

People who are really serious about software
should make their own hardware.

--- Alan Kay

C 程序设计
C Programming

1

计算机基础

附加课程



廈門大學
XIAMEN UNIVERSITY



信息学院
(国家示范性软件学院)
School of Informatics

黃 煒
博士, 副教授
Dr. Wei Huang

知识框架

- 相关名词解释
- 程序员的工作
- 程序设计的意义
- 如何学习C语言
- 计算机基础知识（部分）

内容纲要

1	相关名词解释
2	程序员的工作
3	程序设计的意义
4	如何学习C语言
5	计算机基础知识（部分）

术语：硬件和软件

- 正常工作的计算机，包括两部分。
 - 硬件：物理设备，是看得见、摸得着的实体。
 - 内部设备：CPU、内存、主板、硬盘……
 - 外部设备：键盘、鼠标、音箱、……
 - 软件：逻辑产品，是一套指令。
 - 系统软件：Windows、Android、iOS等
 - 应用软件：Office（Word）、QQ、微信等
- 软件工程：用工程化方法构建和维护软件

术语：软件和团队合作

- 软件：一系列按照特定顺序组织的数据和指令的集合
 - 这里的指令是站在计算机的角度而言的
 - 大型软件需要多个团队共同分担协作完成
- 团队合作：群体中的个体为实现某一目标而相互协作
 - 每个团队由多个人组成，各人书写一个功能模块
 - 队长将一个功能模块划分成多个程序或片段，由队员完成
 - 队员们书写程序，相互使用（调用），形成有机整体

术语：程序和程序员

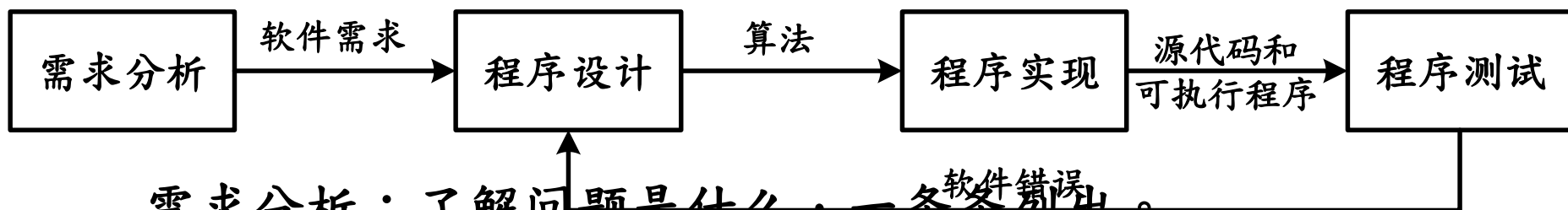
- 程序：为进行某项活动或过程所规定的途径
 - 办事程序：递交申请材料，资格审查，复核，办结
 - 计算机程序：获取两个数字，将其相加，生成两个数的和
- 程序员：书写计算机程序的人员
 - 程序员将办事流程用某种语言书写出来，生成指令
 - 有的程序员写出低质量程序，常需要队友返工，难以合作
 - 有的程序员责任心不强，常拖延工期，难以沟通
 - 为了写出高质量程序，程序员需要熟练掌握编程语言

内容纲要

1	相关名词解释
2	程序员的工作
3	程序设计的意义
4	如何学习C语言
5	计算机基础知识（部分）

程序员的工作流程

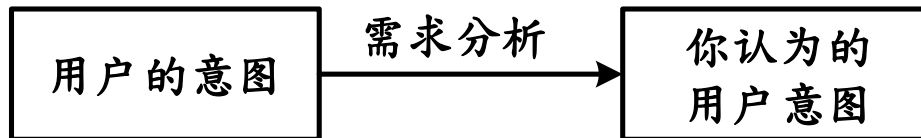
• 程序员的工作流程



- 需求分析：了解问题是什么，一条条列出。
- 程序设计：根据需求，设计解决问题的方案。
- 程序实现：书写具体的程序语言并得到可执行程序。
- 程序测试：根据问题验算程序是否得到预期的结果。
- 修改程序：如果是设计的问题，修改设计，再修改实现

需求分析

- 需求分析：为了了解用户的意图的活动



— 沟通途径：和用户交流、审题

- 包括：用户说出来的，说错的，没说出来的.....

