

Structured Programming - Introduction

Xin Feng

Outline

- Natural language
- Programming language
- C programming language
- Program
- A simple program example

Natural Language

1. U Col egelloc si rta



2. Is liberal art UIC college a



3. Apple is a liberal-arts college



Understand these sentences???

Natural Language

- Recognized by **human beings**
 - Chinese, English, German, French, ...
- To define a natural language, we need to define
 - **Vocabulary (spelling)**
 - Col egelloc si rta
 - **Syntax (grammar)**
 - Is liberal art UIC college a.
 - **Semantics (meaning)**
 - Apple is a liberal-arts college.

Natural Language

UIC is a liberal-arts college.

Correct

- Vocabulary
- Grammar
- Semantics

Programming Language

- Recognized by **computers**
- To define a programming language, we need also to define
 - Grammar (including vocabulary)
 - Semantic
- You can invent a programming language too!

Programming Language & Natural Language

Natural language



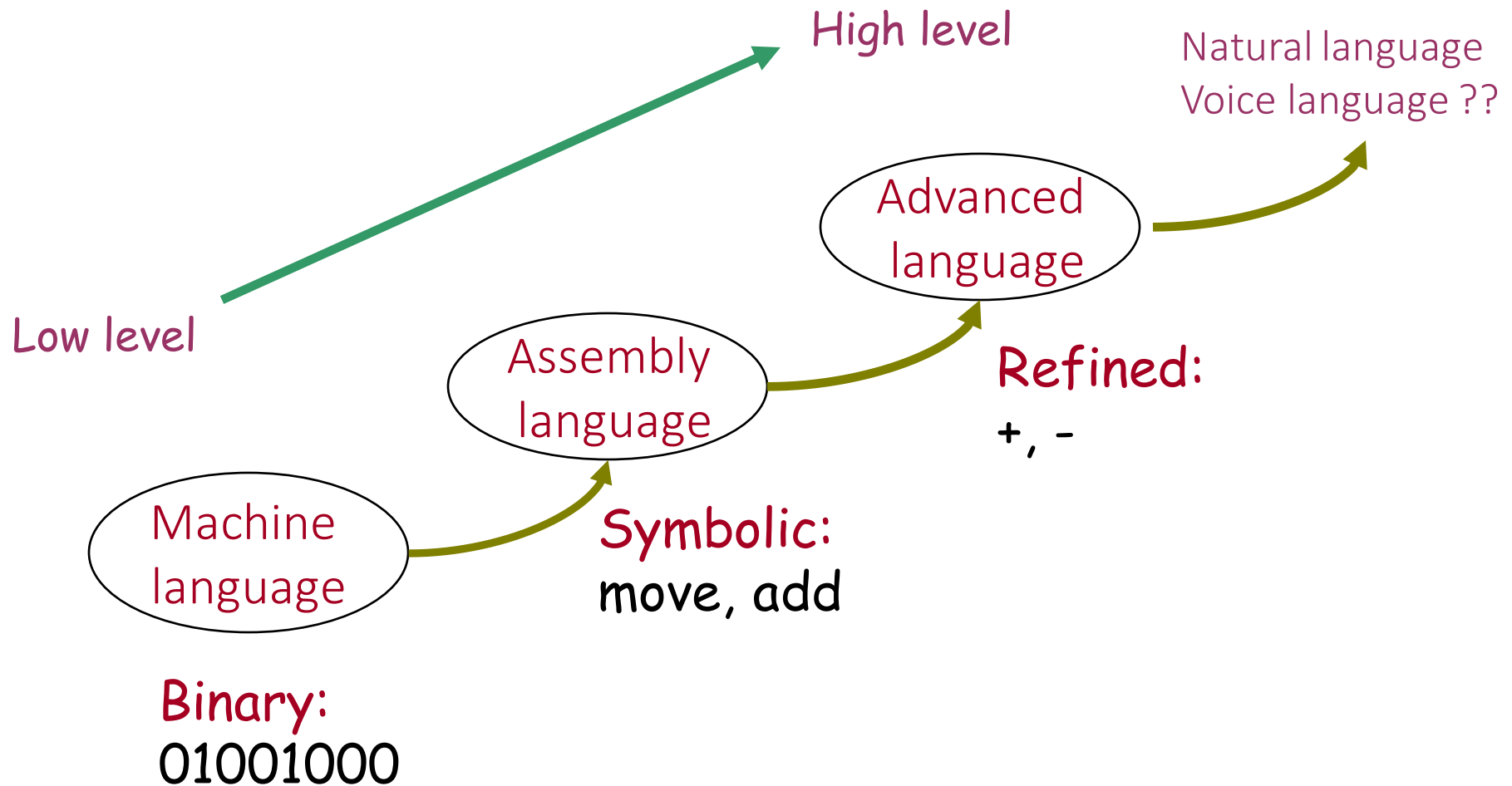
Articles
Documents

Programming language



Instructions
Programs

Evolution of Programming Language



Machine Language

- A CPU accepts *instructions* in machine language
- An instruction consists of 0s and 1s (binary number)
- Difficult for human to understand
- E.g.,

00000101

00010000

00000000

What does this mean



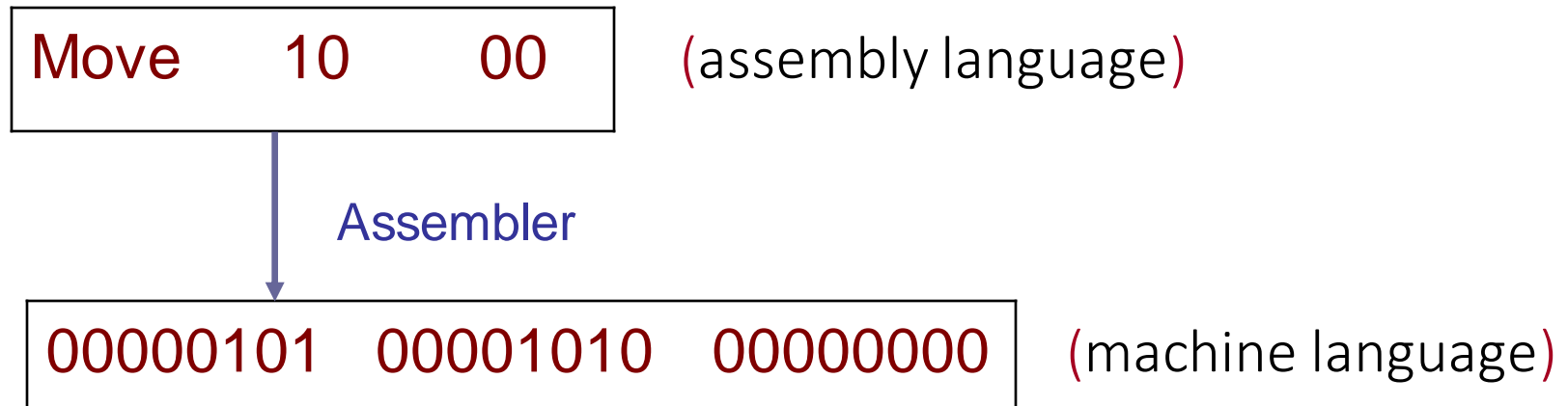
Machine Language

- A CPU accepts **instructions** in machine language
- An instruction consists of 0s and 1s (binary number)
- Difficult for human to understand
- E.g.,

