1. Introduction to Computers

[ECE10002/ITP10003] C Programming

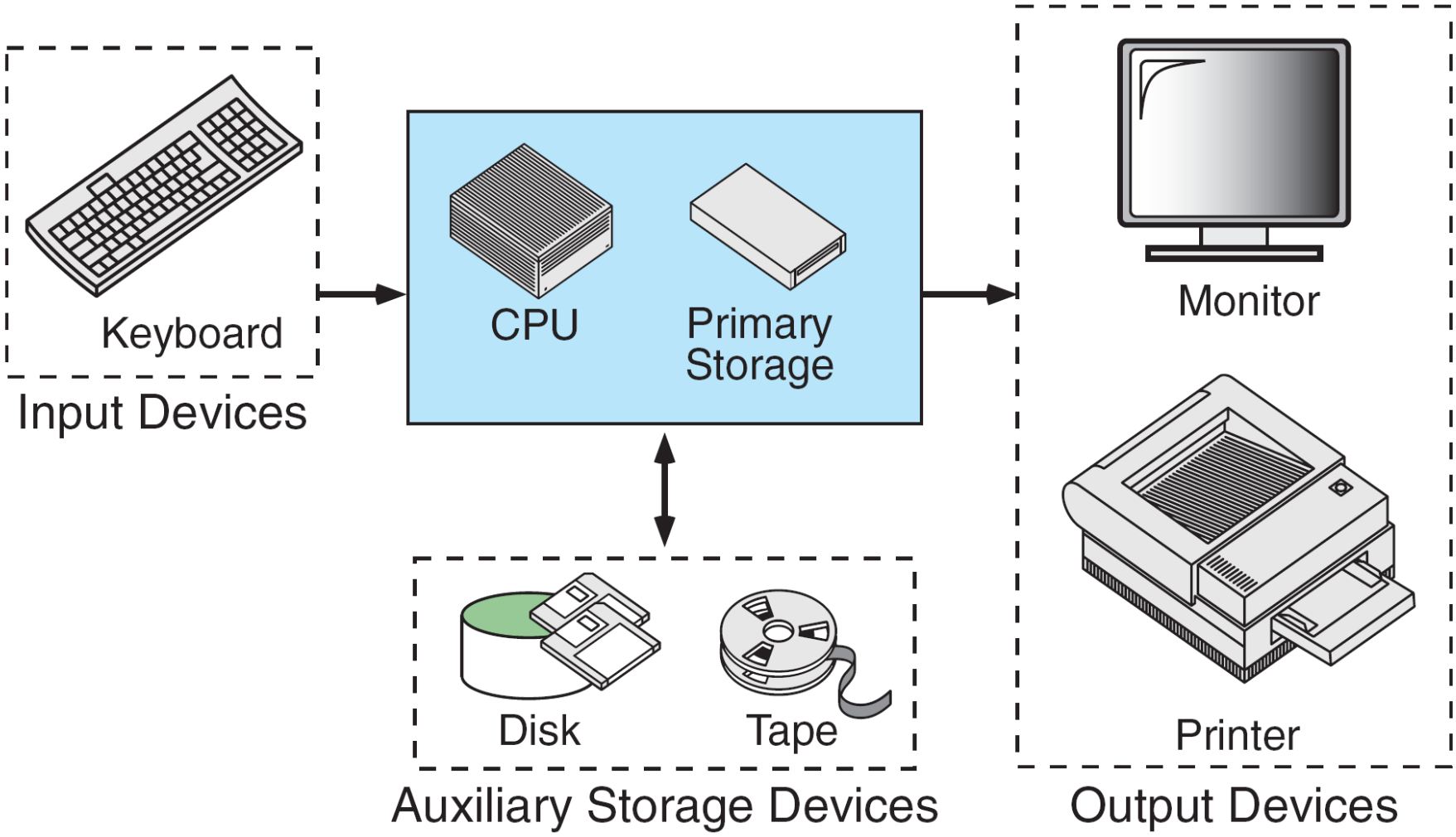
Announcement

- Lab on next class.
  - Bring your laptop computer with fully charged battery
    - Install VS-Code before the class.