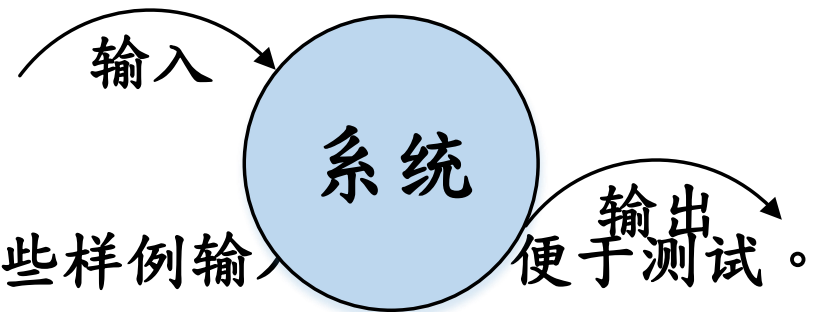
— 需要的能力：语文水平、心理学

— 重要性

- 你认为的意图很可能不是用户的真实意图
- 如果一开始理解错了，后面的设计和实现便都是错的

需求分析：输入输出

- 需求分析应明确系统的输入和输出
 - 从数据传递和加工的角度，一个系统将输入进行处理，最终得到输出。
 - 从而推导出输入和输出之间的联系（设计和实现的内容）
- 输入和输出
 - 输入：外界给我们什么
 - 输出：我们给外界什么
 - 必要时，设计或向用户寻求一些样例输入，便于测试。



