1. Introduction to Computers

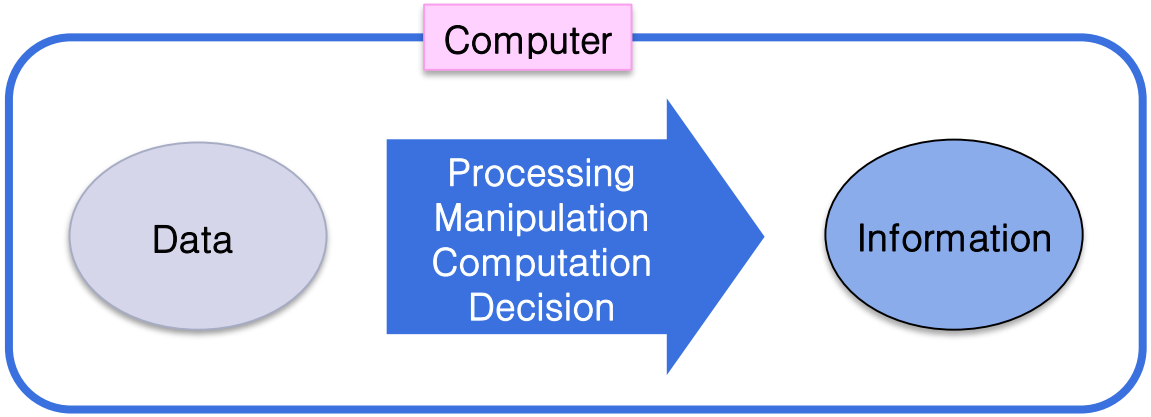
C Programming

Agenda

- Computer Systems
- Programming Languages
- Creating and Running Programs
- Algorithm

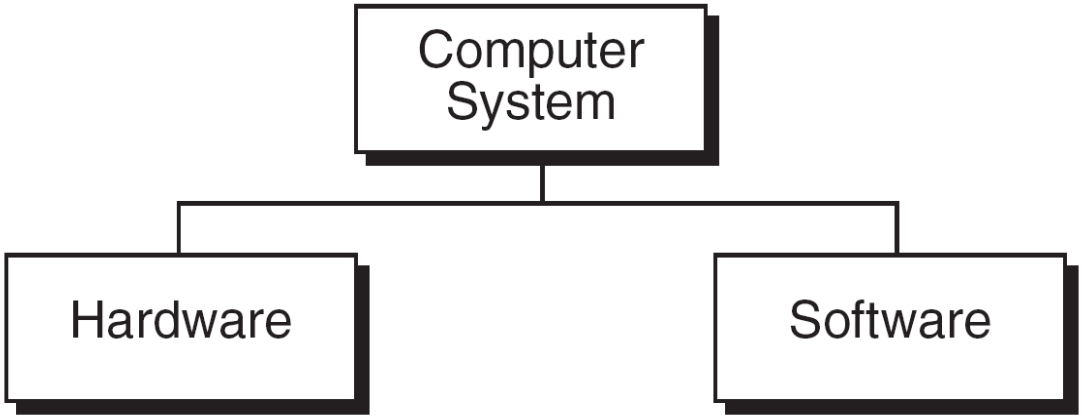
What is a Computer?

- A computer is a machine for manipulating data according to a list of instructions known as program.
  - Computations
  - Making logical decisions
  - ➔ Universal information-processing machines

