西安交通大學
XI'AN JIAOTONG UNIVERSITY

第四讲：
程序是如何执行的

师斌

School of Computer Science & Technology,

Xi'an Jiaotong University

程序是如何执行的?

Related courses

- Programming Language
- Assembly Language
- Compile
- Operating Systems
- Computer Architecture
- And more...
  - Computer Networking
  - Database
  - Discrete Mathematics
  - Data Structure & Algorithm
  - ...

. . .

主要内容

- **计算机程序：设计与编译**
- **程序如何一步步执行：冯诺依曼计算机**
- **操作系统如何管理程序**

计算机程序- hello world!

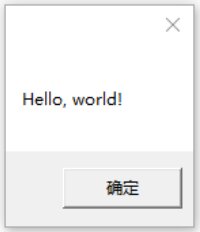
in **JavaScript**

```
<script>
alert("Hello, world!");
</script>
```

Hello, world on web browser

in **VBScript**

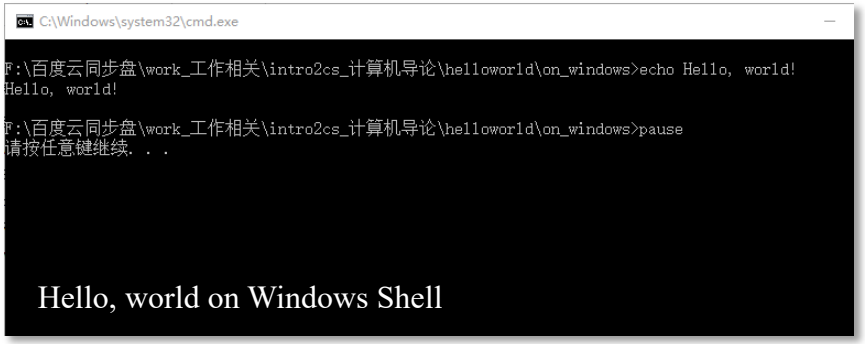
```
MsgBox "Hello, world!"
```



Hello, world on Windows desktop

in **Windows Shell**

```
echo Hello, world!
pause
```



Hello, world on Windows Shell

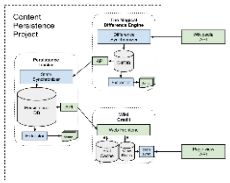
计算机程序- 定义

- A computer program is a **collection of instructions** that performs a specific task when **executed** by a **computer**[1].

Collection Of Instructions

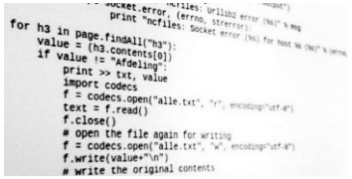
Programming Language

+



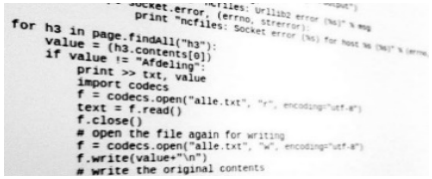
Design

Programming



Source Code

Compile & execute



Source Code

Translate

Native Code / Machine Code

Computer

Hardware

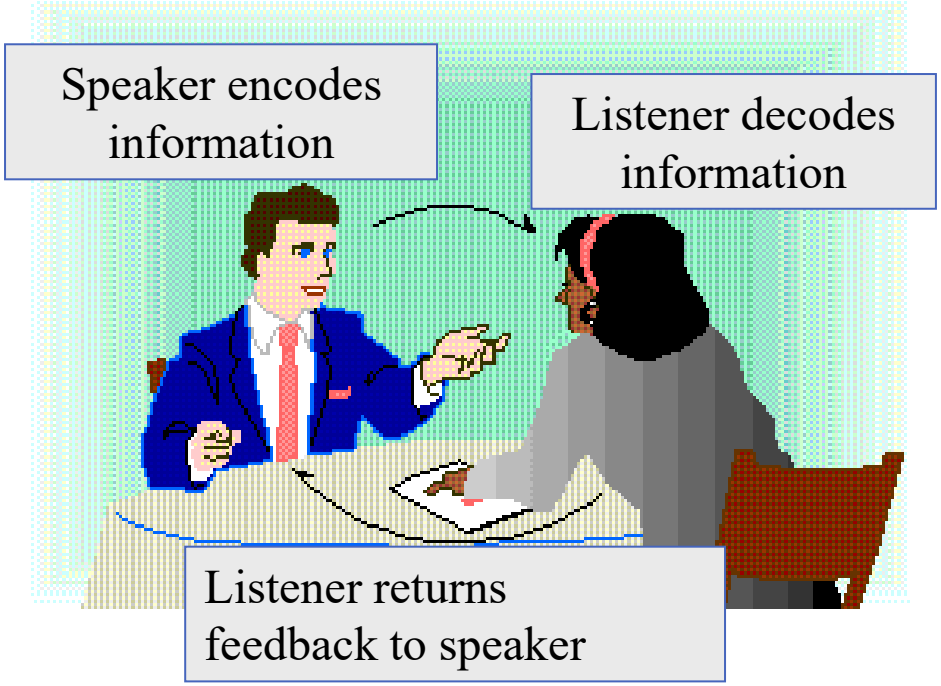
+

Operating System

[1] https://en.wikipedia.org/wiki/Computer_program

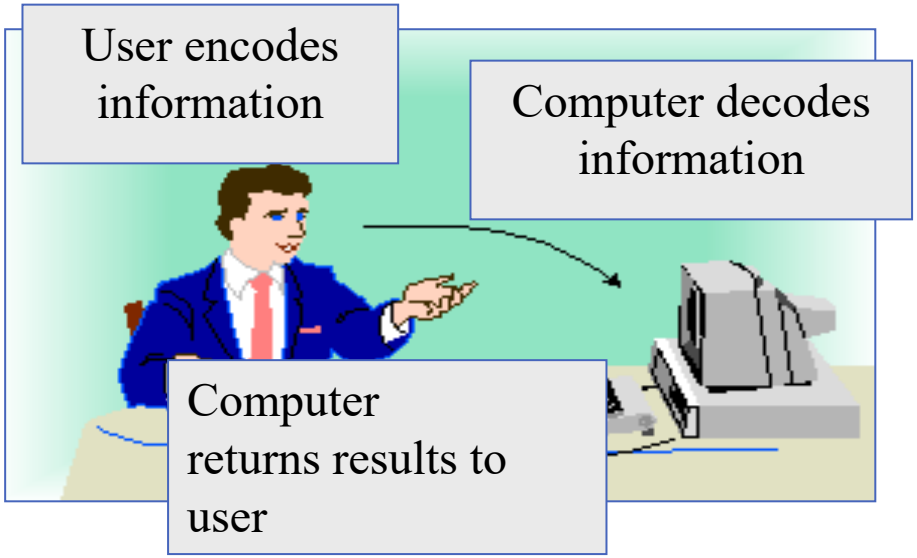
Communication cycle

- One complete unit of communication
  - An idea to be sent
  - An encoder
  - A sender
  - A medium
  - A receiver
  - A decoder
  - A response



程序设计语言

- **Substituting a computer for one of the people in the communication process**
  - Process is basically the same
    - Response may be symbols on the monitor
- **Programming languages bridge the gap between human thought processes and computer binary circuit**



程序设计语言

- Programming Language: A formal computer language or constructed language designed to **communicate instructions to a machine**, particularly a computer[1].

Machine readable

Machine language

Human readable

Compile

Assembly language, FORTRAN, Pascal, C / C++, Java, PHP, Python, Ruby, Perl, MATLAB, Wolfram, Scala, GO, Swift, etc

A picture of books that teaching programming languages and a word cloud of programming language names. There are thousands of programming languages are invented in the history.

[1] https://en.wikipedia.org/wiki/Programming_language

设计一门语言应该考虑哪些问题?

是不是我跟计算机讲什么词，它都能听懂?	是不是我跟计算机说什么格式的数，它都能听懂?	是不是我跟计算机说任何运算符号，它都能听懂?	世界上的逻辑纷繁复杂，要设计多种句式才够用?
计算机语言应该有“关键字”	计算机语言应该定义有限种数据类型。	计算机语言应该定义有限种运算符。	三种句式：顺序、分支、循环

程序设计语言的成分不外四种

Bohm, Corrado; Giuseppe Jacopini. Flow Diagrams, Turing Machines and Languages with Only Two Formation Rules. Communications of the ACM. May 1966, **9** (5): 366–371. doi:10.1145/355592.365646.

程序设计- 数据结构与算法

Design of computer program including 2 levels:

- Data Structure & Algorithm

- **A data structure** is a particular way of organizing and sorting data in a computer so that it can be used efficiently

Related Books

Linear Data Structure

A data structure is said to be linear if its elements form a sequence, or in other words a linear list

- Array
- Stack
- Queue
- Linked List

Non- Linear Data Structure

A non-linear structure is mainly used to represent data containing a hierarchical relationship between elements.