需求分析：示例

- 先读题目
- 确定输入输出
 - 物理意义，单位
- 通过样例输入输出核对理解无误
 - 将来用于初步测试

题目

一辆汽车以一定的初始速度做匀加速直线运动，对给定的初始速度、加速度和时间，计算位移。

输入

初始速度、加速度和时间

输出

位移

样例输入

1 1 1

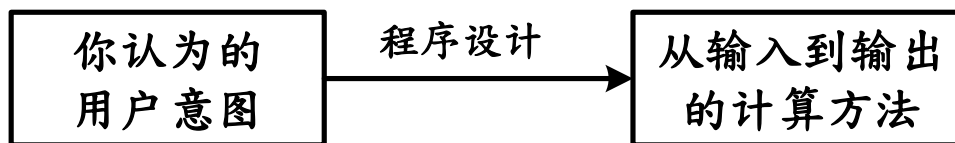
样例输出

1.5

程序设计

- 计算机程序设计

- 建立和设计可执行计算机程序以完成特定计算任务的过程



- 输入：外界提供给计算机程序的信息

- 一个程序有零个、一个或多个输入

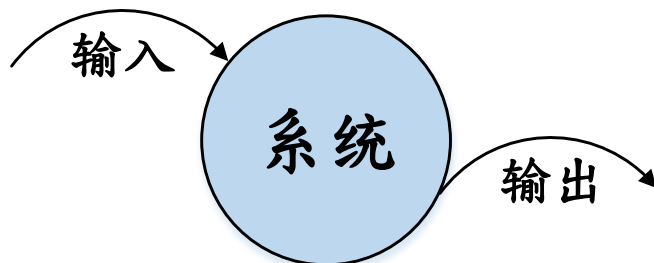
- 输出：计算机程序向外界提供的信息

- 一个程序有一个或多个输出

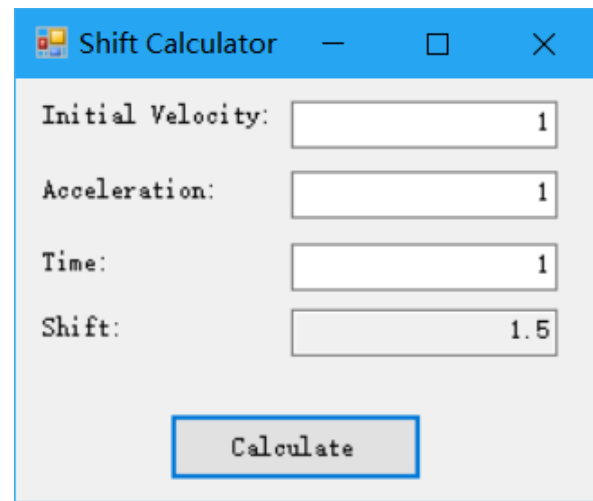
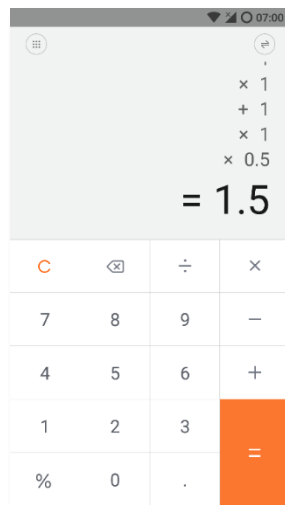
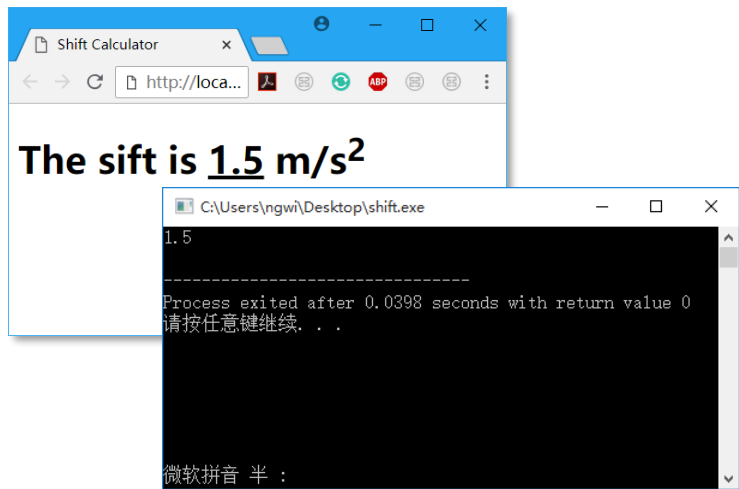
- 处理过程：接受输入，经过计算，将结果输出

