Foundations of C Programming
- Programming Language

Outline

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

Natural Language

1. U Col egellloc 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 egellloc 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)
  - Semantics
- You can invent a programming language too!

Programming Language vs. Natural Language

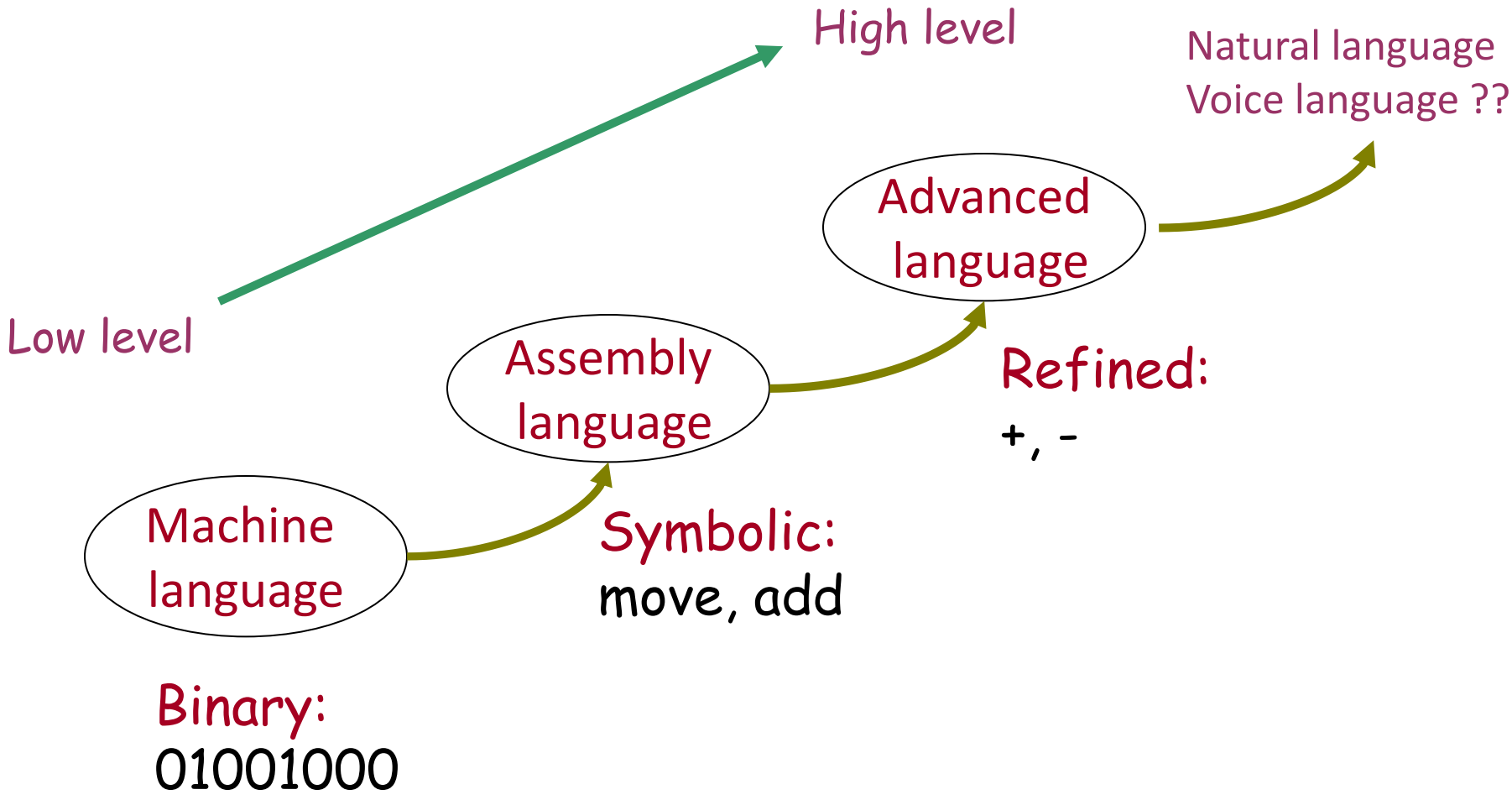
Natural language

Articles
Documents

Programming language

Programs

Evolution of Programming Language

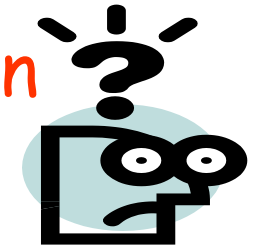


Machine Language

- A CPU accepts instructions in a 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