- Tree
- Graph

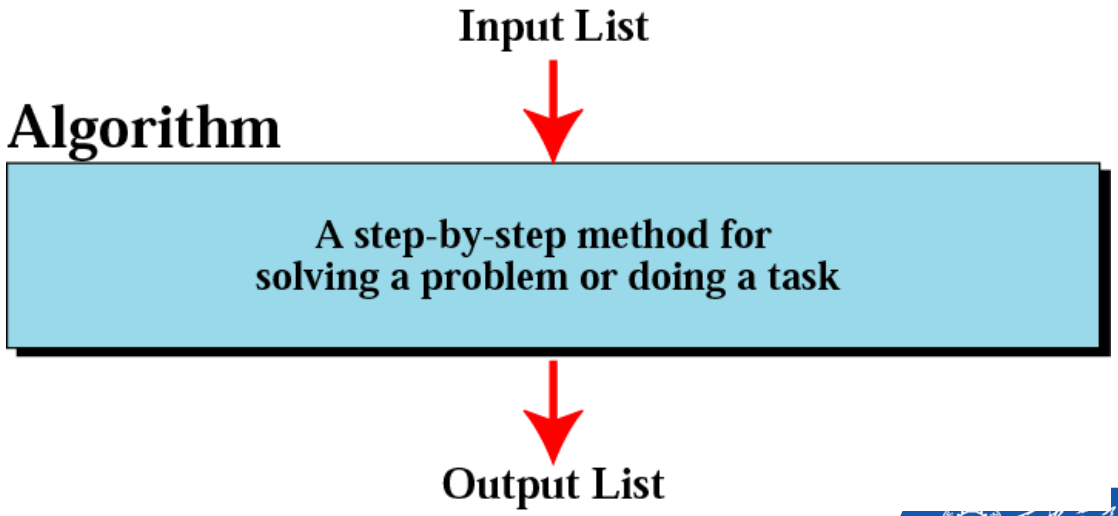
程序设计- 数据结构与算法

Design of computer program including 2 levels:

- Data Structure & Algorithm

Related Books

- It can be defined as a collection of unambiguous executable instructions, whose step by step execution leads to **a predefined goal**, within a finite number of steps.
- **Goal**: Desired input/output relationship.



1 PART

Design: an example

- **Example of Insert Sort**

Input: sequence $\langle a_1, a_2, \ldots, a_n \rangle$ of numbers.

Output: permutation $\langle a'_1, a'_2, \ldots, a'_n \rangle$ such that $a'_1 \le a'_2 \le \cdots \le a'_n$.

Example:

Input: 8 2 4 9 3 6

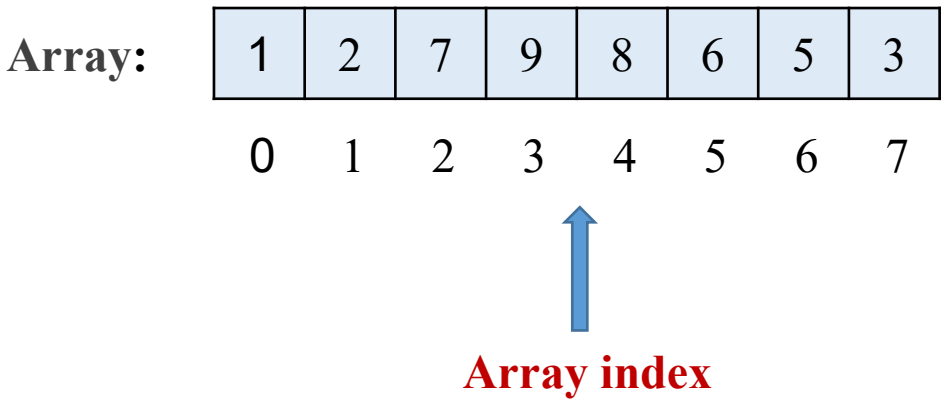
Output: 2 3 4 6 8 9

1 PART

Design: an example

Array

- An array is a data structure consisting of a collection of elements (values or variables), each identified by at least one array index or key
- An array is stored so that the position of each element can be computed from its index tuple by a mathematical formula.