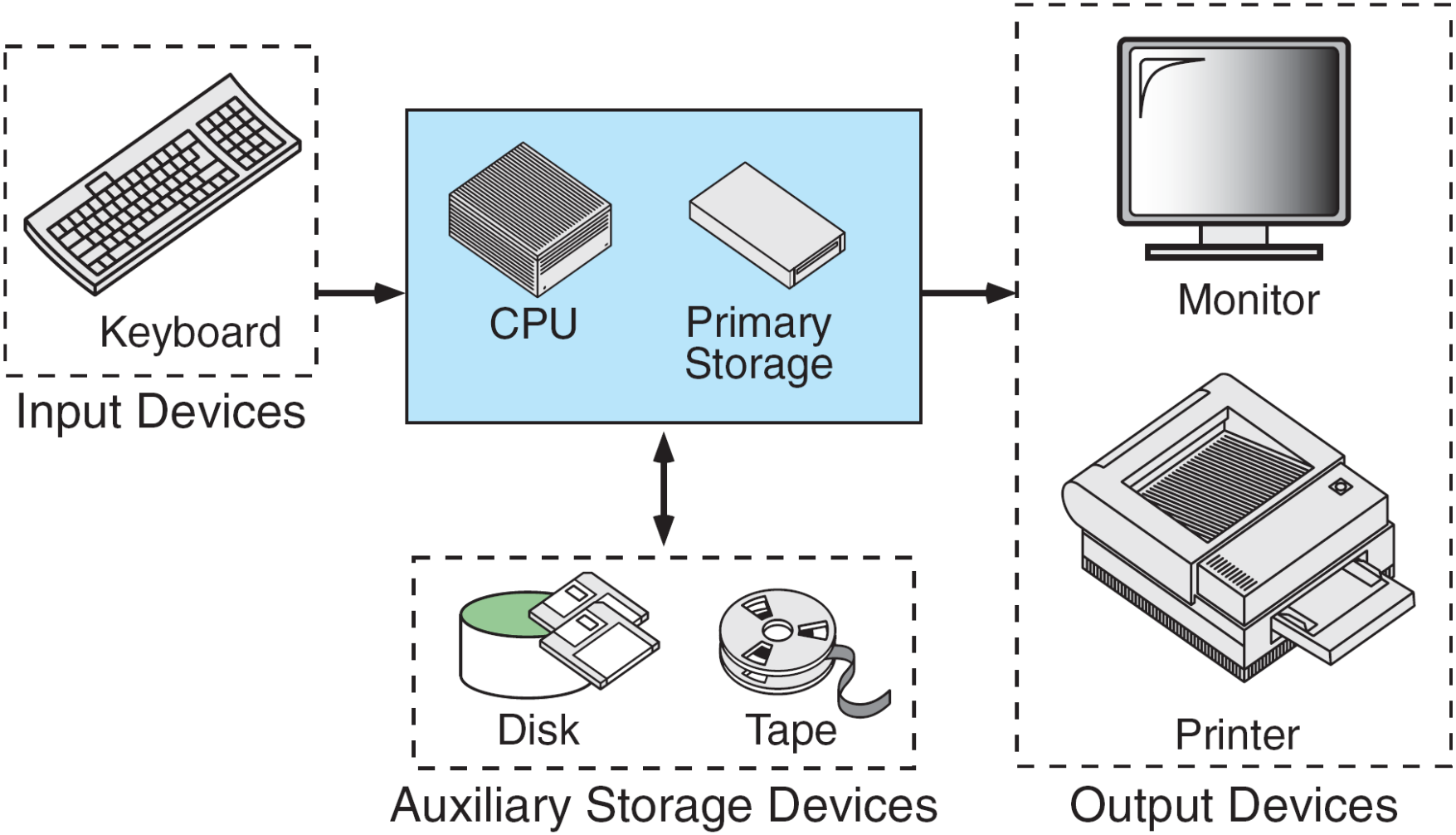


Computer System

- Computer system = hardware + software
  - Hardware: physical components of computer (machine)
  - Software: programs (+ α) that enable a computer to perform a specific task