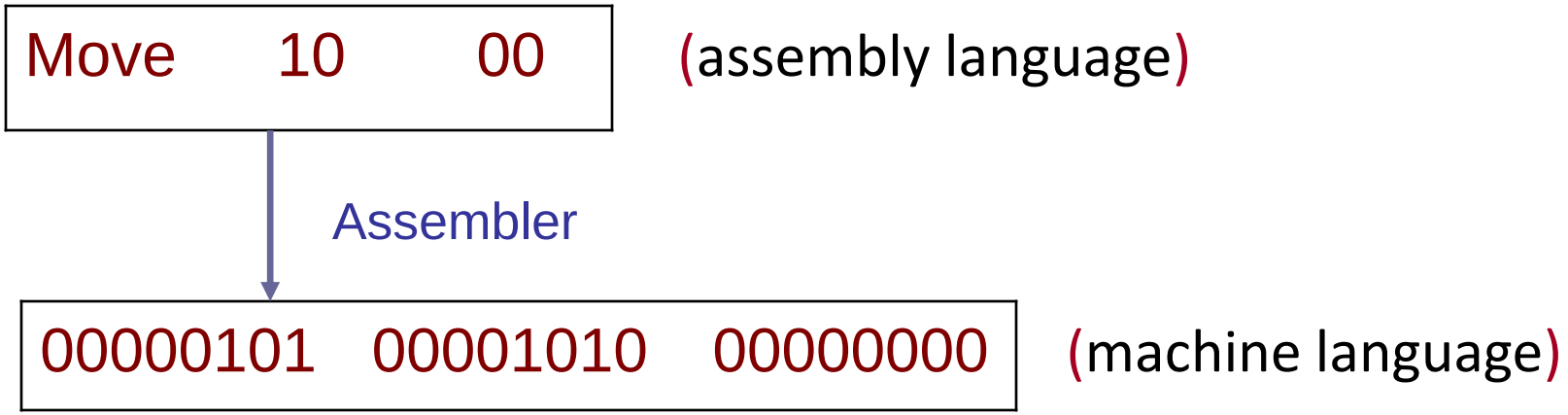
Machine Language

- 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 translates 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,…

```
if (x > 0)
   y = y +1;
else
   y = y - 1
```

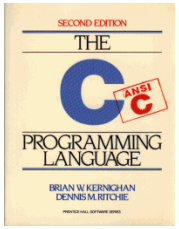
(high level language)


```
00001111    1111111    10100101
01011111    00010101   01010101
01010101    01010010   00000000
01011111    00010101   01010101
00001111    1111111    10100101
00001111    1111111    10100101
```

(machine language)