00000101	00010000	00000000
Move	Value	Address

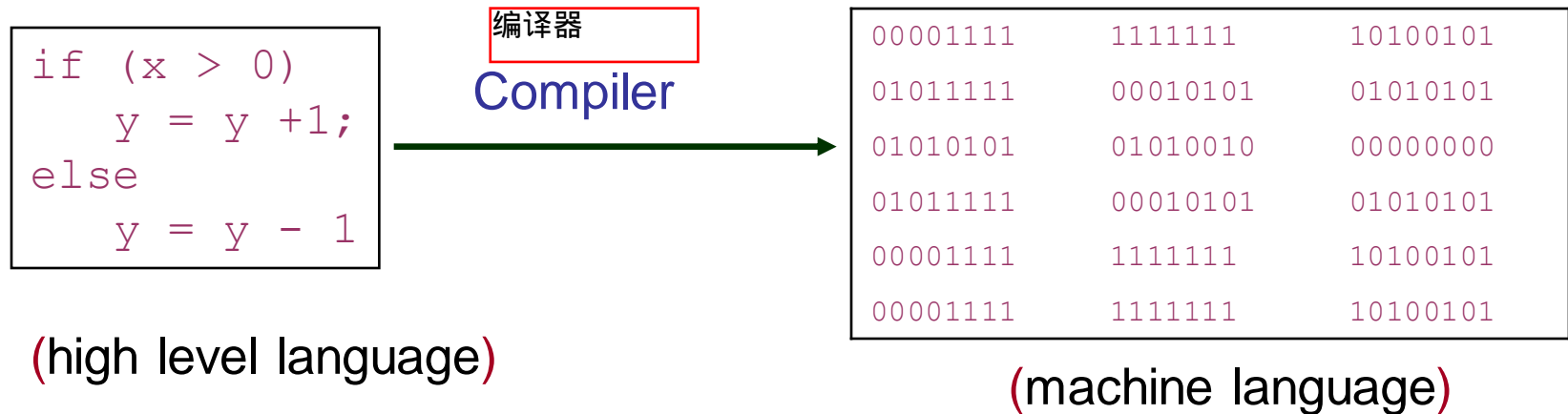
Assembly Language

- An **assembly language** uses symbols to represent the machine language instructions.
- An **assembler** is needed to **translate** symbolic code into machine language



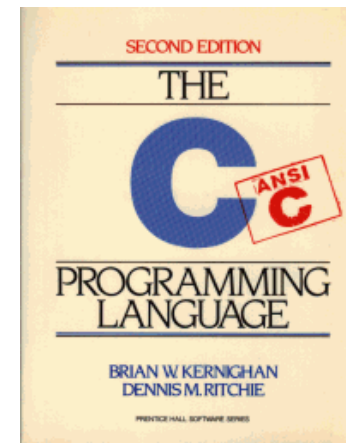
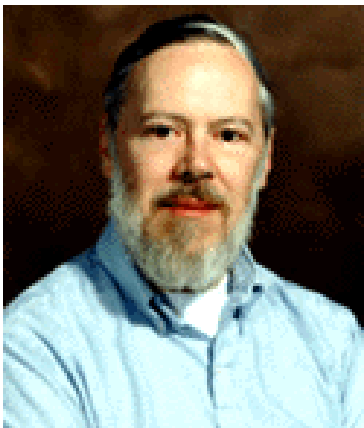
High Level Language

- Close to natural languages, using "if...then...else", etc.
- Make life easy for the programmers
- A **compiler** is needed to **translate** high level language programs into machine language instructions
- Examples
 - Java, **C, C++**, Pascal, Basic, Fortran,...