程序设计：输入输出的含义和表示

- 本课程专注于根据输入运算得到预期输出的规律。



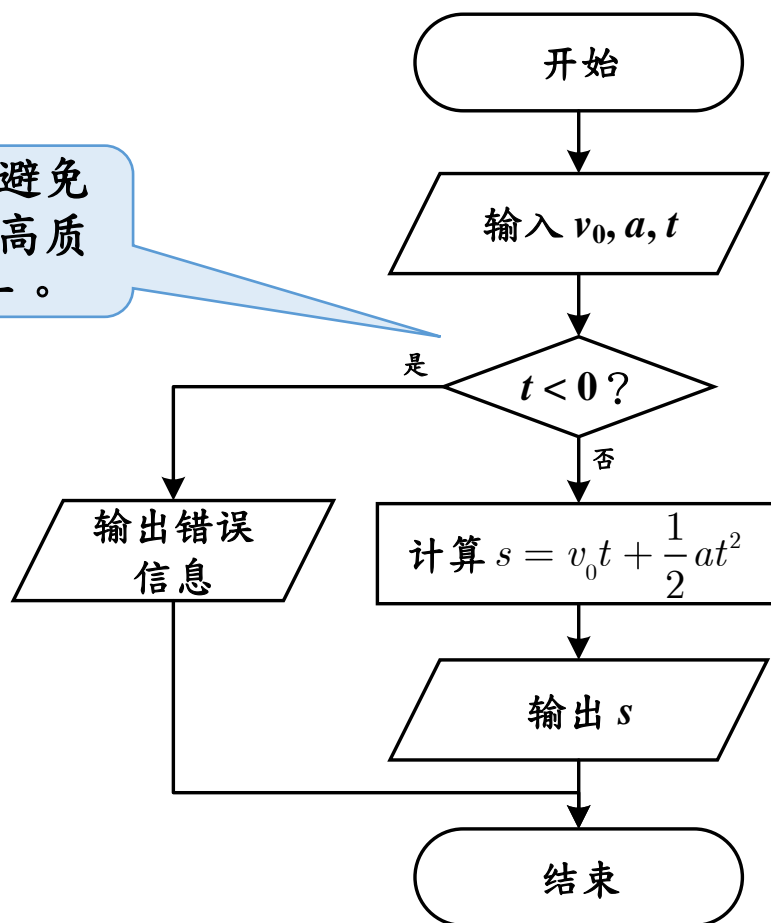
- 至于输入输出的表现形式，不是本课所关心的。



程序设计

- 总结出输入和输出关系的规律

能够在输入错误时避免给出错误答案，是高质量程序的要求之一。



程序实现

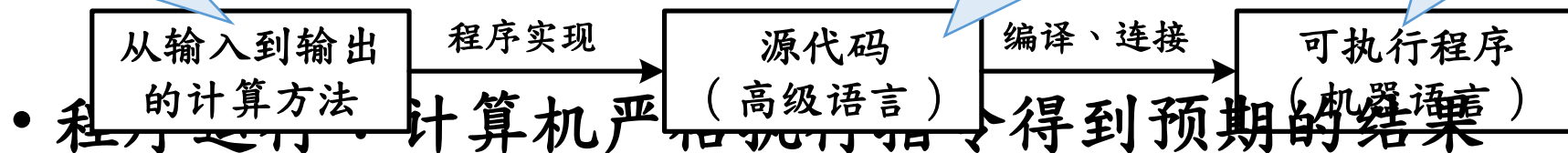
- 程序实现：将上述规律制作成计算机指令

- 计算机指令由计算机能看懂的语言表示

程序设计的
结果

专业人员可以
阅读和书写

机器看得懂，
人写不出来



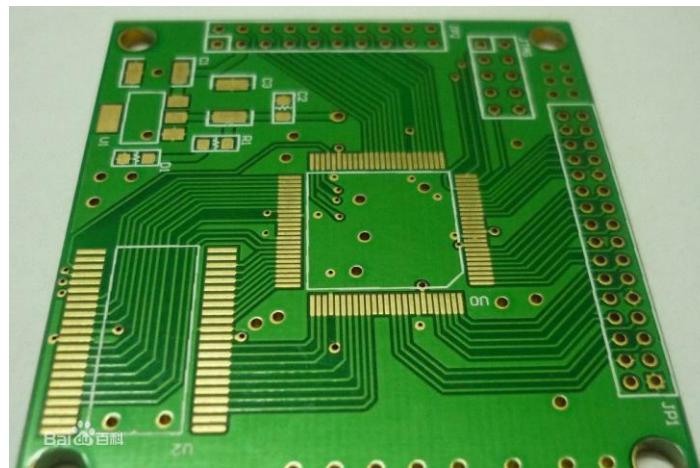
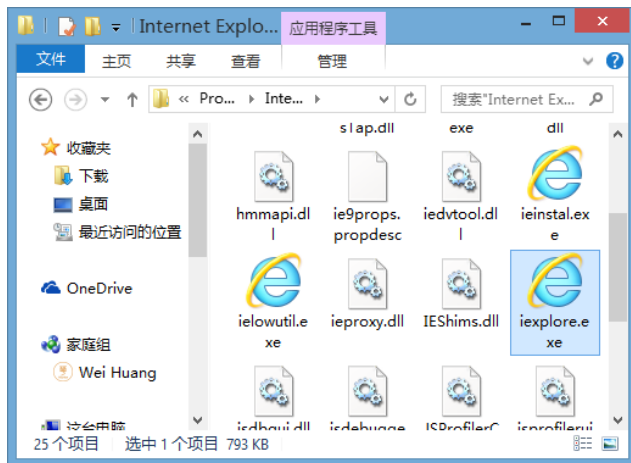
- 程序一次实现，可以多次运行