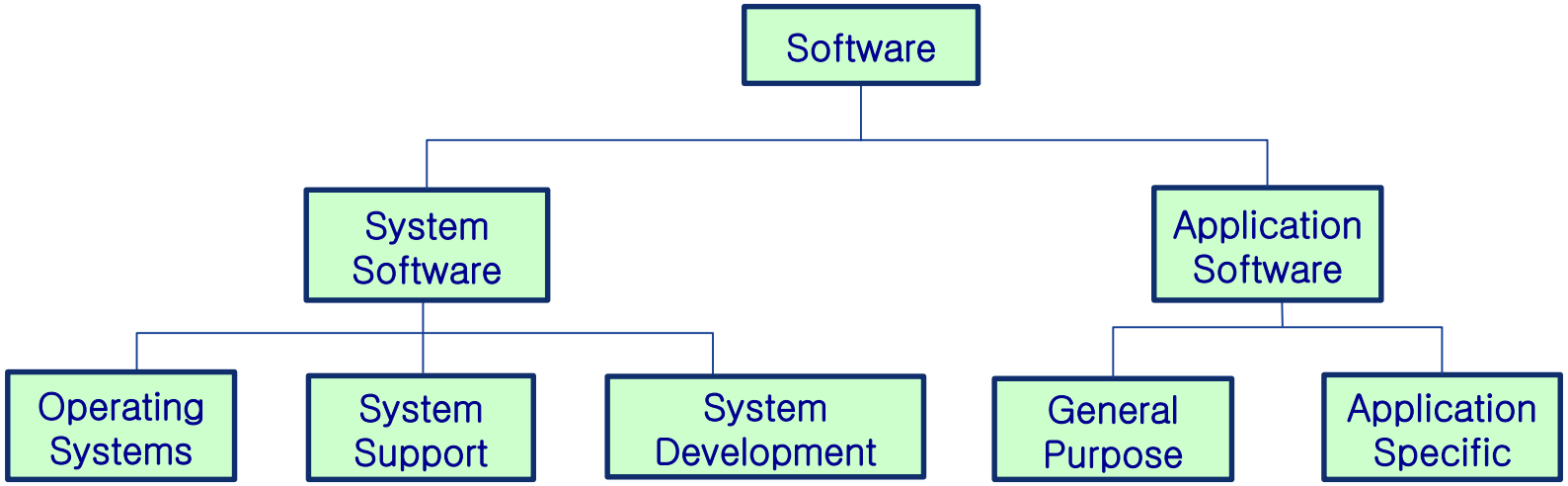


Computer Hardware

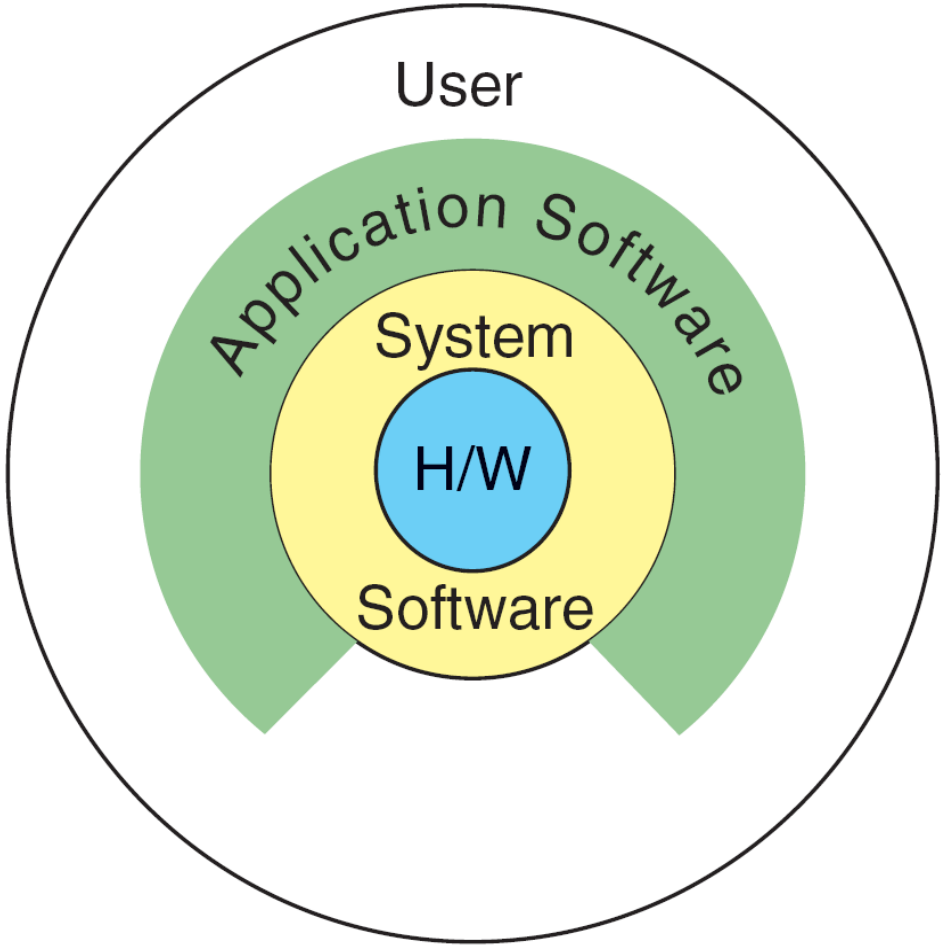


Computer Software

- Categories of software
  - Application software: software to do valuable tasks
    - Business software, educational software, medical software, databases, computer games, …
  - System software: software that provides environment to develop or execute software