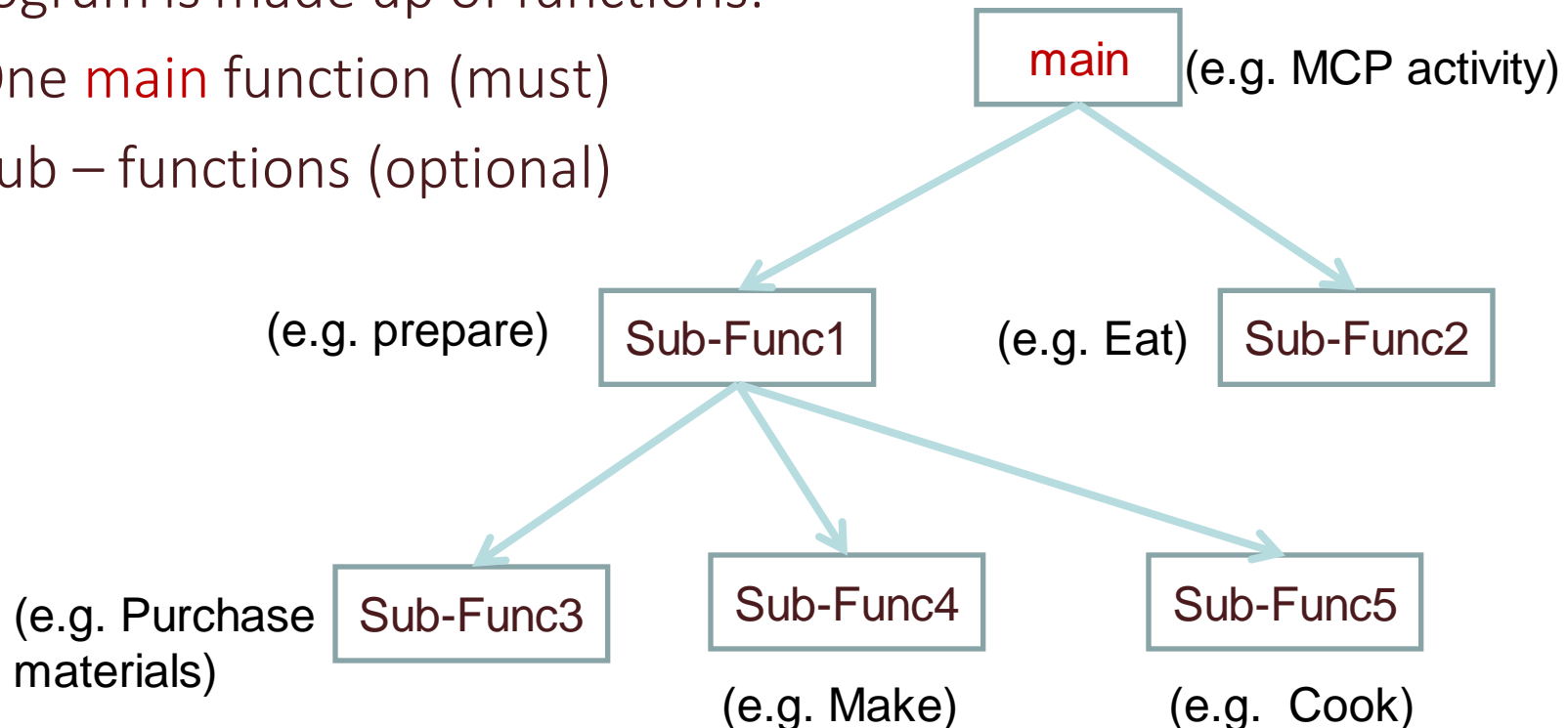
C Programming Language

- C is a high level language
- Created by Dennis M. Ritchie in 1972
 - First textbook: K&R, *The C Programming Language*.
- ANSI (American National Standards Institute)
 - First edition: ANSI C 1983.
 - Second edition: ANSI C 1999



Structured Programming

- C programming language is a structured programming language
- A program is made up of functions.
 - One **main** function (must)
 - Sub – functions (optional)



Our First Program

```
/* Our first program */  
#include <stdio.h>  
int main() {  
    printf( "\nHello World!\n" );  
    return 0;  
}
```

Guess what this program wants to do

Preprocessor

```
/* Our first program */  
#include <stdio.h>  
int main() {  
    printf( "\nHello World!\n" );  
    return 0;  
}
```

- The **preprocessor** is used to tell compilers some info used in compiling.
- `stdio.h`: **header file**. If you want to use “printf” in the program, you must write `#include <stdio.h>` at the beginning.

Comments

```
/* Our first program */  
#include <stdio.h>  
int main() {  
    printf( "\nHello World!\n" );  
    return 0;  
}
```

- A **comment** is used to explain some parts in the program.