1 PART

Design: an example

- **Example of Insert Sort**

1 PART

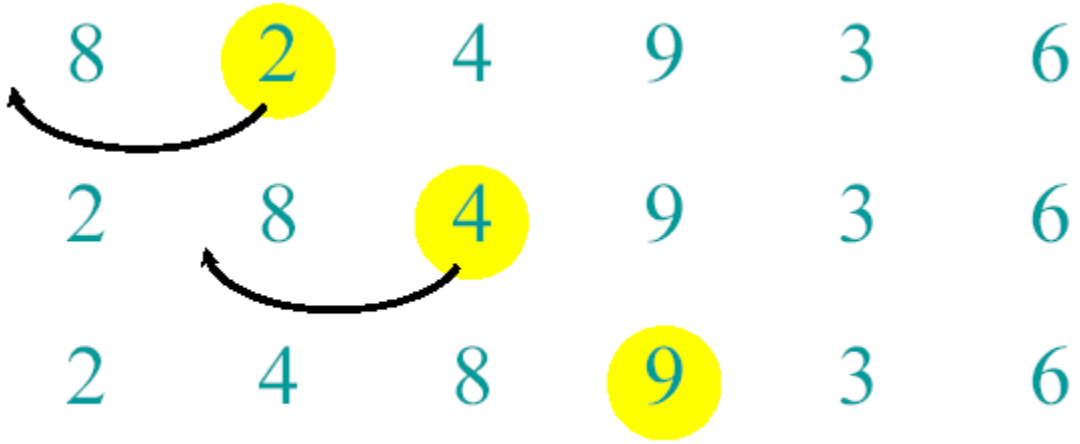
Design: an example

- **Example of Insert Sort**

1 PART

Design: an example

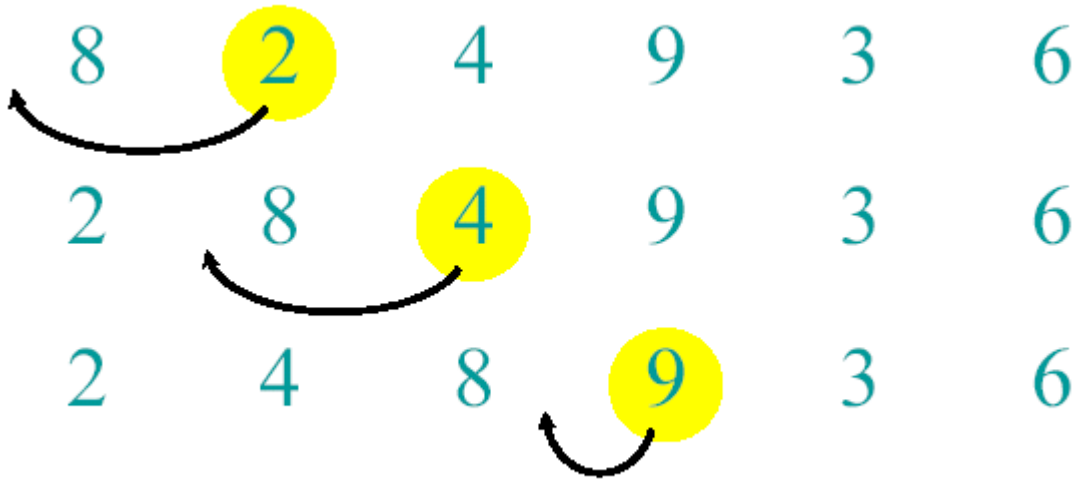
- **Example of Insert Sort**



1 PART

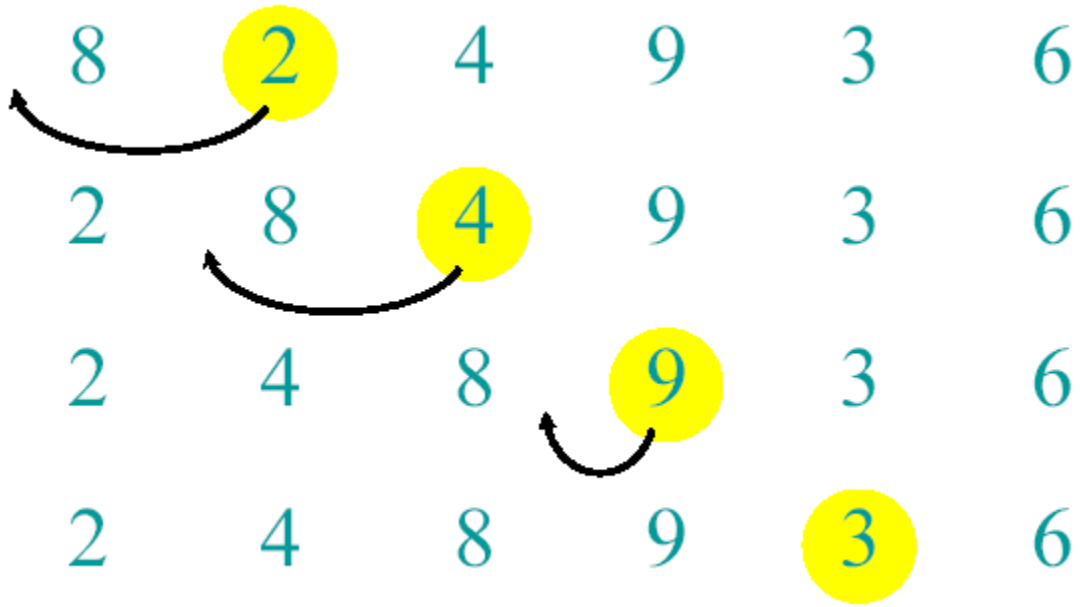
Design: an example

- **Example of Insert Sort**



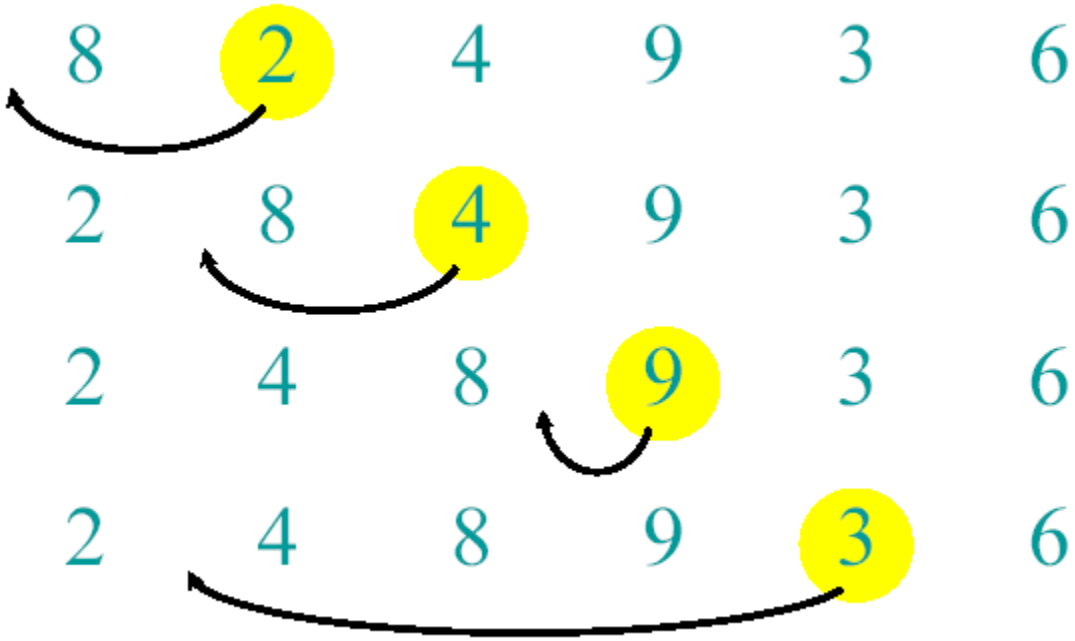
Design: an example

- **Example of Insert Sort**



Design: an example

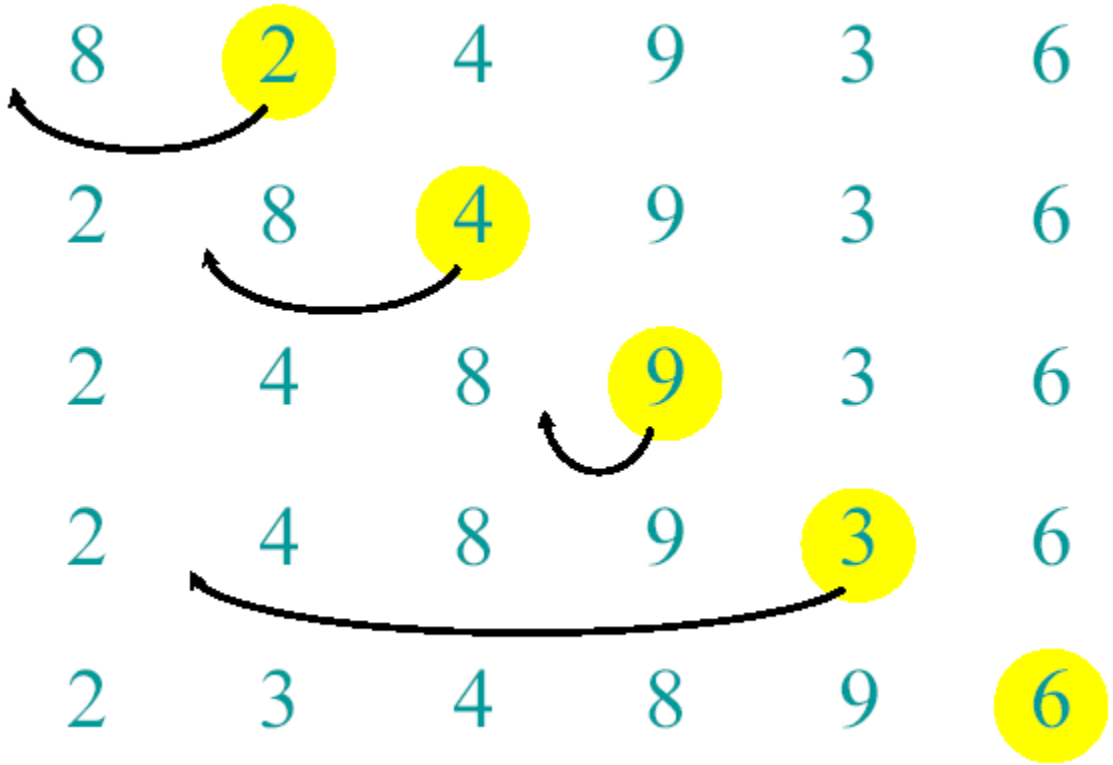
- **Example of Insert Sort**



1 PART

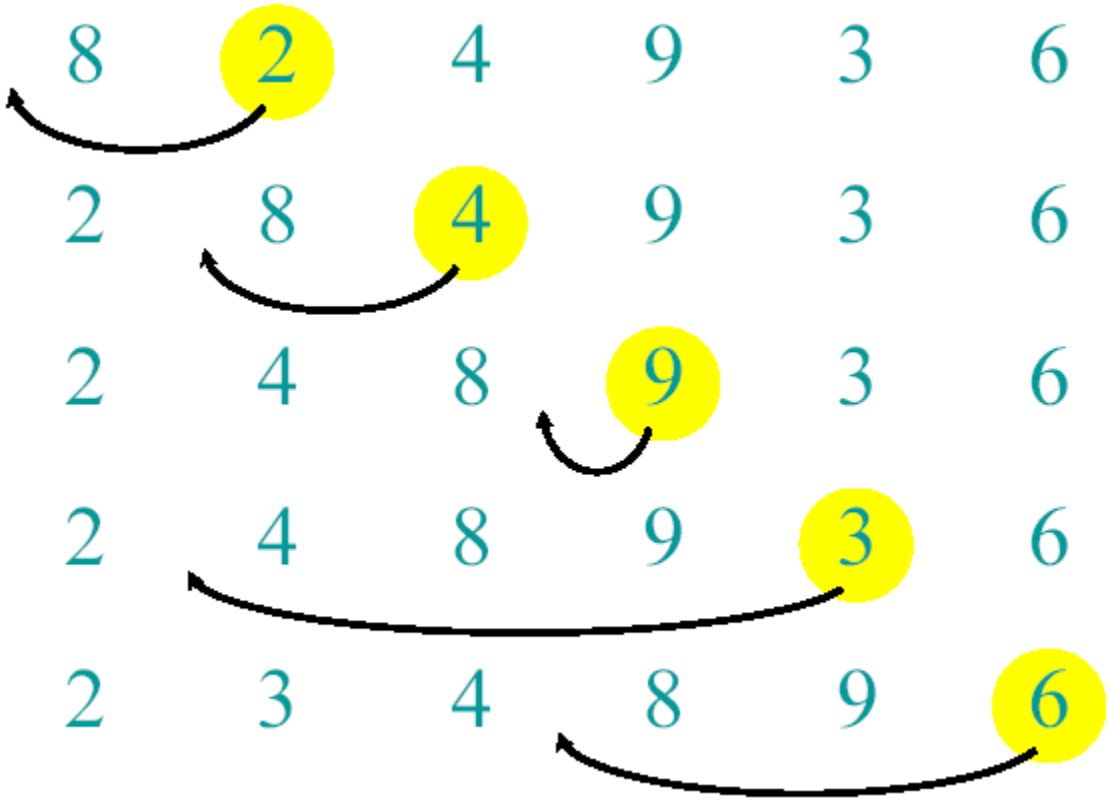
Design: an example

- **Example of Insert Sort**



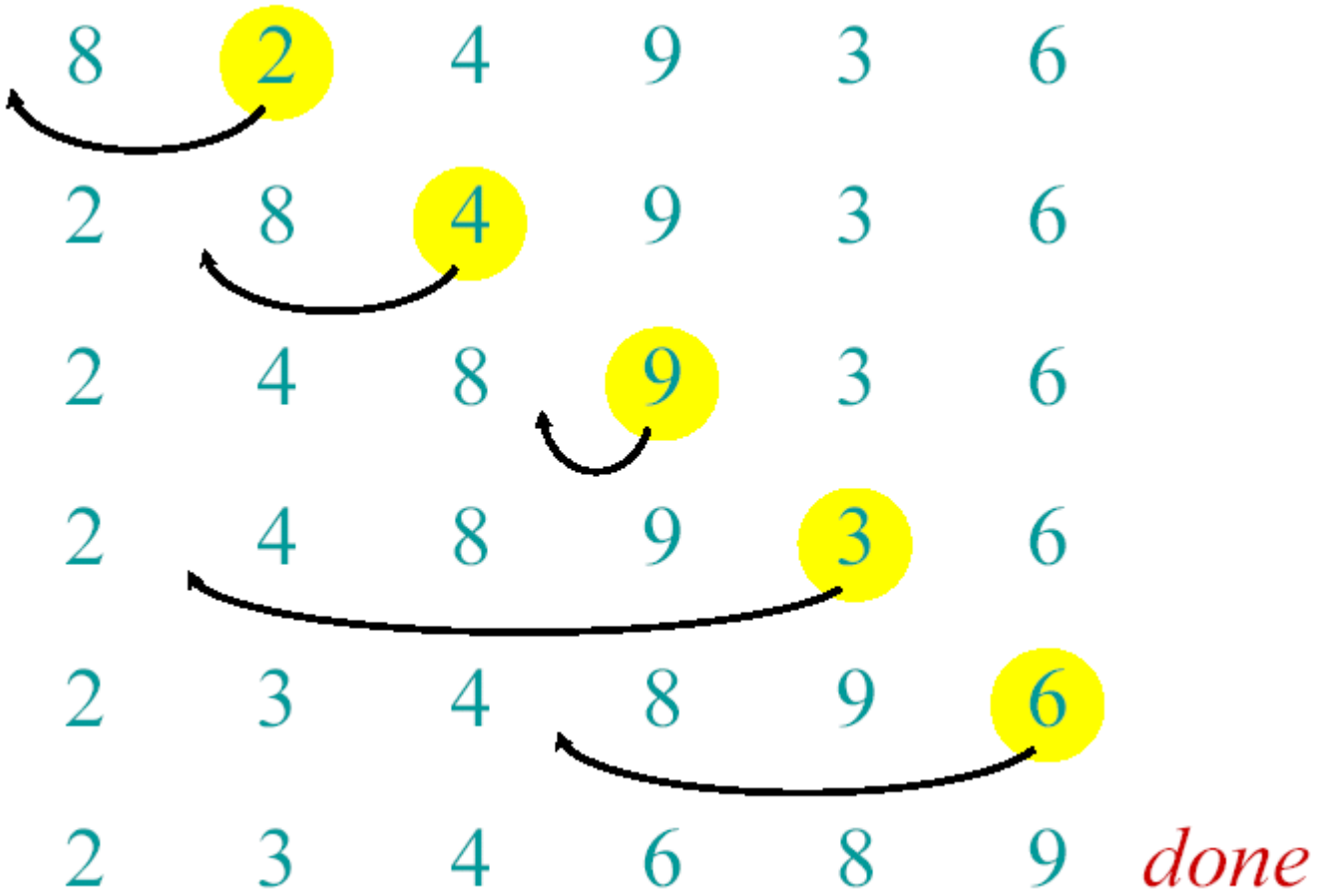
Design: an example

- **Example of Insert Sort**



Design: an example

- **Example of Insert Sort**



1 PART

Design: an example

Example of Insert Sort

A *pseudocode* for insertion sort (INSERTION SORT).