Layers of Computer System

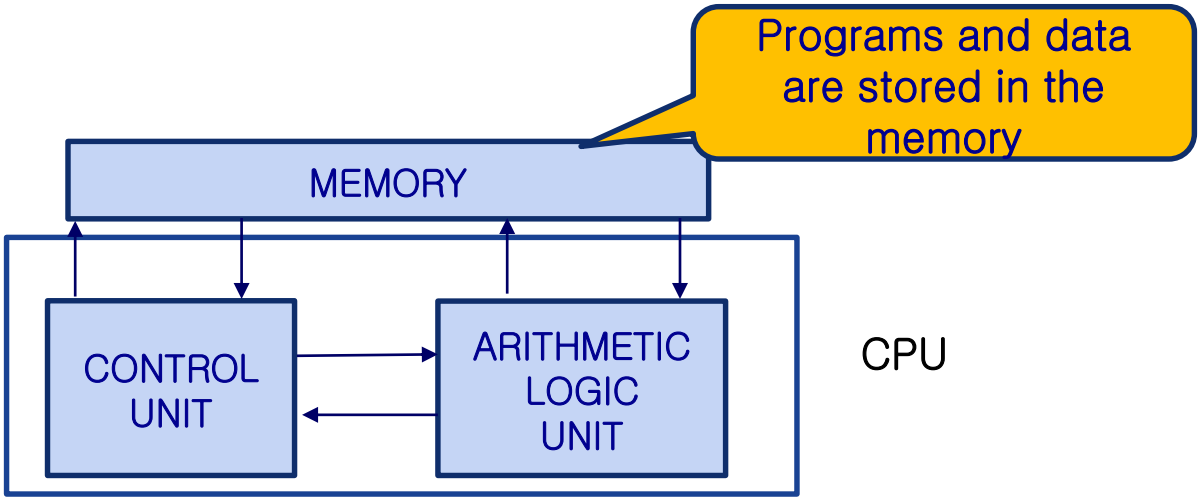


Agenda

- Computer Systems
- Programming Languages
- Creating and Running Programs
- Algorithm

Program

- A set (or sequence) of instructions for computer to execute
- We can specify function of a computer by writing proper programs



Programming Language

- Programming language: artificial language to write computer programs
  - A method to specify the works a computer should do.

- Types of programming languages
  - Machine languages
  - Assembly languages
  - High-level languages
    - C, C++, Java, C#, …
    - Basic, Pascal, Fortran, Cobol, …

Machine Language

- Machine languages
  - Consist of streams of 0's and 1's (machine instructions)
  - The only languages understood by computer hardware

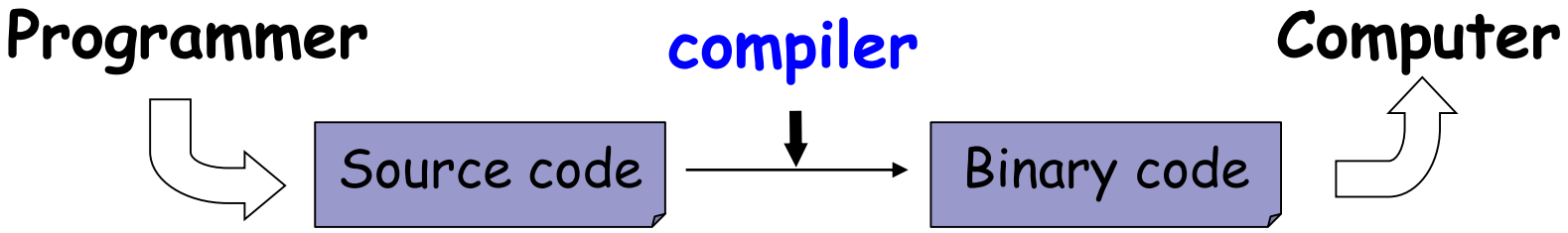
→ Too difficult to write or understand

```
          00000000 00000100 0000000000000000
01011110  00001100 11000010 0000000000000010
          11101111 00010110 0000000000000101
          11101111 10011110 0000000000001011
11111000  10101101 11011111 0000000000010010
          01100010 11011111 0000000000010101
11101111  00000010 11111011 0000000000010111
11110100  10101101 11011111 0000000000011110
00000011  10100010 11011111 0000000000100001
11101111  00000010 11111011 0000000000100100
01111110  11110100 10101101
11111000  10101110 11000101 0000000000101011
00000110  10100010 11111011 0000000000110001
11101111  00000010 11111011 0000000000110100
          01010000 11010100 0000000000111011
                   00000100 0000000000111101
```

High-Level Languages

- More human-friendly programming language
  Ex) C/C++, Java, C#, Pascal, Basic, Python, …
  - Easy to write and read program

- To be executed, programs in high-level language (source code) should be translated into machine language (binary code) by compiler



High-Level Languages

```
1  /* This program reads two integers from the keyboard
2     and prints their product.
3        Written by:
4        Date:
5  */
6  #include <stdio.h>
7
8  int main (void)
9  {
10 // Local Definitions
11    int number1;
12    int number2;
13    int result;
14
15 // Statements
16    scanf  ("%d", &number1);
17    scanf  ("%d", &number2);
18    result = number1 * number2;
19    printf ("%d", result);
20    return 0;
21 }  // main
```