- 平时：计算机指令一般存储在磁盘文件（可执行程序）中

- 运行时：将磁盘加载入内存，按时序在CPU调用执行

程序实现：机器语言

- 机器语言的两种符号：高电平、低电平（1、0）
 - 与门(乘)： $0 \& 0 = 0$ $0 \& 1 = 0$ $1 \& 0 = 0$ $1 \& 1 = 1$
 - 或门(加)： $0 | 0 = 0$ $0 | 1 = 1$ $1 | 0 = 1$ $1 | 1 = 1$
- 机器语言是计算机能看得懂的语言 “可执行文件”



程序实现：机器语言

- 由位（0/1）组成，通常按16位一组表示

... ..（这里离文件头2208个字符）

111100100000111100010000000010111111000010110000100000100000000011
0011000001111001011110100010111010100011101100000010110000011110010
0011111111111010110101000011110010
1110010000011110101100101000101110
1101000010000101100101000001000000
1111001001111001000001111010110010100110111010100111100100000111101
0110010100110111010100111100100000111101011000110000011111001000001
111000100010100111100100000111101011000110000011111001000001

第一行可以替换为机器语言
示例中的第一段红字

第二行可以替换为机器语言示
例中的第一段蓝字（以此类推）

指令替代零一序列，更直观

jbe: jump below equal

movsd: move single double-byte

某个可执行文件的内容
0：低电平；1：高电平
一般人完全看不懂

- 很难被人阅读，也很难被人编写

程序实现：汇编语言

- 用英文缩写符号作为指令，按一定格式组成

```
movsd      xmm0,mmword ptr ds:[0F558F8h]
comisd     xmm0,mmword ptr [t]
jbe        main+0FCh
or         eax,0FFFFFFFFh
jmp        main+14Ch
movsd      xmm0,mmword ptr [v0]
mulsd      xmm0,mmword ptr [t]
movsd      xmm1,mmword ptr ds:[0F55908h]
mulsd      xmm1,mmword ptr [a]
mulsd      xmm1,mmword ptr [t]
mulsd      xmm1,mmword ptr [t]
addsd     xmm0,xmm1
movsd      mmword ptr [s],xmm0
```

第一行可以替换为机器语言
示例中的第一段红字

第二行可以替换为机器语言示
例中的第一段蓝字（以此类推）

指令替代零一序列，更直观
jbe: jump below equal
movsd: move single double-byte

- 有编译软件可汇编语言翻译为机器语言

程序实现：高级语言（C语言为例）

- 由英文单词按一定的语法构成，较接近人类自然语言

```
movsd xmm0,mmword ptr ds:[0F558F8h]
comisd xmm0,mmword ptr [t]
jbe     main+0FCh
or      eax,0FFFFFFFFh
jmp     main+14Ch
movsd   xmm0,mmword ptr [v0]
mulsd   xmm0,mmword ptr [t]
movsd   xmm1,mmword ptr ds:[0F55908h]
mulsd   xmm1,mmword ptr [a]
mulsd   s = v0*t + 1.0 / 2 * a*t*t;
mulsd   xmm1,mmword ptr [t]
addsd   xmm0,xmm1
movsd   mmword ptr [s],xmm0
```

橙色方框中的源代码，等价于其对应所覆盖的汇编语言指令

易理解的语言

程序实现：自然语言

- 人类可以直接阅读和编写的语言

If the time is less than zero, it is invalid.
Otherwise, the shift s can be calculated by

$$s = v_0 t + \frac{1}{2} a t^2,$$

where v_0 is the initialized velocity, a is the acceleration, and t is the time.