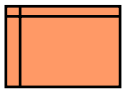
```
INSERTION-SORT(A)
1   for j ← 2 to length [A]
2       do key ← A[ j]
3       ∇ Insert A[j] into the sortted sequence A[1,..., j-1].
4        i ←  j – 1
5        while i > 0 and A[i] > key
6              do A[i+1] ← A[i]
7                    i ← i – 1
8       A[i +1] ←  key
```

6 5 3 1 8 7 2 4

程序设计- 架构设计

- Architecture
- Data Structure & Algorithm

Related Books

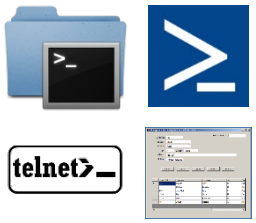


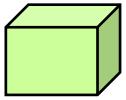
Client (Browser/ Mobile)

Browser

Mobile

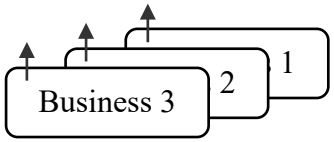
Management





Server

Dynamic



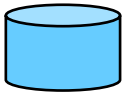
Service-Oriented Architecture

Static

Content Delivery Network

Binary

Text

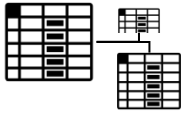


Data

Persistent

Cached

Structured

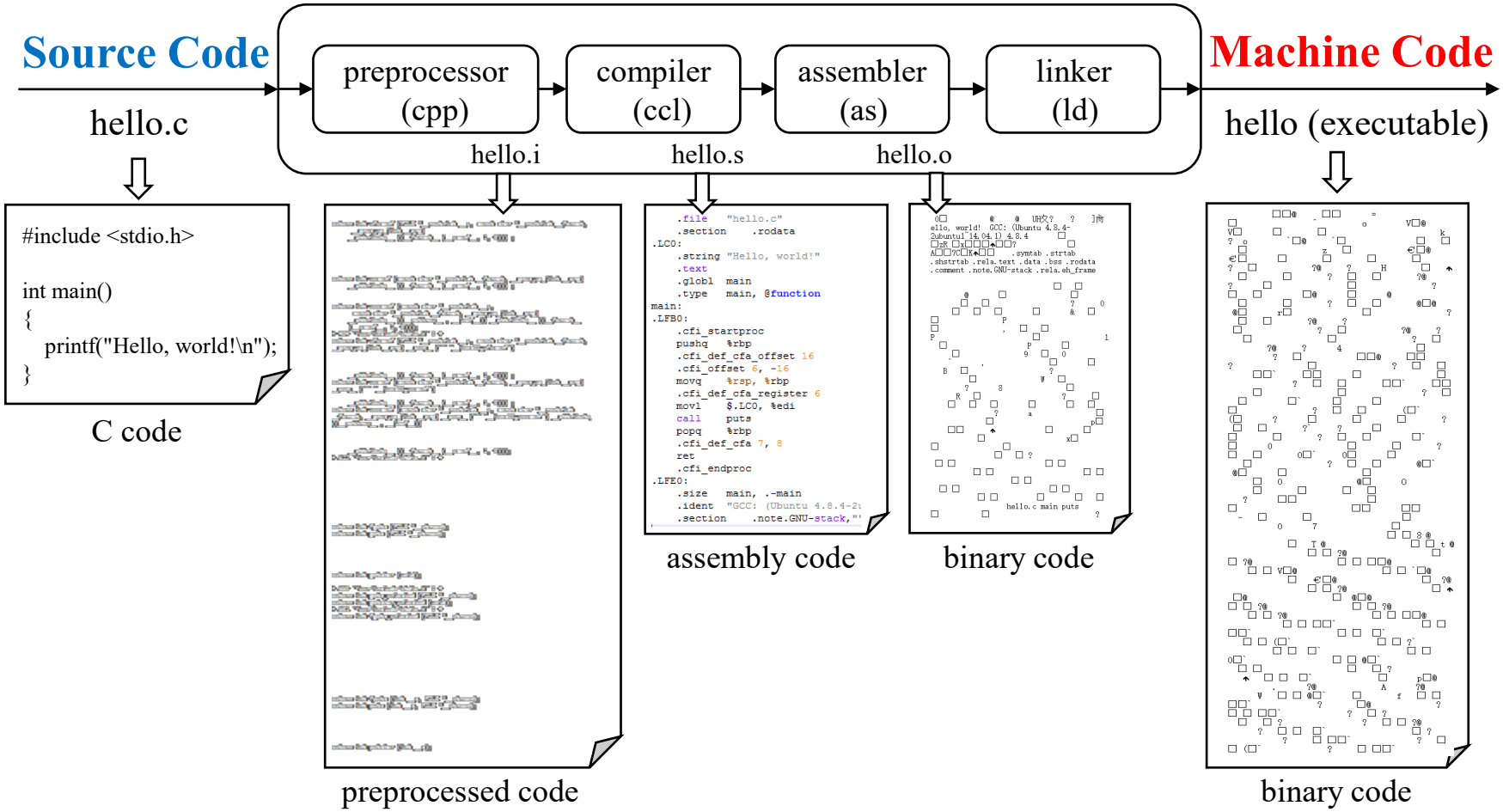


Non-Structured

Semi-Structured

编译

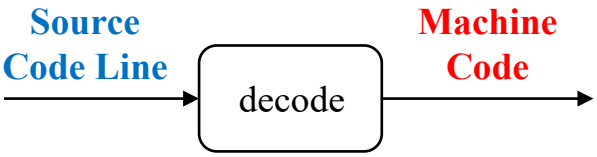
Translate **source code** from a human-readable language into either **machine code** or **object code**.



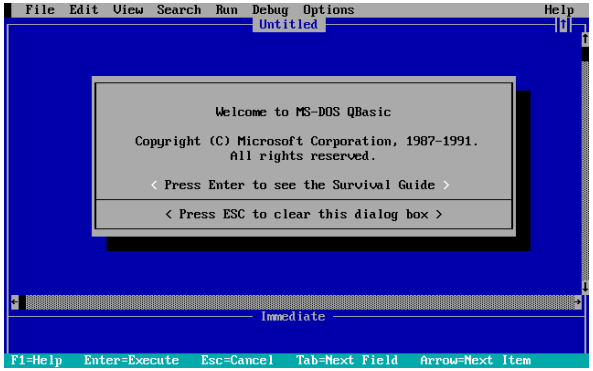
编译- 解释

Execute each line in source code from a programming language immediately, without an intermediate file.

First method

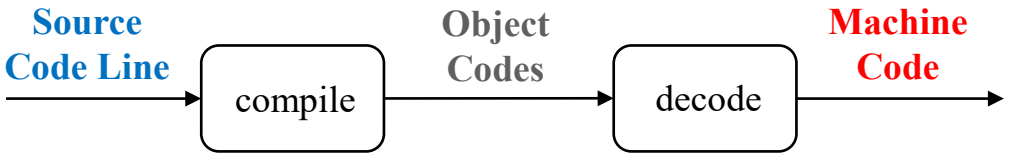


BASIC, APL

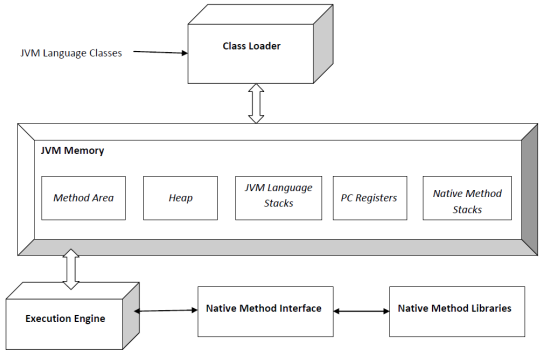


Microsoft QBasic Interface

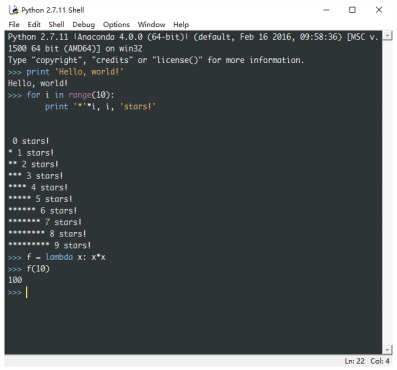
Second method



Java, Python, Ruby, Perl, JavaScript, Go, etc.