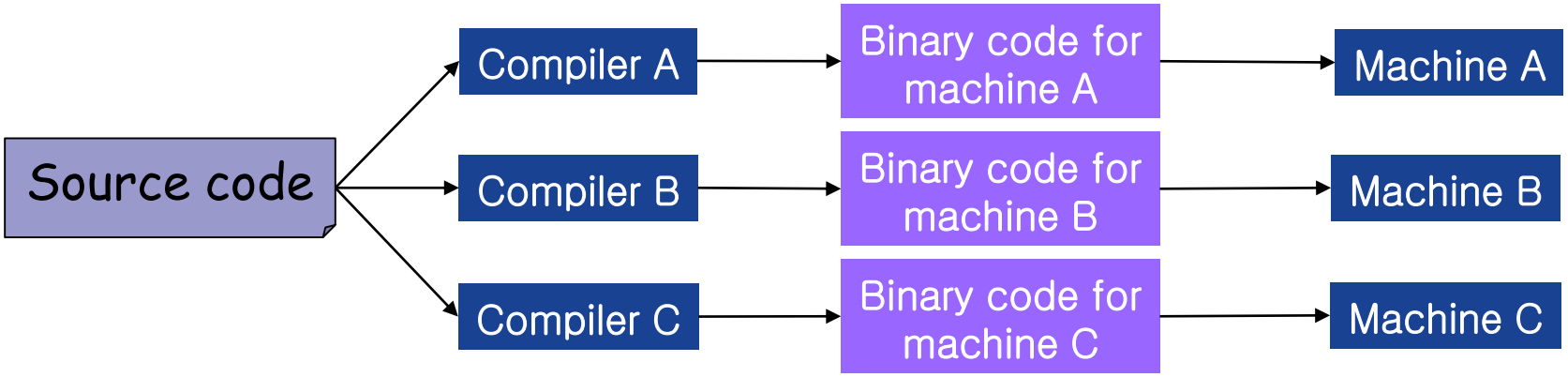
High-Level Languages

```
>>> money = 2000
>>> if money > 1000:
        print("I am rich!!")
 else:
        print(I am not rich!!")
        print("But I might be later…")
```

High-Level Languages

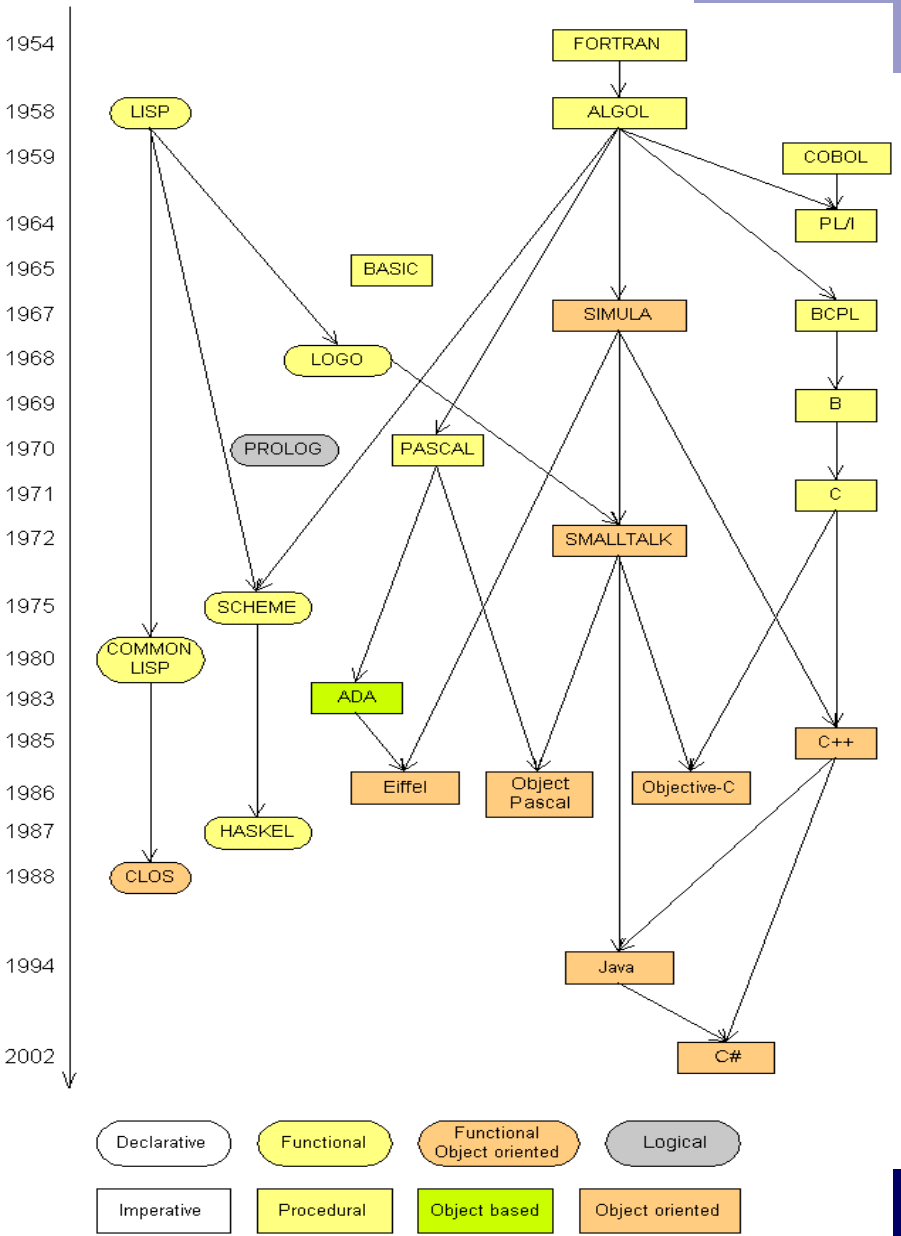
- Advantages of high-level language
  - Programming efficiency (easy to program)
  - Programmers can concentrate on application problem
  - Portability