Agenda

- Computer Systems
- (Stored Program Architecture)
- Programming Languages
- Creating and Running Programs
- Algorithm

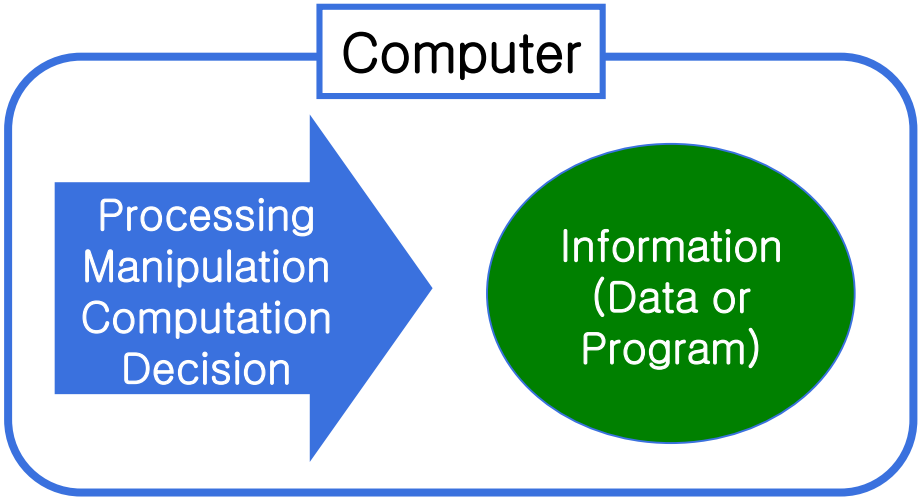
Computer H/W


Keyboard
Input Devices
CPU
Primary Storage
Disk
Tape
Auxiliary Storage Devices
Monitor
Printer
Output Devices

What is a Computer?

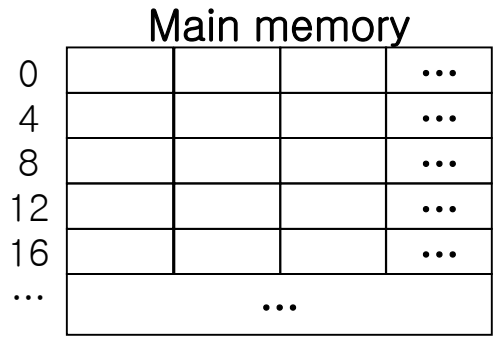
- A computers is a machine for manipulating data according to a list of instructions known as a program
  - Computations
  - Making logical decisions

➔ Universal information-processing machines



Computer Hardware

- CPU
  - Arithmetic operations, logical operations
  - Control
- Main memory
  - List of cells to store data or instruction
  - Each cell is identified by its address.
  - Fast, volatile



- Auxiliary memory
  - SSD, HDD, optical disks (CD,DVD), magnetic tapes, …
  - Large capacity, non-volatile, slow, cheap
- I/O devices: the means by which the computer communicates with the outside world
  - Input devices: keyboard, mouse, scanner, camera, microphone, …
  - Output device; monitor, printer, speaker, …

Agenda

- Computer Systems
- (Stored Program Architecture)
- Programming Languages
- Creating and Running Programs
- Algorithm