Java Virtual Machine Structure



Python IDLE Interpreter

主要内容

- 计算机程序：设计与编译
- 程序如何一步步执行：冯诺依曼计算机
- 操作系统如何管理程序

冯诺伊曼计算机

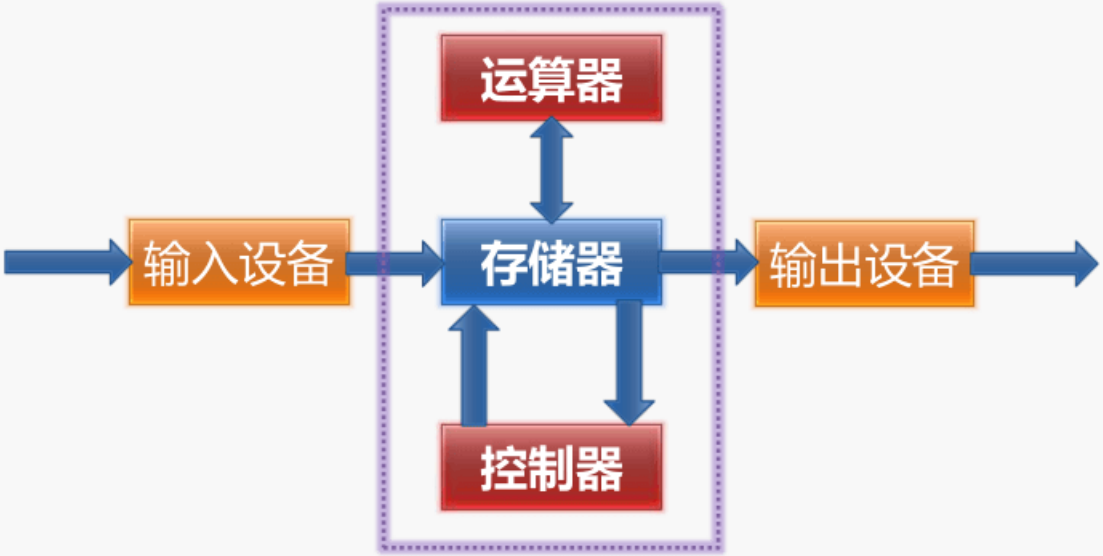
冯·诺依曼结构的要点

① 计算机应由运算器、控制器、存储器、输入设备和输出设备共5个部分组成

② 数据和程序均以二进制代码形式不加区别地存放在存储器中，存放位置由存储器的地址指定

③ 计算机在工作时能够自动地从存储器中取出指令加以执行

④ ……



冯诺伊曼计算机

主存的组织形式

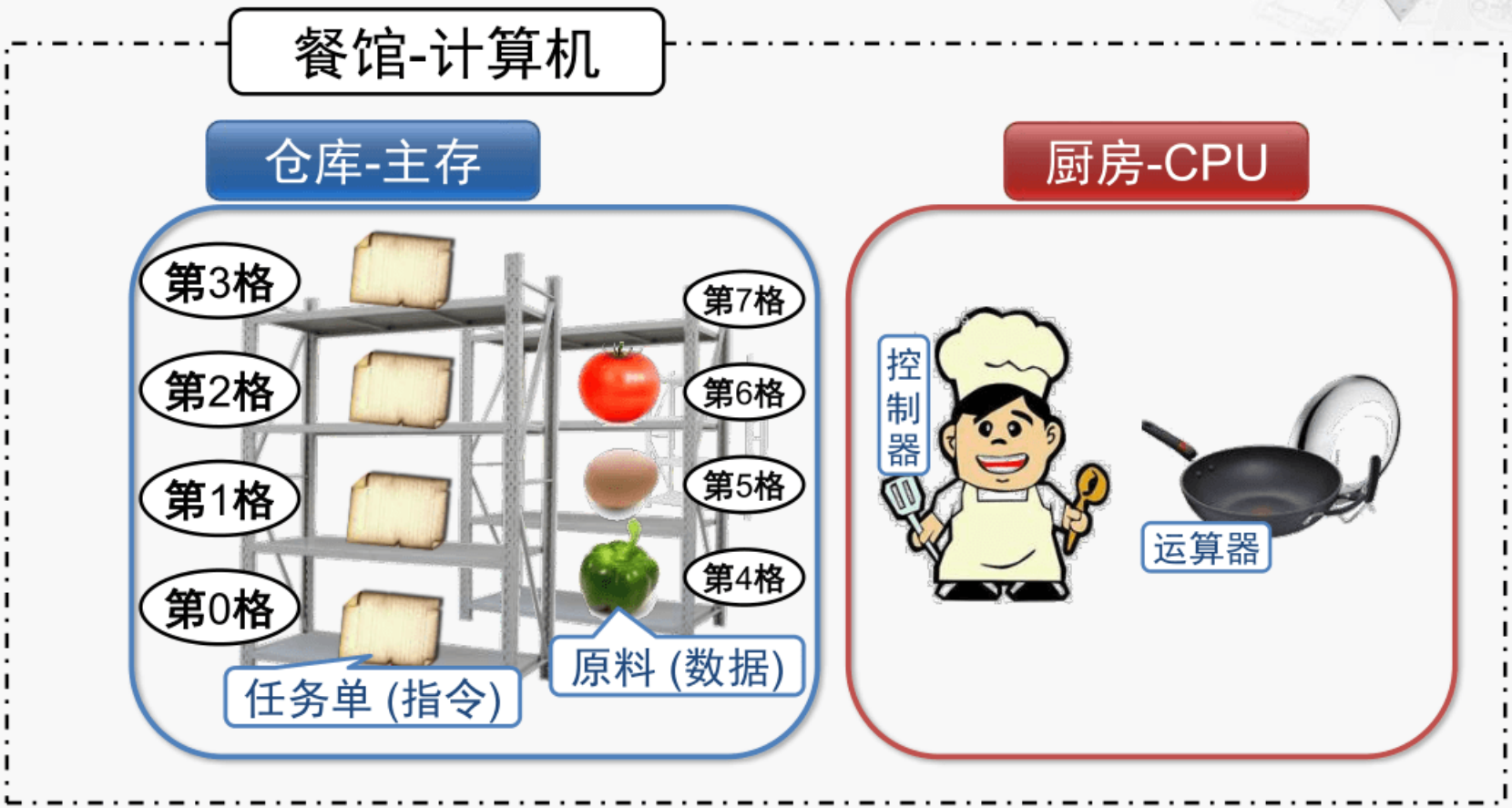
地址：每个存储单元对应的序号

内容：存储单元中存放的信息

地址（二进制）	内容（二进制）
…	……
0011	00001100
0010	00100010
0001	00000000
0000	01101101

冯诺伊曼计算机

冯·诺依曼计算机的类比



冯诺伊曼计算机

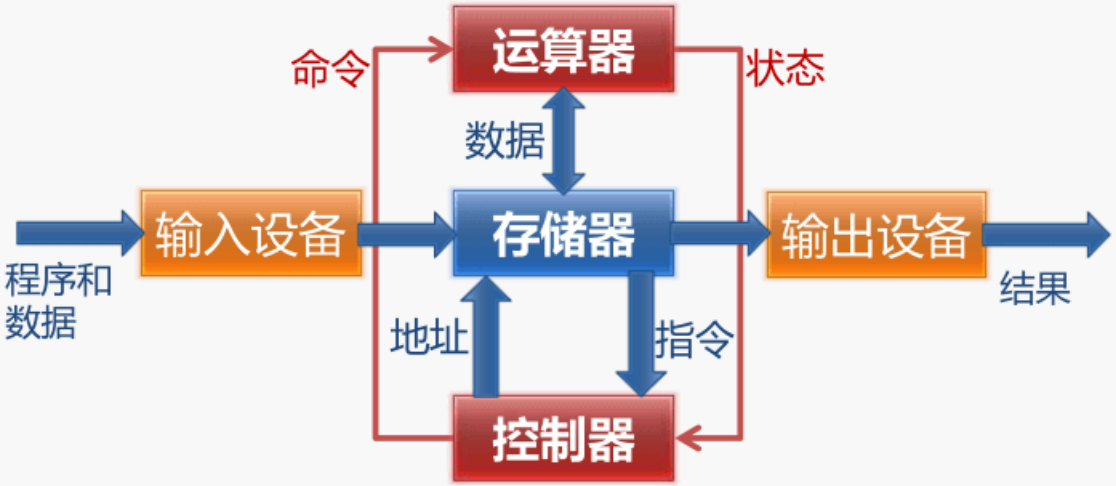
冯·诺依曼结构的要点

① 计算机应由运算器、控制器、存储器、输入设备和输出设备共5个部分组成

② 数据和程序均以二进制代码形式不加区别地存放在存储器中，存放位置由存储器的地址指定