    Cf. machine languages depend on hardware
  - ETC.



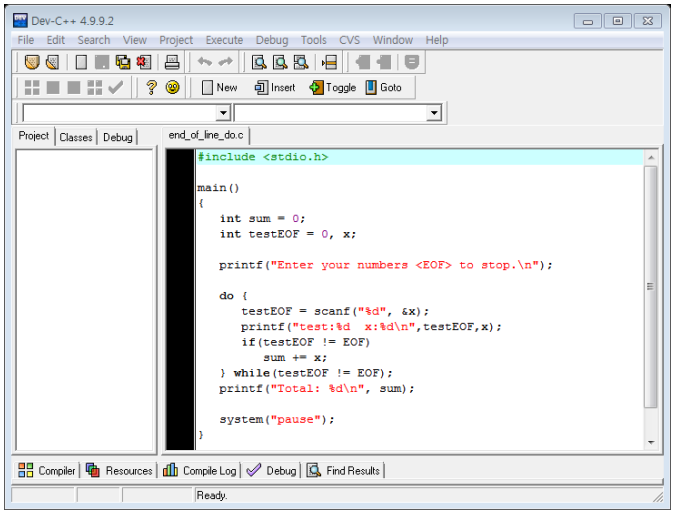
Languages

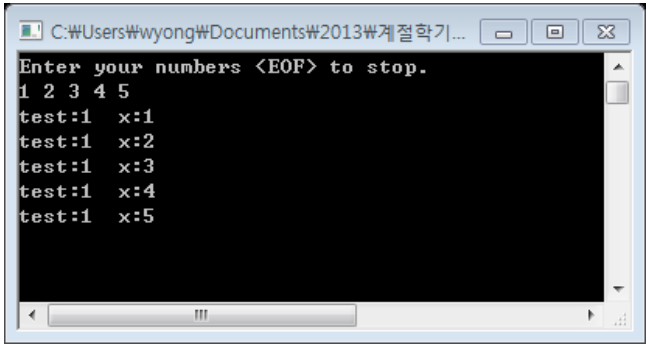
- Some influential ones:
  - FORTRAN
    - science / engineering
  - COBOL
    - business data
  - LISP
    - logic and AI
  - BASIC
    - a simple language



Programming basics

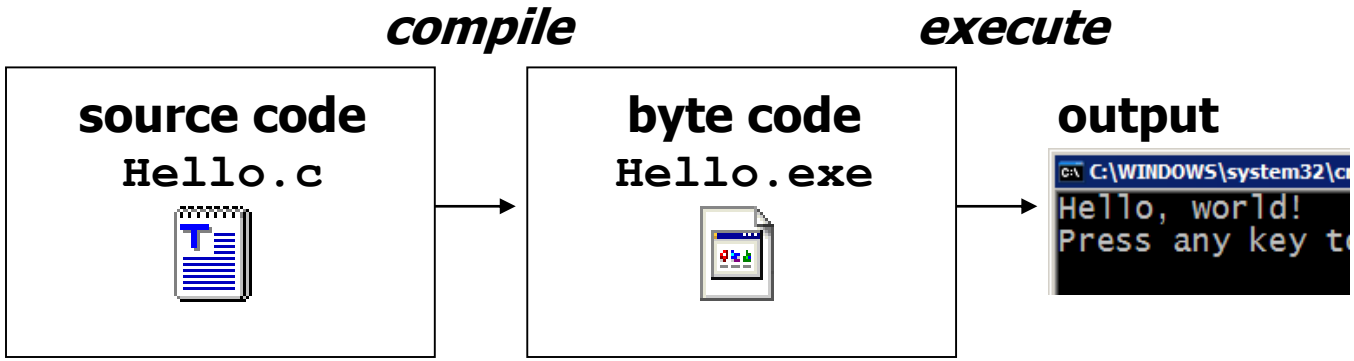
- code or source code: The sequence of instructions in a program.
- syntax: The set of legal structures and commands that can be used in a particular programming language.
- output: The messages printed to the user by a program.
- console: The text box onto which output is printed.
  - Some source code editors pop up the console as an external window, and others contain their own console window.



