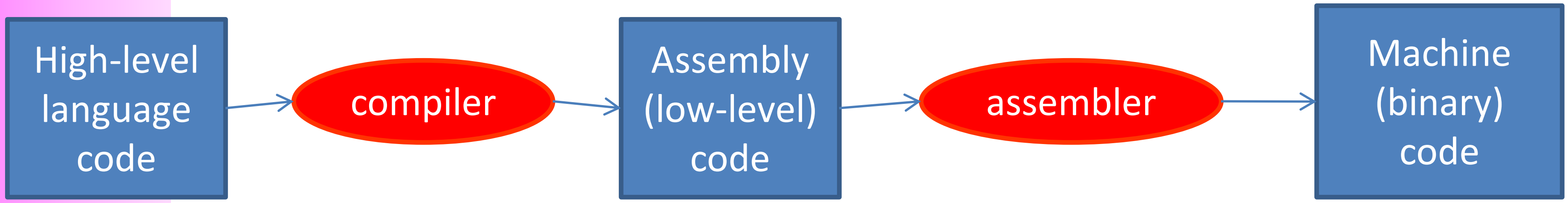
	MEMORY
…	…….
5	1010 1010
4	0011 1100
3	0101 0100
2	1001 0000
1	1100 1011
0	0100 0001
Address	value

8- 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

Program Instructions…

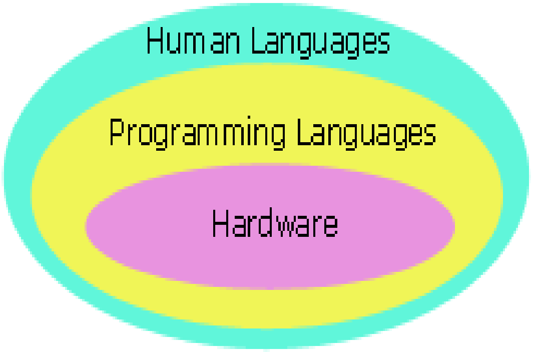


```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

```
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
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

9- 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.

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)