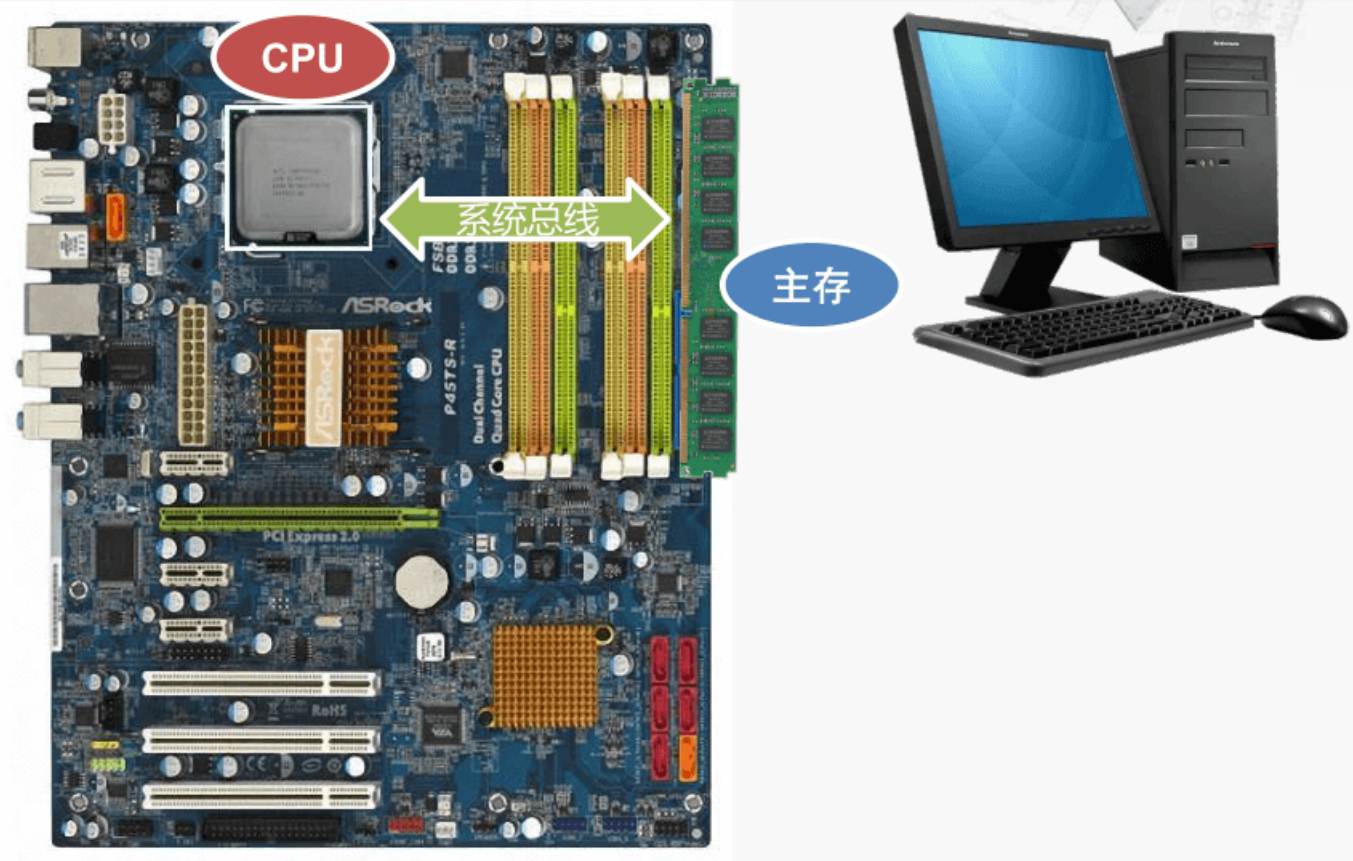
③ 计算机在工作时能够自动地从存储器中取出指令加以执行

④ ……



冯诺伊曼计算机

现代个人计算机中的CPU和主存



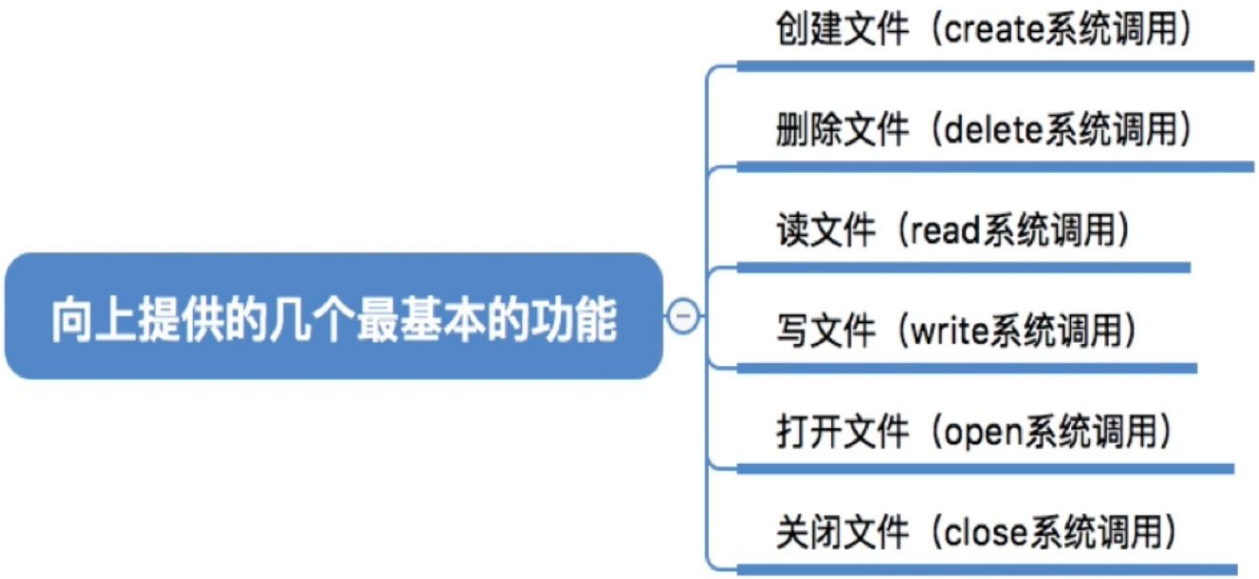
主要内容

- 计算机程序：设计与编译
- 程序如何一步步执行：冯诺依曼计算机
- **操作系统如何管理程序**

示例：磁盘管理

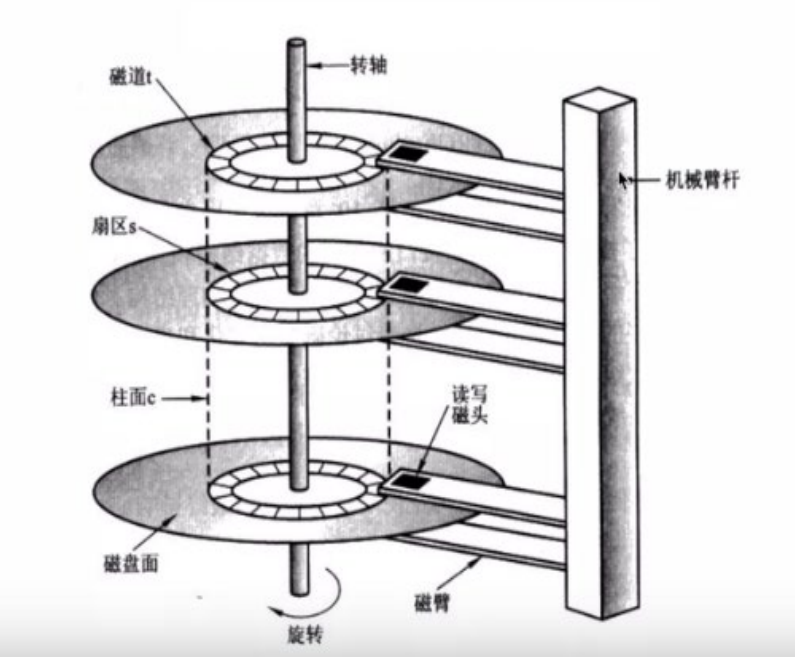
我们平时访问磁盘是如何访问的？

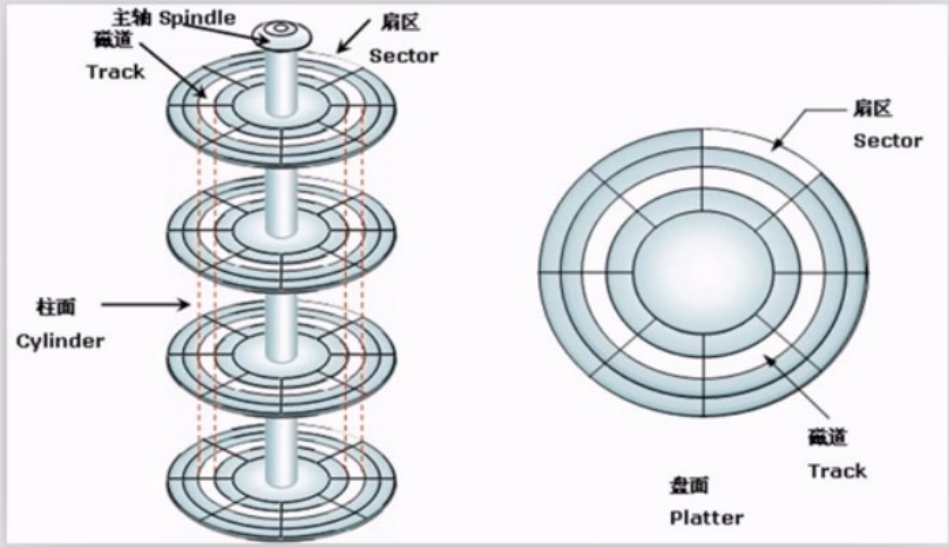
- 文件是最小单元



示例：磁盘管理

（从计算机角度）是如何访问的？





- 按扇区访问：每个扇区，存储一些数据，如512字节。磁头可以按编号寻找对应扇区

示例：磁盘管理

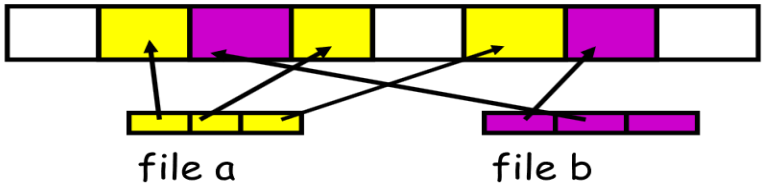
用户使用磁盘如此方便，真实的磁盘使用如此困难，你知道这背后发生了什么吗？

- 操作系统文件管理模块（文件系统）：
  - 文件系统将磁盘空间进行规划和编号处理，这样通过文件名就可以找到具体的数据，而不用关心数据到底是怎么存储的。
  - 两大功能：
    - 以文件名找到对应的磁盘块
    - 持续跟踪空闲空间