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

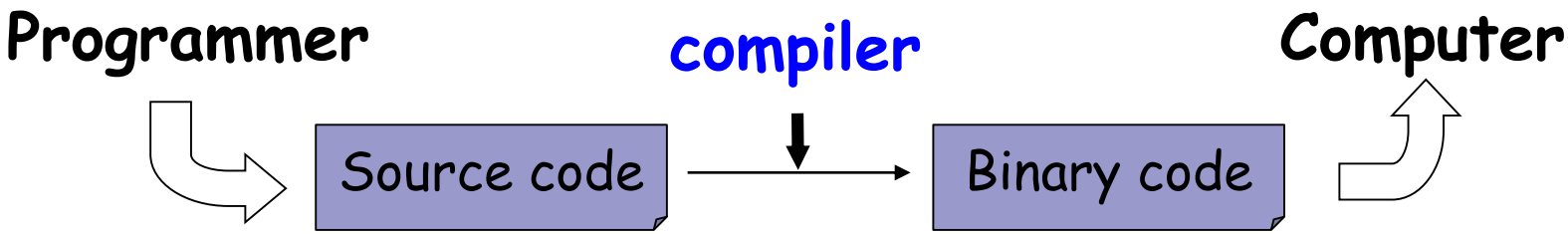
→ Too difficult to write or understand

```
          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

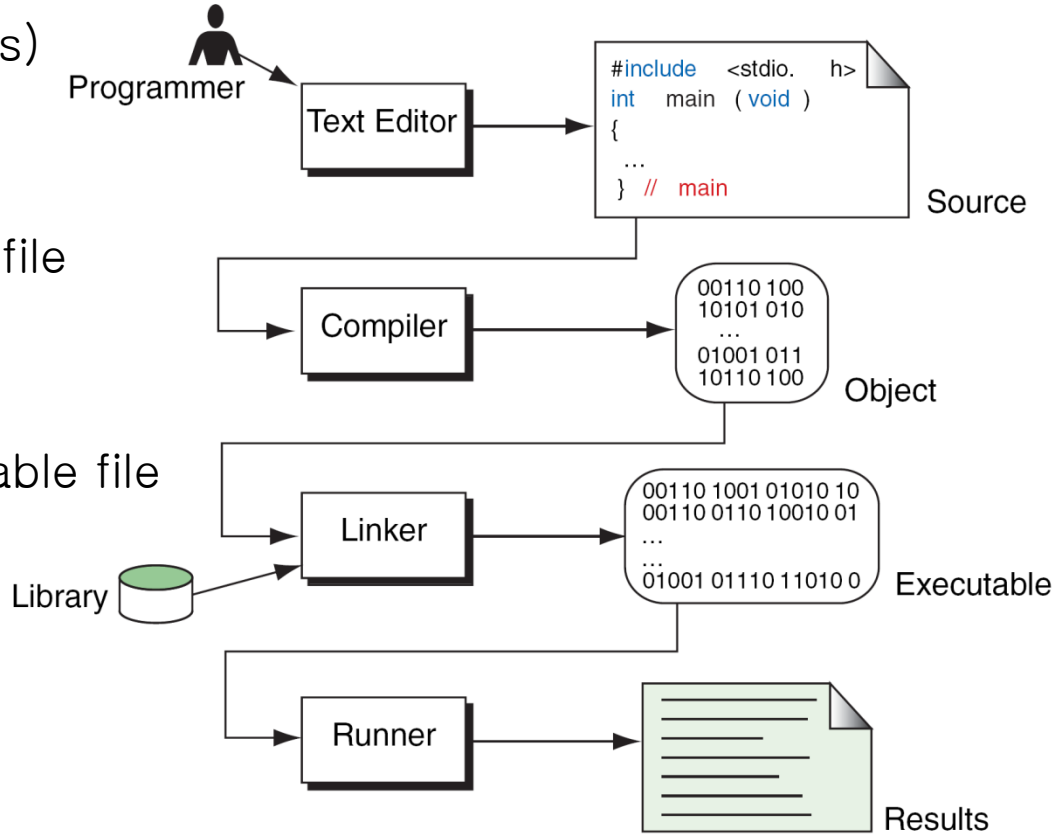
```
1  /* This program reads two integers from the keyboard
2     and prints their product.
3        Written by:
4        Date:
5  */
6  #include <stdio.h>
7
8  int main (void)
9  {
10 // Local Definitions
11    int number1;
12    int number2;
13    int result;
14
15 // Statements
16    scanf  ("%d",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

Agenda

- Computer Systems
- (Stored Program Architecture)
- Programming Languages
- Creating and Running Programs
- Algorithm