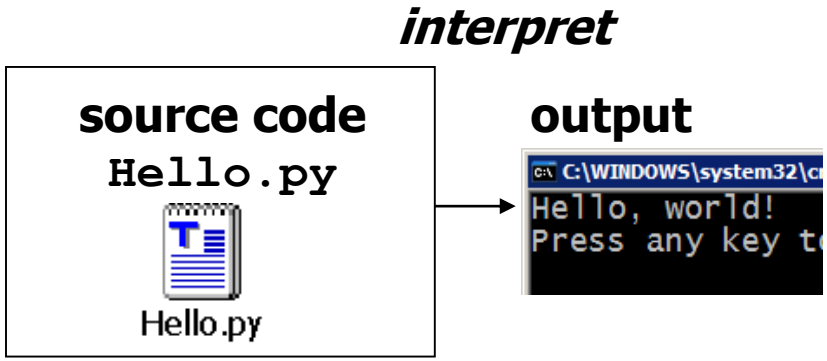


Compiling and interpreting

- Many languages require you to *compile* (translate) your program into a form that the machine understands.



- Python is instead directly *interpreted* into machine instructions.

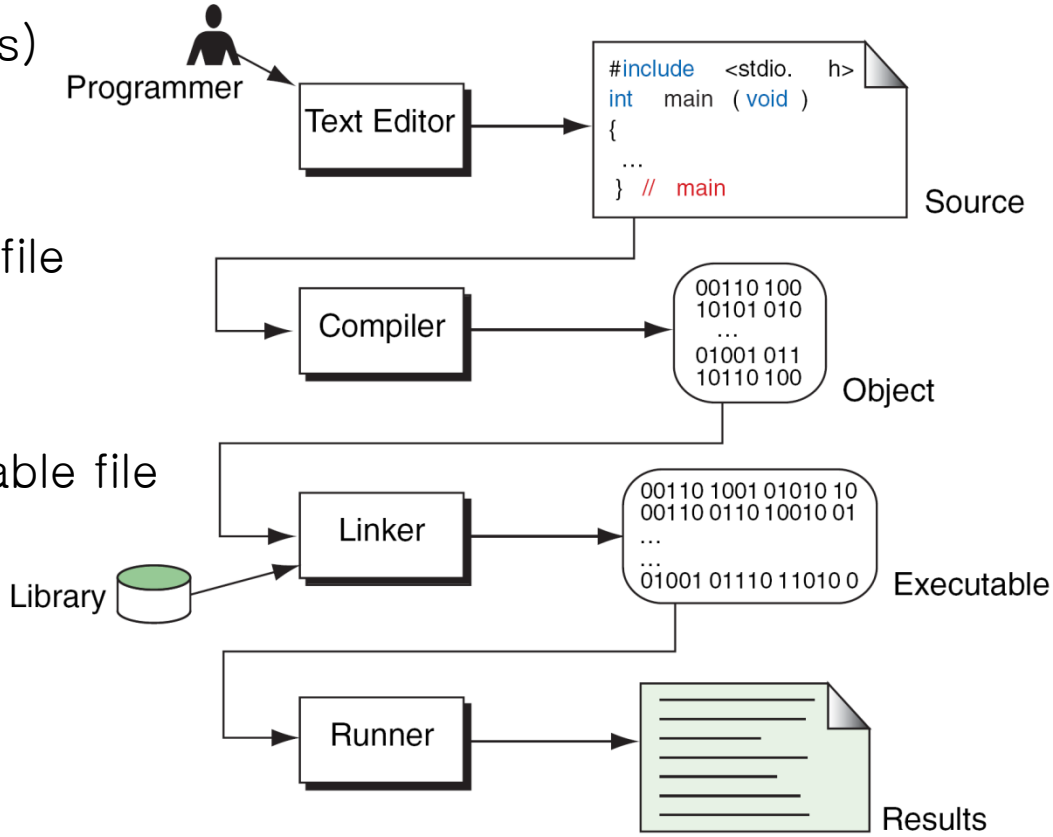


Agenda

- Computer Systems
- Programming Languages
- <u>Creating and Running Programs</u>
- Algorithm