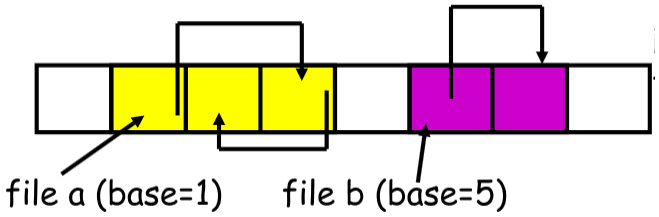
示例：磁盘管理

如何通过文件名找到对应的磁盘块呢？

- 直接索引：采用最最直接的指针方式来索引数据块。



- 间接索引：单个指针指向文件的第一个数据块，每个数据块的末尾存储下一个块地址。



- 他们的优缺点是啥？分层索引又是啥？

示例：磁盘管理

存放索引的数据叫什么？

- 元数据（Data about Data）

这部分元数据需要存储在磁盘中吗？

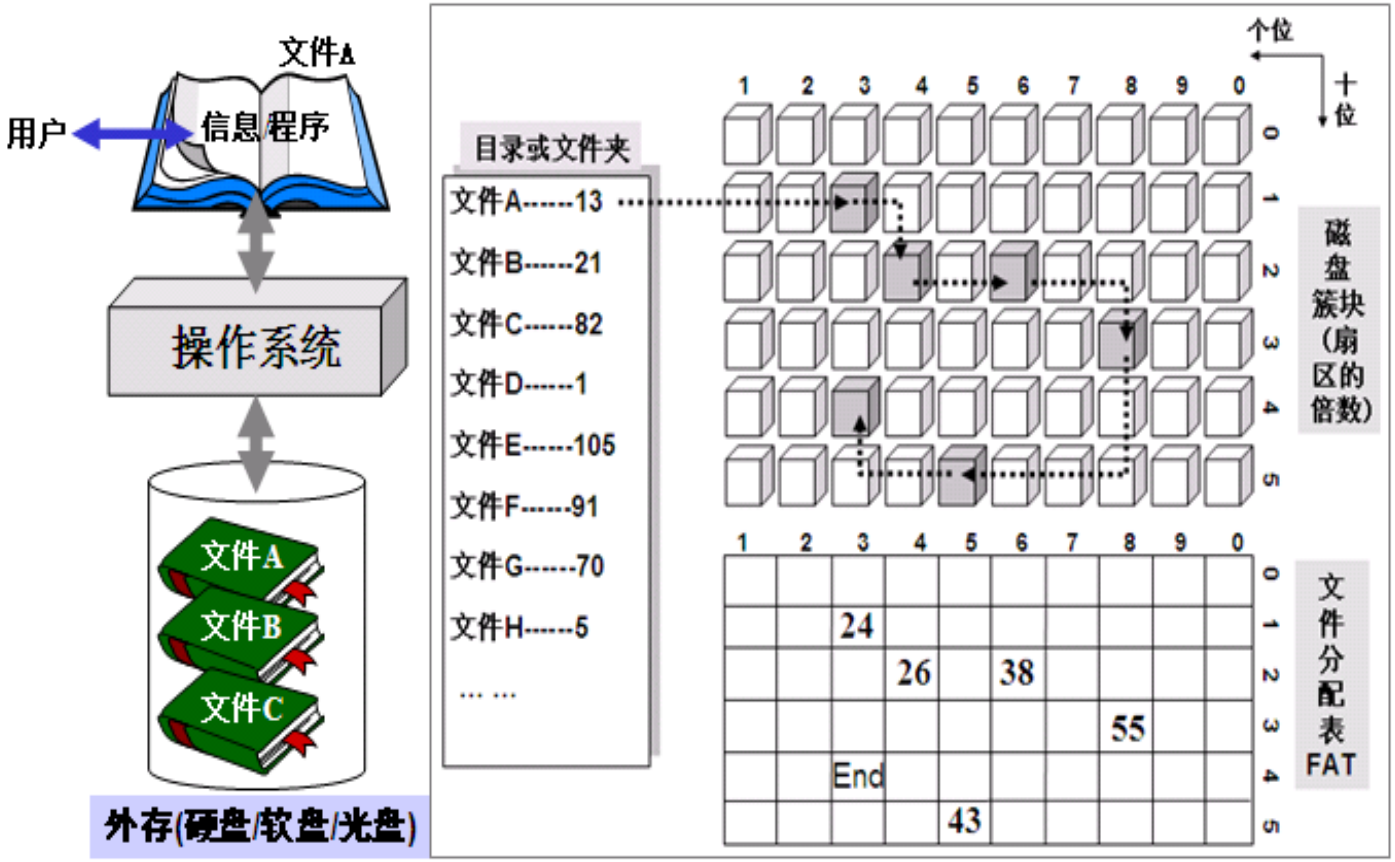
- 需要，要不重启或计算机掉电后，我们就找不到文件对应的磁盘位置了

这部分元数据又该如何索引呢？

- 一般放在规定好的位置（例如磁盘最前面）

示例：磁盘管理

FAT文件系统示例



示例：内存-磁盘分级体系

程序在存储器中，理想存储器应该是什么样子？

- 无限空间
- 零延迟
- 便宜
- 高带宽

示例：内存-磁盘分级体系

现实是什么？

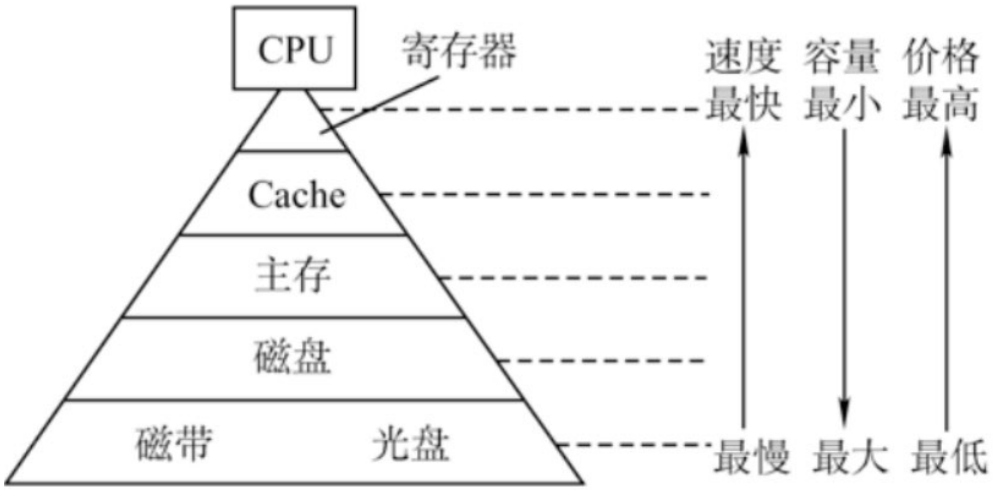
- 速度越高，位价就越高
- 容量越大，速度必越低

存储器	硬件介质	单位成本（美元/MB）	随机访问延时
L1 Cache	SRAM	7	1ns
L2 Cache	SRAM	7	4ns
Memory	DRAM	0.015	100ns
Disk	SSD（NAND）	0.0004	150μs
Disk	HDD	0.00004	10ms

示例：内存-磁盘分级体系

如何同时满足多个理想目标的需求呢？？

- 宿舍衣柜不够大，你们怎么放衣服？



分级存储体系：

相对不容易被访问的内容，放入便宜、容量大、速度慢的存储

相对容易被访问的内容，放在快速、昂贵的存储

同时解决了内存掉电后失效的问题

示例：内存-磁盘分级体系

Memory	DRAM	0.015	100ns
Disk	SSD(NAND)	0.0004	150μs

应用局部性原理的效率计算

- 假设我们现在有100万个字长的数据需要存储
  - 我们将所有数据都放入Disk。现在CPU要访问100万次数据，那么这种情况需用时多少，成本多少？
  - 我们假设CPU的90%访问都集中访问数据中的10%。那么我们设计这样一个结构，将这10%的数据放在Memory，剩下的90%数据继续存放在Disk中。现在CPU同样访问100万次数据，这种情况用时多少，成本多少？
  - 答案：价格贵了4.65倍，速度快了94倍

示例：内存-磁盘分级体系

为什么多级存储体系可以工作呢？

- 程序局部性原理：
  - 程序在执行时呈现出局部性规律，即在一段时间内，整个程序的执行仅限于程序中的某一部分。
  - 时间局部性：
    - 被引用过一次的存储器位置在未来会被多次引用（例如：循环中的变量；被反复调用的函数）
  - 空间局部性：
    - 如果一个存储器的位置被引用，那么将来他附近的位置也会被引用（例如：顺序执行的程序）。

示例：内存-磁盘分级体系

你能举出更多生活中的局部性原理的例子吗？？

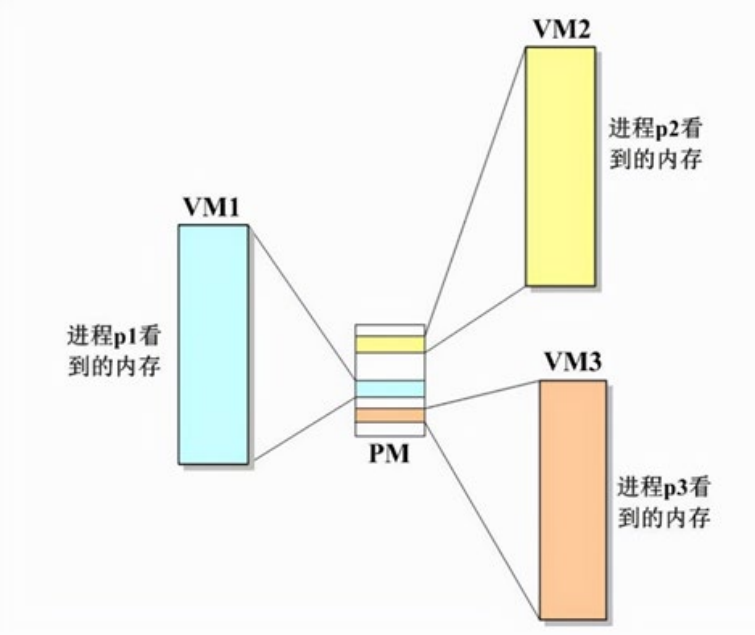
思维启示：

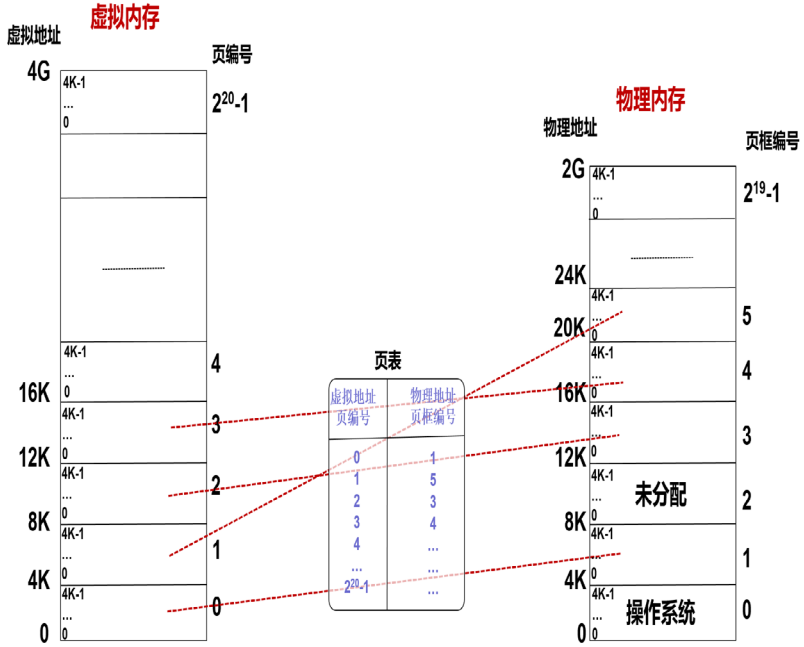
- 1. 计算机是一项系统工程，往往需要各种平衡和折衷（比如价格，性能）。
- 2. 多种资源组合利用往往有更好的效果。