Creating and Running Programs

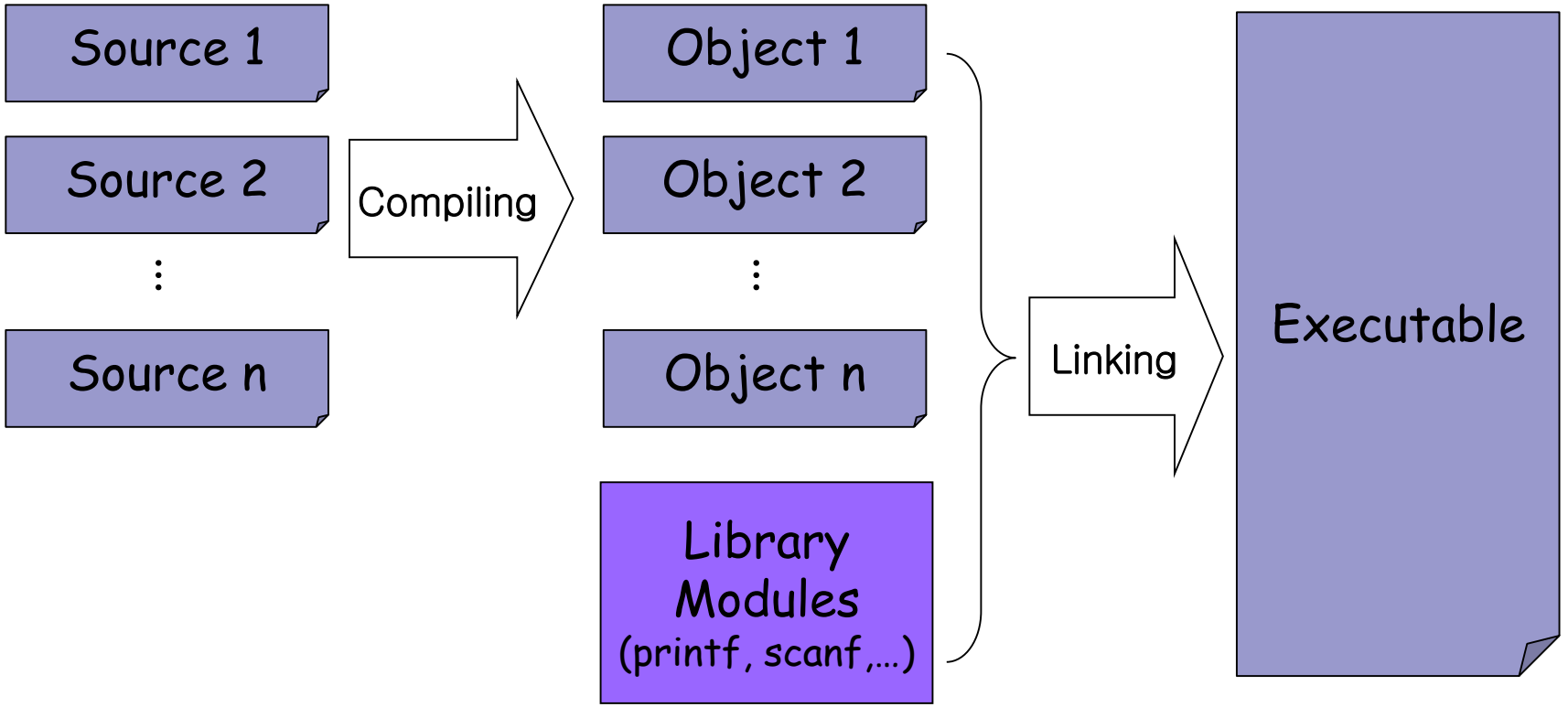
- Process to write and execute a program
  1. Write program
     - Stored in source file(s)
  2. Compile
     - Source file → object file
  3. Link
     - Object file → executable file
  4. Execute



Creating and Running Programs

- Link
  - Integrating objects and library modules required to execute

Note! a program can be distributed in multiple source files.



Agenda

- Computer Systems
- (Stored Program Architecture)
- 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

An Example

- A program is a sequence of instructions each of which specifies an action.

  Ex) Putting an elephant into a refrigerator?

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.

Description of Algorithm

- Programming language
  - Representation including implementation details
- Alternative representation
  - Pseudo code
    - Text-based representation
  - Flow chart
    - Graphical representation

Pseudo Code

- Pseudo code: a description of an algorithm in the form of annotations and informative text written in a natural language.
  - It has no syntax like any of the programming language and thus can't be compiled or interpreted by the computer.

  Ex) Adding two numbers
    - Read two numbers in variable A and B
    - Add A and B, and store the result in C.
    - Print C

- Advantages
  - Easy to read and write.
  - Can be written with any text editor. (as comments)

Flow Chart

- Graphical representation of algorithm
- Drawn using special purpose symbols
  - Parallelogram, rectangles, diamonds, ovals, circles, …