C Programming Language

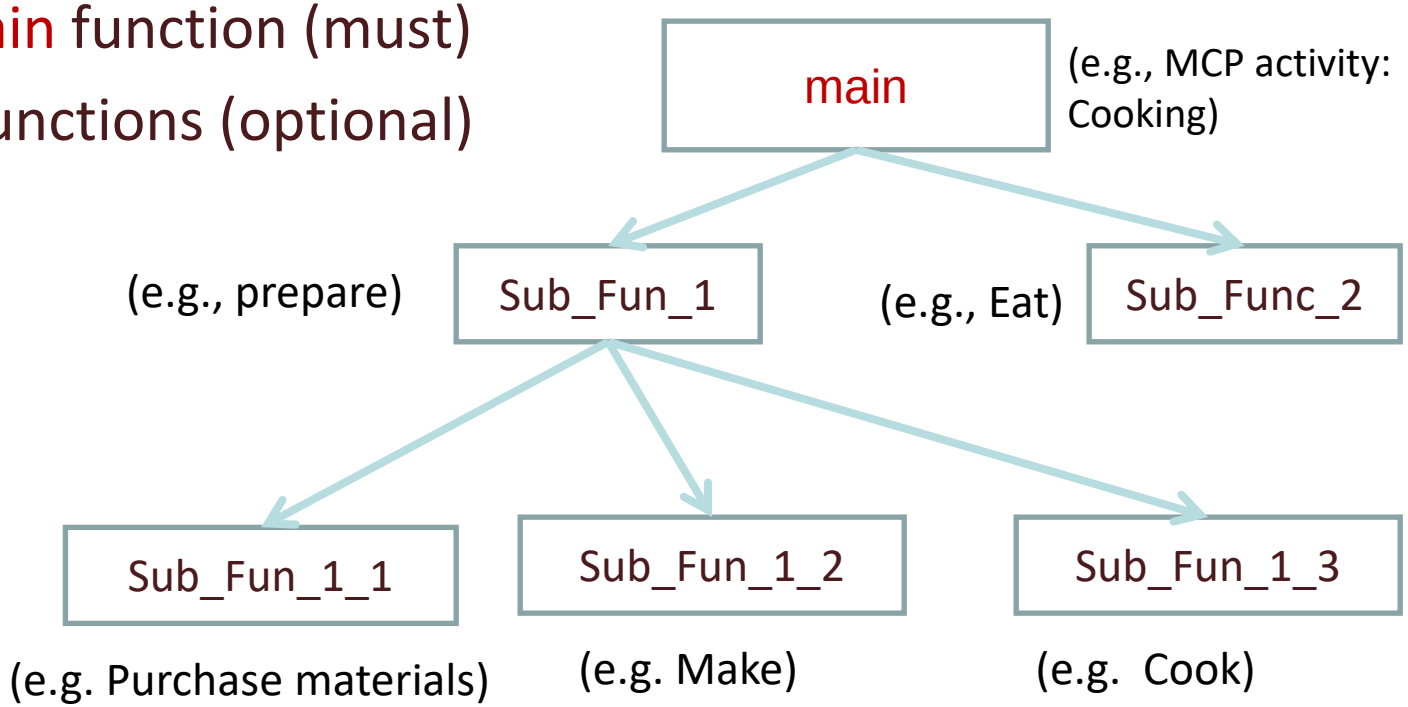
- C is a high level language
- Created by Dennis M. Ritchie in 1972
  - First textbook: K&R, *The C Programming Language*.



- ANSI (American National Standards Institute)
  - 1st edition: ANSI C 1983 (C89)
  - 2nd edition: ANSI C 1990 (C90)
  - 3rd edition: ANSI C 1999 (C99)
  - 4th edition ANSI C 2011 (C11)

