1 - Introduction

1.1 Programming Language

• Programming language: a formal language that specifies a set of instructions for a computer to implement specific algorithms to solve problems

High-level program

Eg: C, Java, Python, ECMAScript

```
int i, a = 0;
for (i=1; i<=10; i++) {
        a = a + i*i;
}</pre>
a = 0
for i in range(1,11):
        a = a + i*i
}
```

Low-level program

Eg: MIPS (IT5002)

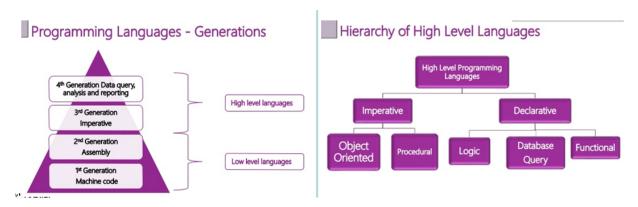
```
addi $t1, $zero, 10
add $t1, $t1, $t1
addi $t2, $zero, 10
Loop: addi $t2, $t2, 10
addi $t1, $t1, -1
beq $t1, $zero, Loop
```

Machine code

Computers can execute only machine code directly.

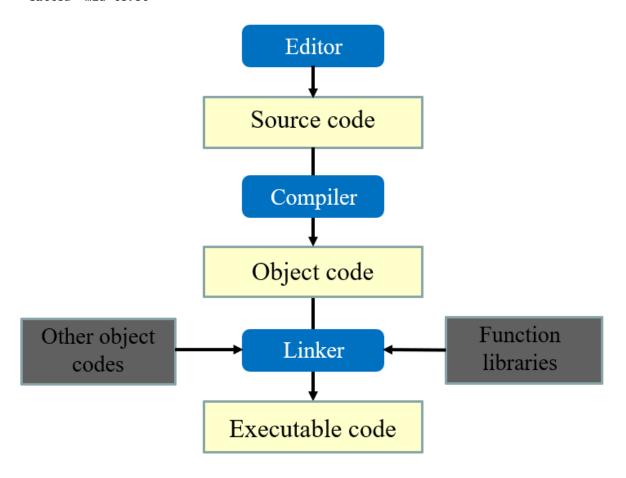
- 1st Generation languages:
 - Machine language
 - o Directly executable by machine
 - Machine dependent
 - Efficient code but difficult to write
- 2nd generation languages:
 - o Assembly language
 - Need to be translated(assembled) into machine code for execution
 - o Efficient code, easier to write than machine code
- 3rd generation language:
 - o Closer to English
 - Need to be translated (complied or interpreted) into machine code for execution
 - Example: FORTRAN, COBOL, C, BASIC
- 4th generation language:
 - Require fewer instructions than 3GL
 - Used with databases (query languages, report generators, forms designers)
 - o Example: SQL, PostScript, Mathematica
- 5th generation language:
 - a Head mainly AT masaanch

- o Declarative languages
- Functional languages (Lisp, Scheme, SML)
- Logic programming (Prolog)
- "Generational" classification of high level languages (3GL and later) was never fully precise
- A different classification is based on paradigm



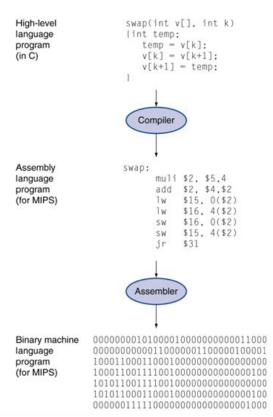
1.1.1 C Programming Language

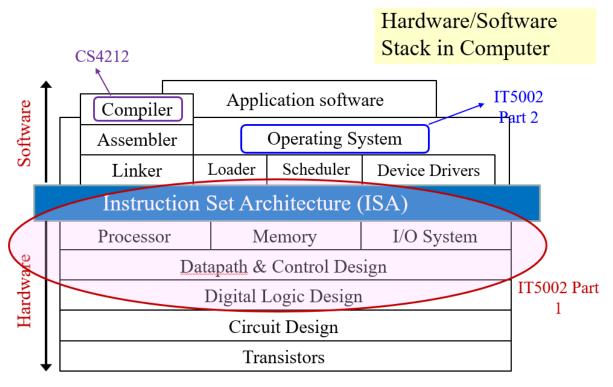
- C is an imperative procedural language (命令式程序语言)
- C provides constructs that map efficiently to typical machine instructions
- C is a high-level language very close to the machine level, hence sometimes it is called "mid-level"



1.2 Abstraction

- High-level language
 - Level of abstraction closer to problem domain
 - Provides productivity and portability
- Assembly language
 - Textual and symbolic representation of instructions
- Machine code (object code or binary)
 - Binary bits of instructions and data





Level of Representation

```
temp = v[k];
                                     v[k] = v[k+1];
     High Level Language
        Program (e.g., C)
                                     v[k+1] = temp;
                  Compiler
                                           $t0, 0($2)
                                      lw
     Assembly Language
                                           $t1, 4($2)
                                           $t1, 0($2)
$t0, 4($2)
        Program (e.g., MIPS)
                                      SW
                  Assembler
      Machine Language
                                    0000 1001 1100 0110 1010 1111 0101 1000
         Program (MIPS)
                                    1010 1111 0101 1000 0000 1001 1100 0110
                                    1100 0110 1010 1111 0101 1000 0000 1001
                                    0101 1000 0000 1001 1100 0110 1010 1111
Machine
Interpretation
                                               00 00 00 00 00
                                               04 00 00 00 00
Hardware Architecture Description
                                               08 00 00 00 00 D
   (Logic, Logisim, etc.)
Architecture
Implementation
   Logic Circuit Description
      (Logisim, etc.)
```