Programming Languages…

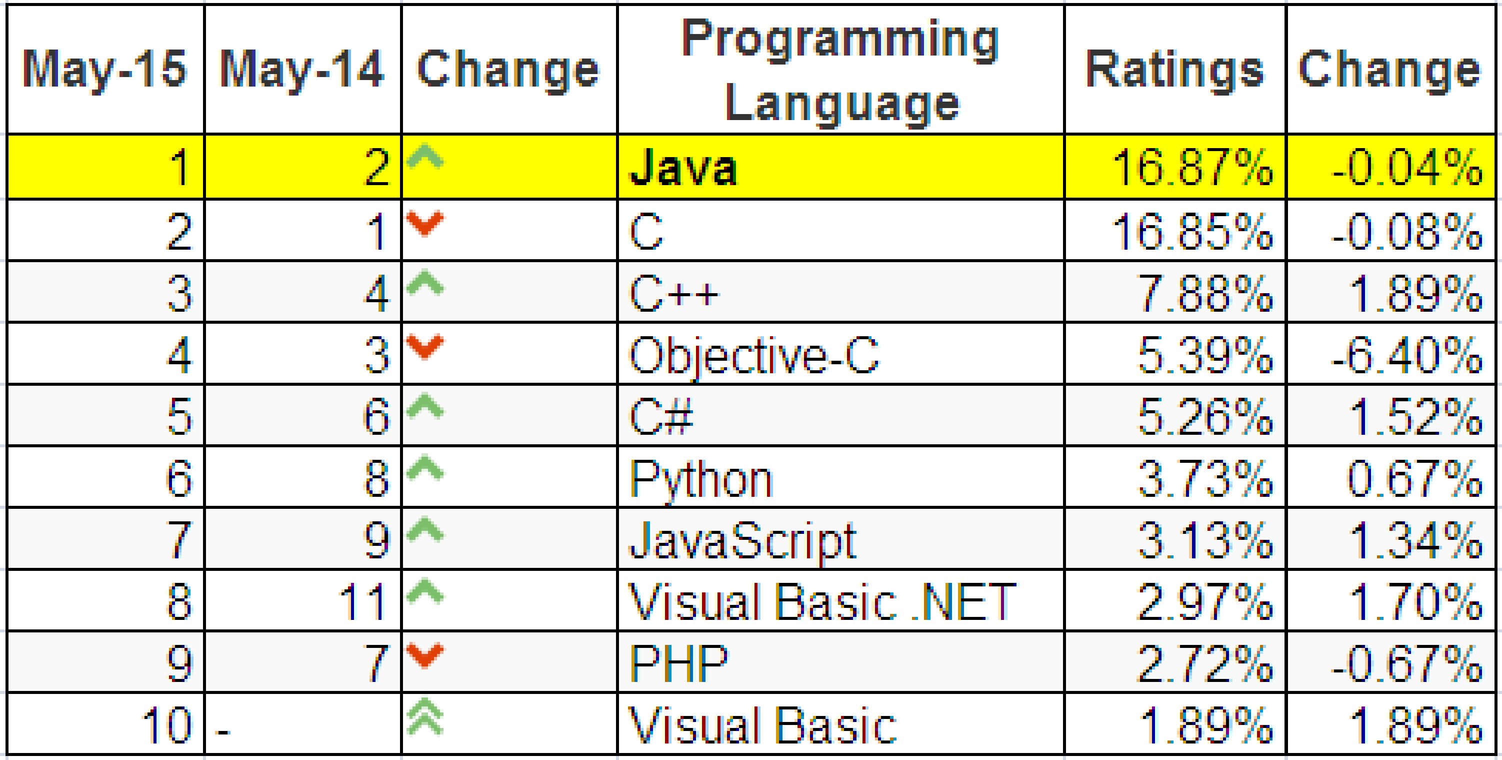
- The higher the level, the closer to the human languages and the further from native machine languages
  - Each third generation language statement ~ 5-10 machine language statements.
  - Each fourth generation language ~ 30-40 machine language statements.

10- Translating and Executing a Program

- Program code in a high level language can not run, It must be translated to binary code (machine code) before running.
- 2 ways of translations:
  - – Interpreting: one-by-one statement is translated then run → **Interpreter**
  - – Compiling: All statements of program are translated then executed as a whole → **Compiler**
- **C translator is a compiler**

11- Why C is the 1st Language?

- **Top ten common programming languages:**



May-15	May-14	Change	Programming Language	Ratings	Change
1	2	↑	**Java**	16.87%	-0.04%
2	1	↓	C	16.85%	-0.08%
3	4	↑	C++	7.88%	1.89%
4	3	↓	Objective-C	5.39%	-6.40%
5	6	↑	C#	5.26%	1.52%
6	8	↑	Python	3.73%	0.67%
7	9	↑	JavaScript	3.13%	1.34%
8	11	↑	Visual Basic .NET	2.97%	1.70%
9	7	↓	PHP	2.72%	-0.67%
10	-	⇈	Visual Basic	1.89%	1.89%

From http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

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 of 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++.
  - The most common languages, such as Java, C#, are similar to C.
  - C supports basic ways which help us understanding memory of a program. These can be hidden in higher languages.