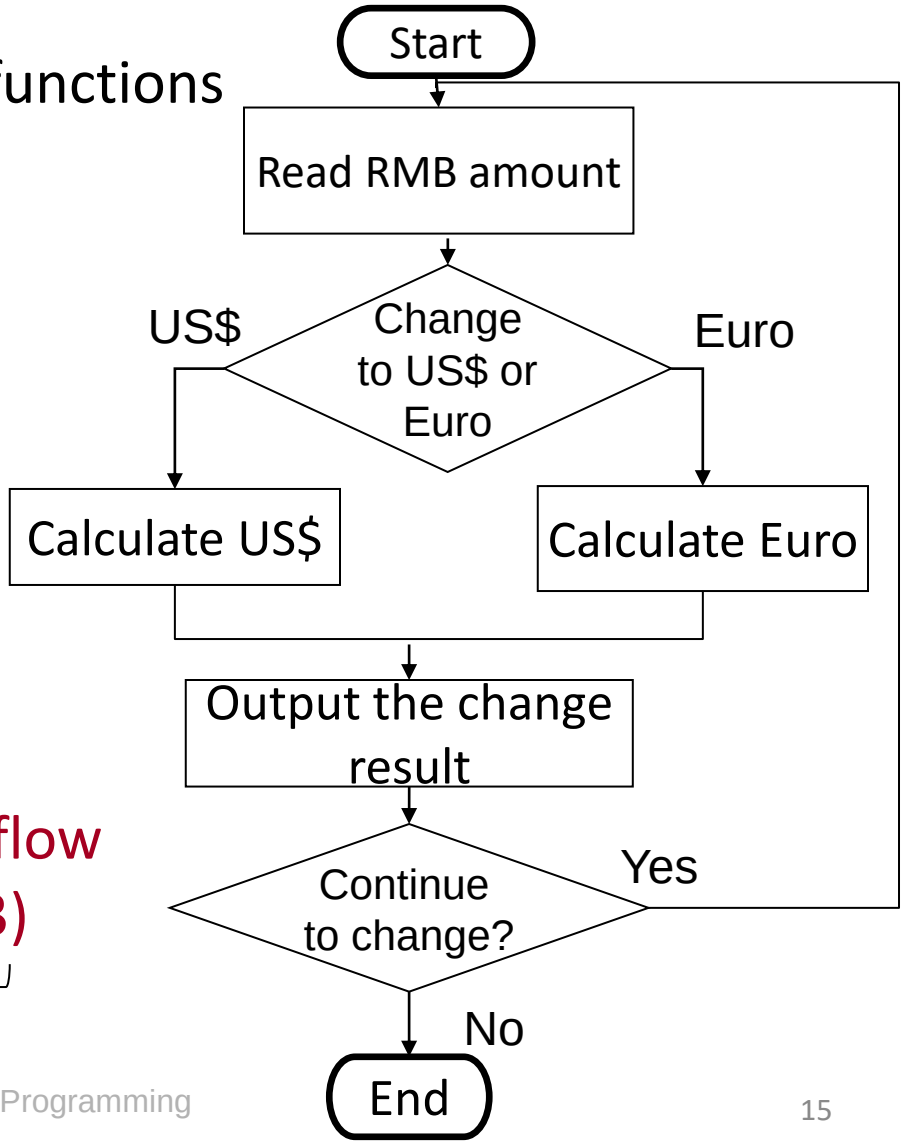
Structured Programming

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



Structure in a Function

- A C program is composed of functions
- Structures in a function
  - Sequence
  - Decisions
  - Loops
- Three structures can be embedded in each other



The structure in the program represented by the right control flow diagram: loop(s1, s2(decision), s3)

sequence

Our First Program

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

Guess what this program wants to do

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

Attention: In CRA rubrics, comment is one evaluation item.

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.

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 and conditions
- How many statements in this program?

Indentation: Which one looks clearer?

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

With indentation

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

Without indentation

Attention: In CRA rubrics, program style is one evaluation item.

printf

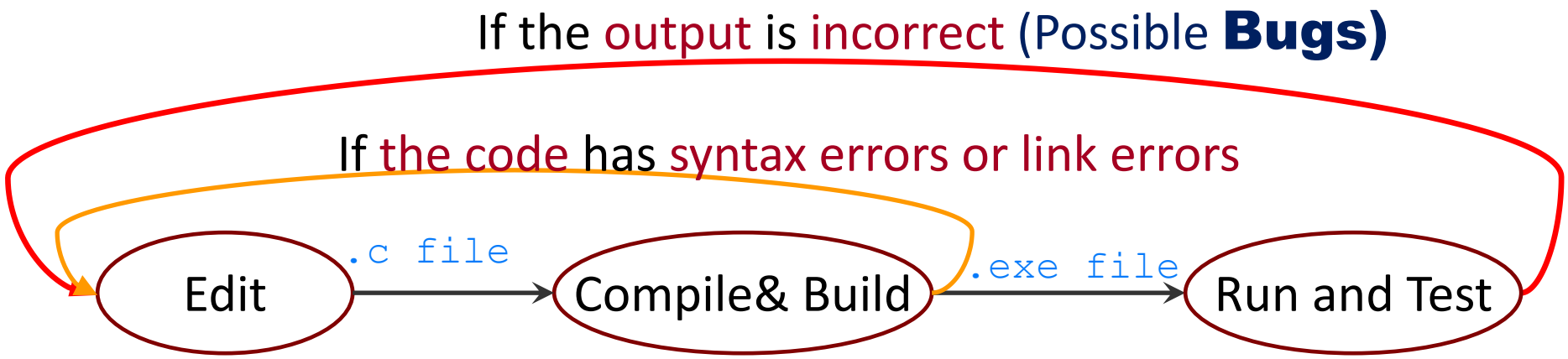
- Function name
- Output information to screen in the described format

Output:

```
Hello World!
```

Attention: " " (English mode) and "" (Chinese mode) are different. C program accepts English mode.

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




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

Source file
(sample.c)

```
00001111    1111111
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");
  printf("Hello World\n");

  printf("Hello World\nHello World\n");

  return 0;
}
```

HelloWorld.c

Summary

- History of C language
- Simple examples
- Process for a program to work
- Basic structures of C programs