Comments

- Two ways to insert comments into a C program
 - 1. `//` (double slash)
 - Only for one line
 - E.g., `//Our first program`
 - 2. `/* */`
 - Can be used for multiple lines
 - E.G., `/*our first program */` or
`/*our first
program*/`
- Compiler will ignore comments
 - Comments will not be translated into machine language
- It is a good habit to insert comments into your programs
 - Programs will be more readable

Main function

```
/* Our first program */  
#include <stdio.h>  
int main() {  
    printf( "\nHello World!\n" );  
    return 0;  
}
```

- All the programs written in C must have a **main function**
 - **main**: function name
 - **int**: function return type
- The part between the first '{' and the last '}' is called **body of function**

Statements

```
#include <stdio.h>
/* Our first program */
int main() {
    printf( "\nHello World!\n" );
    return 0;
}
```

- A **statement** is an instruction telling a computer what to do
- A **simple statement** ends with ‘;’
- A **compound statement** will include a sequence statements or conditions
- How many statements in this program?

Indentation

```
/* Our first program */  
#include <stdio.h>  
int main() {  
    printf( "\nHello World!\n" );  
    return 0;  
}
```

With indentation

Which one looks more clear?

```
/* Our first program */  
#include <stdio.h>  
int main() {  
printf( "\nHello World!\n" );  
return 0;  
}
```

Without indentation

printf



The diagram shows the code `printf(“\n Hello World! \n”);` enclosed in a rectangular box. Three arrows point to specific parts of the code: one from the text 'Function name' to the `printf` text, one from the text 'Start at a new line' to the first `\n` inside the quotes, and another from the text 'Start at a new line' to the second `\n` inside the quotes.

```
printf( “\n Hello World! \n” );
```

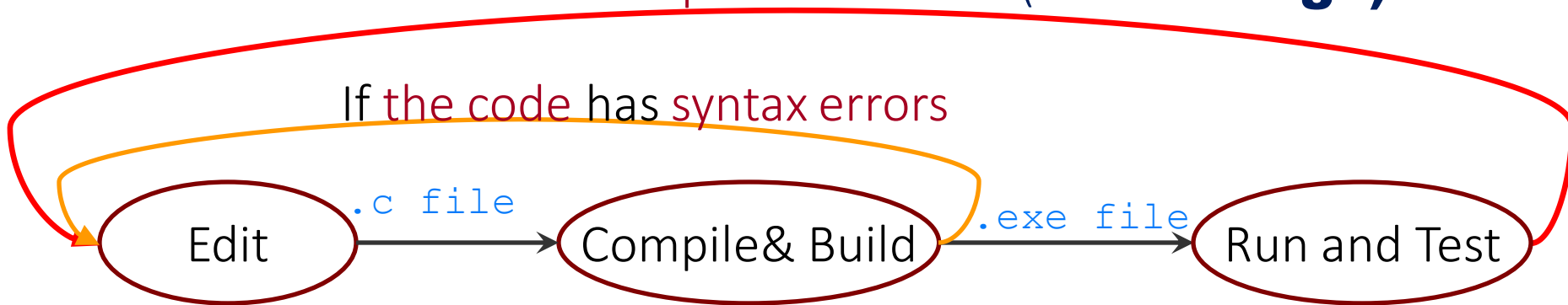
- Function name
- Output information to screen in the described format

Output:

Hello World!

How Does A Program Work (for a single source file)

If the output is incorrect (Possible **Bugs**)



```
/* Our first program */
#include <stdio.h>
int main(){
    printf( "\nHello World!\n" );
    return 0;
}
```

Source file
(sample.c)

```
00001111  11111111
01011111  00010101
01010101  01010010
01011111  00010101
```

Executable file
(sample.exe)

Class Exercises

What is the output of this program?

```
/* just an example */  
#include<stdio.h>  
int main() {  
    printf("Hello World\n"); // 1st  
    printf("Hello World\n"); // 2nd  
    printf("Hello World\n"); // 3rd  
    return 0;  
}
```

HelloWorld1.c

Class Exercises

What is the output of this program?

```
/* just an example */  
#include<stdio.h>  
int main() {  
    printf("Hello World\nHello World\nHello  
    World\n");  
    return 0;  
}
```

HelloWorld2.c

Class Exercises

Can you guess what the output of this program is?

```
#include<stdio.h>
int main() {
    int value1, value2, sum;
    value1 = 50;
    value2 = 25;
    sum = value1 + value2;
    printf("The sum of %d and %d is %d\n", value1, value2, sum);
    return 0;
}
```