Creating and Running Programs

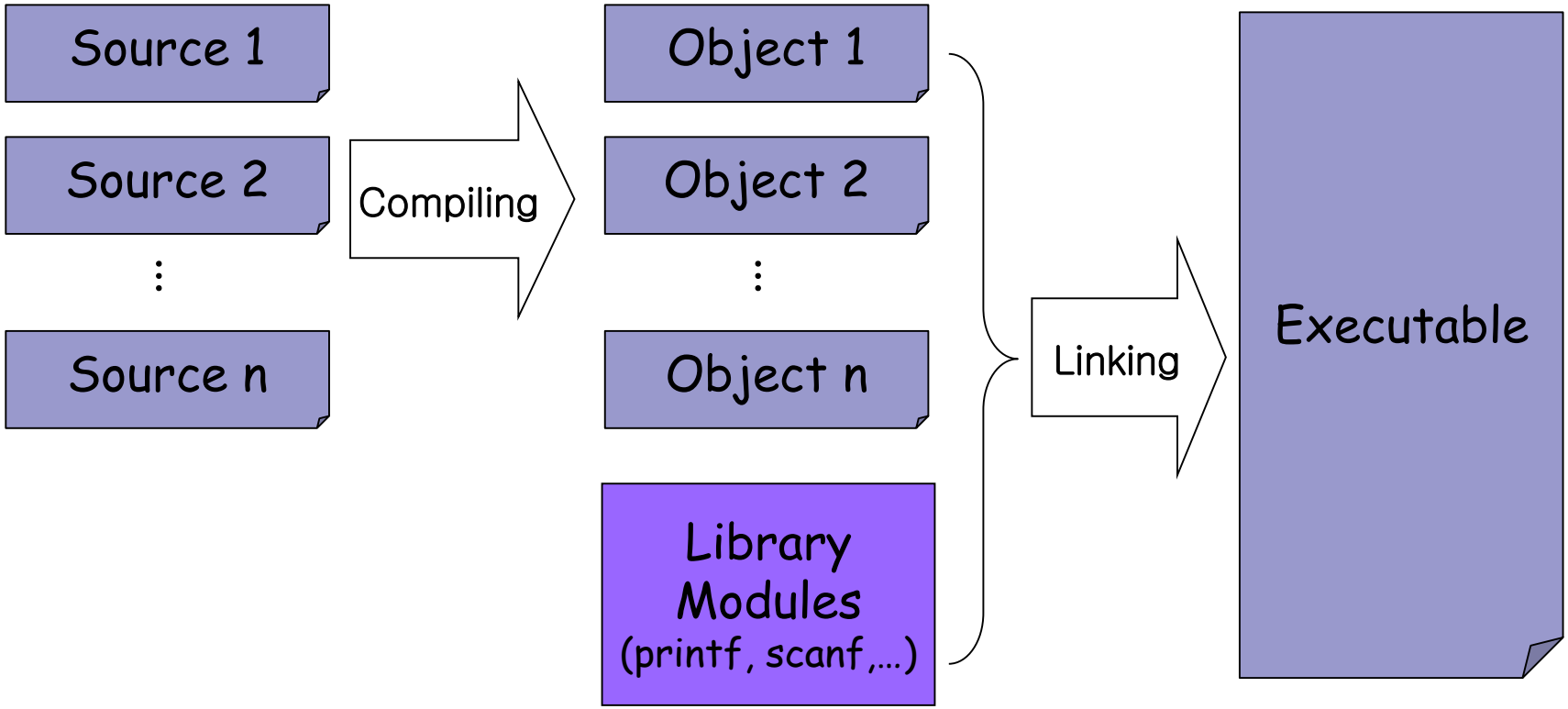
- Process to write and execute a program
  1. Write program
     - Stored in source file(s)
  2. Compile
     - Source file → object file
  3. Link
     - Object file → executable file
  4. Execute



Creating and Running Programs

- Link
  - Integrating objects and library modules required to execute

Notice! a program can be distributed in multiple source files



Agenda

- Computer Systems
- Programming Languages
- Creating and Running Programs
- Algorithm

General Steps of Program Development

1. Understand the problem
   - Clarify exact purpose and goal

2. Develop the solution
   - Design and describe solution in a way which is easy to write and understand

   "Resist the temptation to code"

3. Write the solution in programming language
   - Implement the solution in programming language

4. Test the program

Algorithm

- Computing problems can be solved by executing a series of actions in a specific order

- Algorithm: procedure in terms of
  - Actions to be executed
  - The order in which these actions are to be executed

  Ex) *Rise-and-shine* algorithm

  Get out of bed
  Take off pajamas
  Take a shower
  Get dressed
  Eat breakfast
  Carpool to work

➔ A program can be regarded as an algorithm written in a programming language

Example

- How to put an elephant into a refrigerator?

- Sort following integers in ascending order.
  27, 94, 1, 588, 38, 63