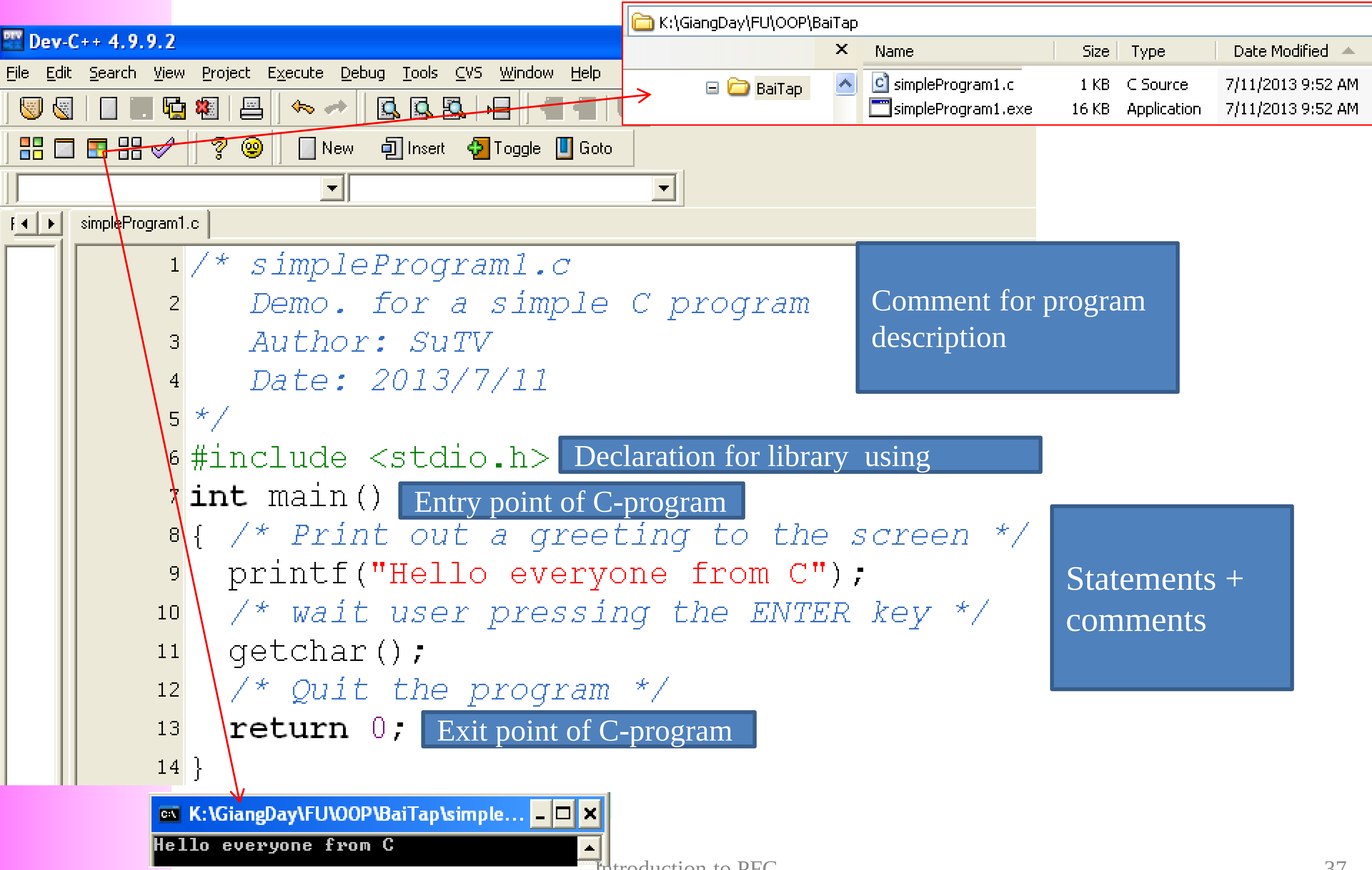
12- Some Notable C Features

- 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' .

13- 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
BaiTap

K:\GiangDay\FU\OOP\BaiTap

Name	Size	Type	Date Modified
simpleProgram1.c	1 KB	C Source	7/11/2013 9:52 AM
simpleProgram1.exe	16 KB	Application	7/11/2013 9:52 AM

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

Comment for program description

Declaration for library using

Entry point of C-program

Statements + comments

Exit point of C-program

K:\GiangDay\FU\OOP\BaiTap\simple...

```
Hello everyone from C
```

Structure…: C program Entry Points

Entry point: the point where 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 related to programming
- How to make a good software?
- Steps to develop a software?
- Computer hardware.
- 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.