示例：内存管理

内存管理：什么数据会被调入内存？

- 当作业或进程创建后系统会为他们分配内存空间，当结束后内存空间也会被回收。保证个个作业在自己的内存空间内运行，互不干扰。

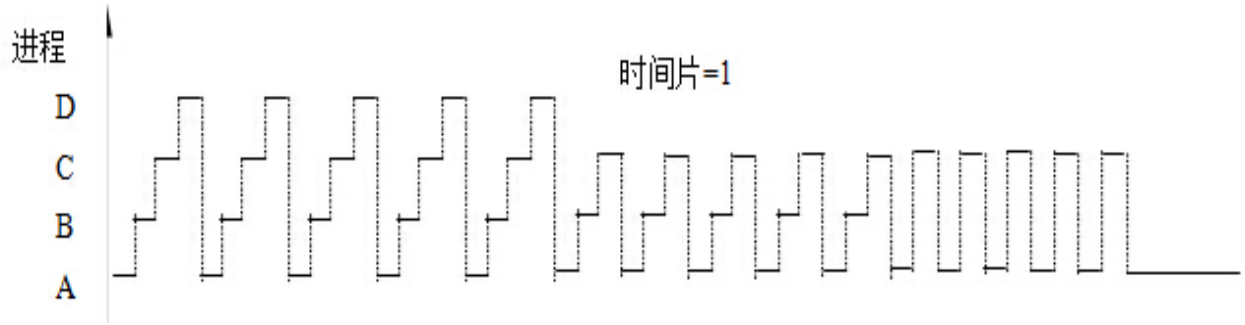




示例：进程调度

进程管理：进程的创建、调度与销毁

- 程序一旦被装入内存执行，就被称之为进程。
- 进程中不只有代码，还有程序计数器的值（代表程序执行到哪里），堆栈段（包括临时数据）和数据段（包括全局变量）等。
- 操作系统根据调度算法，让不同的进程分时、依次执行。



示例：中断管理

中断管理：完成了点击怎么响应

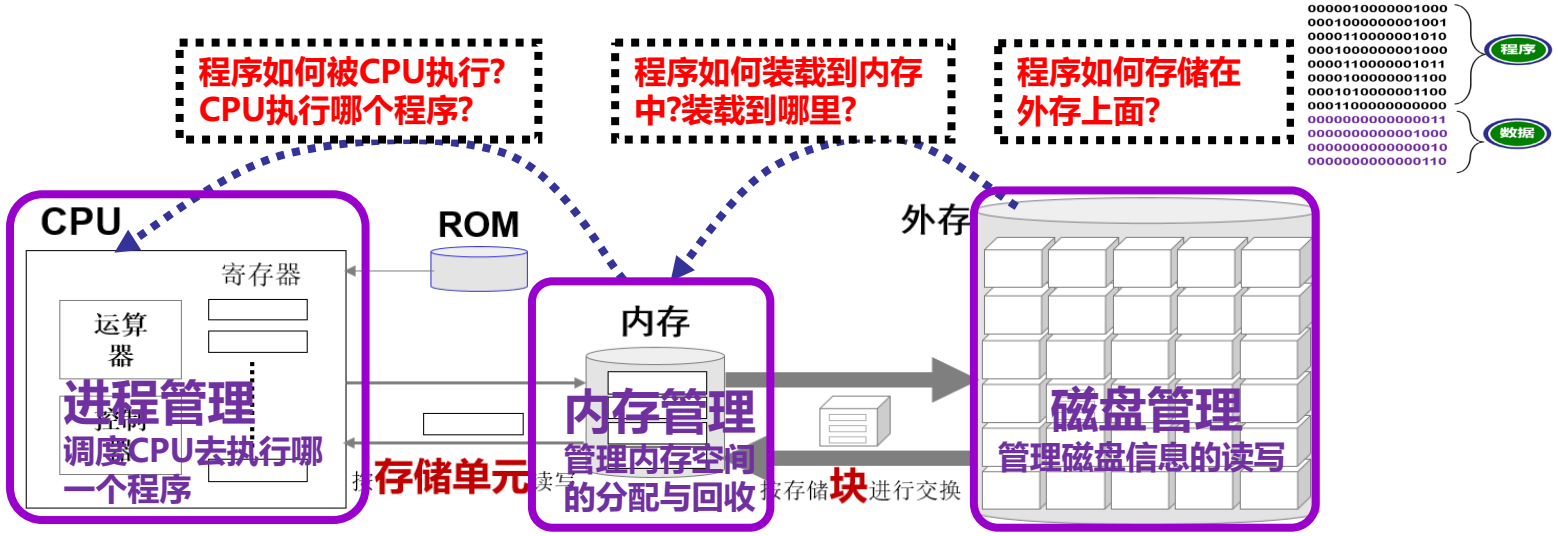
- 当设备收到输入信号时，便通过控制线向CPU发送一个中断信号，当CPU收到中断信号后，再通过中断响应程序进行处理。
- 优点：在I/O设备输入每个数据的过程中，由于无需CPU干预，因而可使CPU与I/O设备并行工作，仅当完成一个数据输入时，才需CPU花费极短的时间去做一些中断处理。

操作系统如何管理程序

根本问题：程序是如何被执行的？

操作系统是一组“管理各种资源以便执行应用程序”的程序

(1)分工。首先独立管理每个部件：磁盘管理、内存管理、进程管理



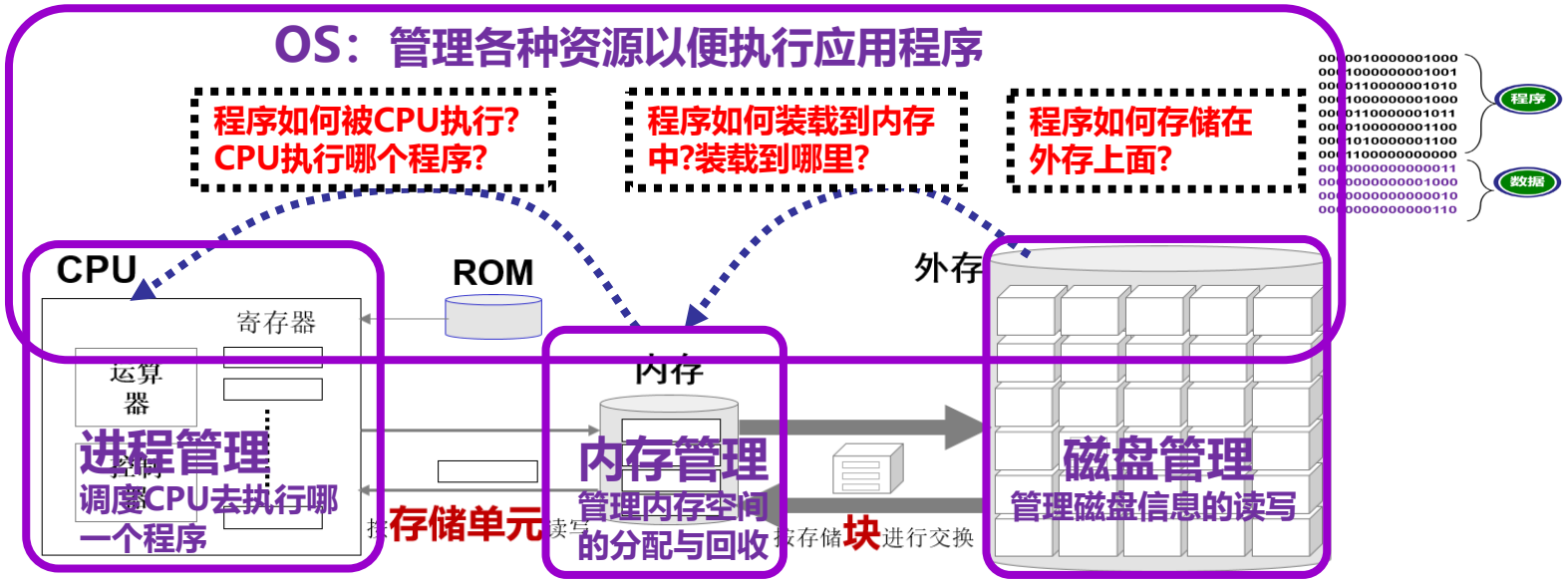
操作系统如何管理程序

根本问题：程序是如何被执行的？

操作系统是一组“管理各种资源以便执行应用程序”的程序

(1)分工。首先独立管理每个部件：磁盘管理、内存管理、进程管理等

(2)合作。各部件合作完成“让CPU执行存储在外存上的程序”



作业（思考、调研）

1. 阅读1966年论文，谈谈自己的理解
2. 从程序员写程序，到用户拿鼠标双击exe文件打开这个程序，这期间都发生了什么？你能运用本堂课知识，详细描述一下吗？
3. 操作系统是程序吗？操作系统管理如何将程序载入内存，那OS自己又是如何被放入计算机内存中的呢？