— 不同人叙述习惯不同

程序实现：高级语言的优势

- 高级语言是人容易编写、容易生成机器指令的语言

语言	容易阅读和编写	容易生成机器语言
机器语言	☆☆☆☆☆	★★★★★
汇编语言	★★☆☆☆	★★★★☆
高级语言	★★★★☆	★★★★☆
自然语言	★★★★★	☆☆☆☆☆

- 不同的同级语言有不同的语法组织方式，但共有共性
 - 语法：构词的规则和组词成句的规则。
- 本课以C语言为例：介绍语法（个性）和语感（共性）

利用高级语言生成计算机指令

```
#include <stdio.h> /* Standard I/O header */
```

```
int main(char *argv, int argc)
{
    double v0 = 0; //initial velocity
    double a = 0; //acceleration
    double t = 0; //time
    double s = 0; //shift
    printf("Enter the Initial Velocity (m/s): ");
    scanf("%lf", &v0);
    printf("Enter the Acceleration (m/(s*s)): ");
    scanf("%lf", &a);
}
```

```
00000140: 00 01 11 00 00 00 00 44 3A 5C 44 65 73 6B 74 6F
00000150: 70 5C 73 68 69 66 74 2E 6F 62 6A 00 3A 00 3C 11
00000160: 00 22 00 00 07 00 12 00 00 00 0D 52 01 00 12 00
00000170: 00 00 0D 52 01 00 4D 69 63 72 6F 73 6F 66 74 20
00000180: 28 52 29 20 4F 70 74 69 6D 69 7A 69 6E 67 20 43
00000190: 6F 6D 70 69 6C 65 72 00 00 00 00 45 6E 74 65 72
000001A0: 20 74 68 65 20 49 6E 69 74 69 61 6C 20 5F 6F 66
000001B0: 6F 63 69 74 79 20 28 6D 2F 73 29 3A 20 6F 66
000001C0: 6C 66 00 45 6E 74 65 72 20 74 68 65 20 4F 6F 66
000001D0: 65 6C 65 72 61 74 69 6F 6E 20 28 6D 2F 6F 66
000001E0: 73 29 29 3A 20 00 00 25 6C 66 00 45 6E 74 65 72
```

```
.....D:\Desktop\shift.obj.:<
.. R.Microsoft
(R) Optimizing C
ompiler....Enter
Vel
..%
Acc
/(s*
nter
```

Library Code (.lib)

库代码

Start-up Code (.exe)

启动代码

```
00000000: 4D 5A 90 00 03 00 00 00 04 00 00 00 FF 00 00 00
00000010: B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000040: 0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 B8 01
00000050: 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6E
00000060: 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20
00000070: 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 00
00000080: 4A EC B7 36 0E 8D D9 65 0E 8D D9 65 0E 8D D9 65
00000090: D3 72 12 65 0D 8D D9 65 0E 8D D8 65 43 8D D9 65
```

Source Code (.c)

源代码

Compiler

编译器

Object Code (.obj)

目标代码

Linker

链接器

Executable Code (.exe)

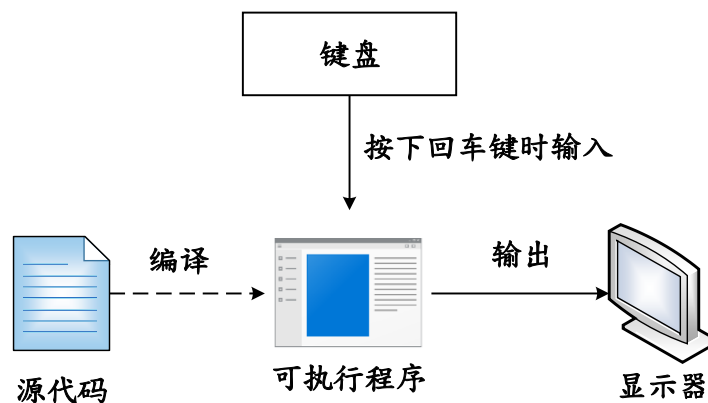
可执行代码

程序实现：源代码

- 程序员利用高级语言编写出源代码（Source Code）
 - 源代码：按照一定的程序设计语言规范书写的文本文件
 - 编程：编写程序，其核心过程为“程序设计”
- 源代码经过编译和链接形成机器指令
 - 编译器是将编程语言按照一定规律替换为机器指令的软件
 - 该规律是编程语言规定的，具有固定的格式和词汇
 - 必须在规律框架下严格遵守，否则会出错，达不到目的
 - 不同编程语言的格式和词汇不一样

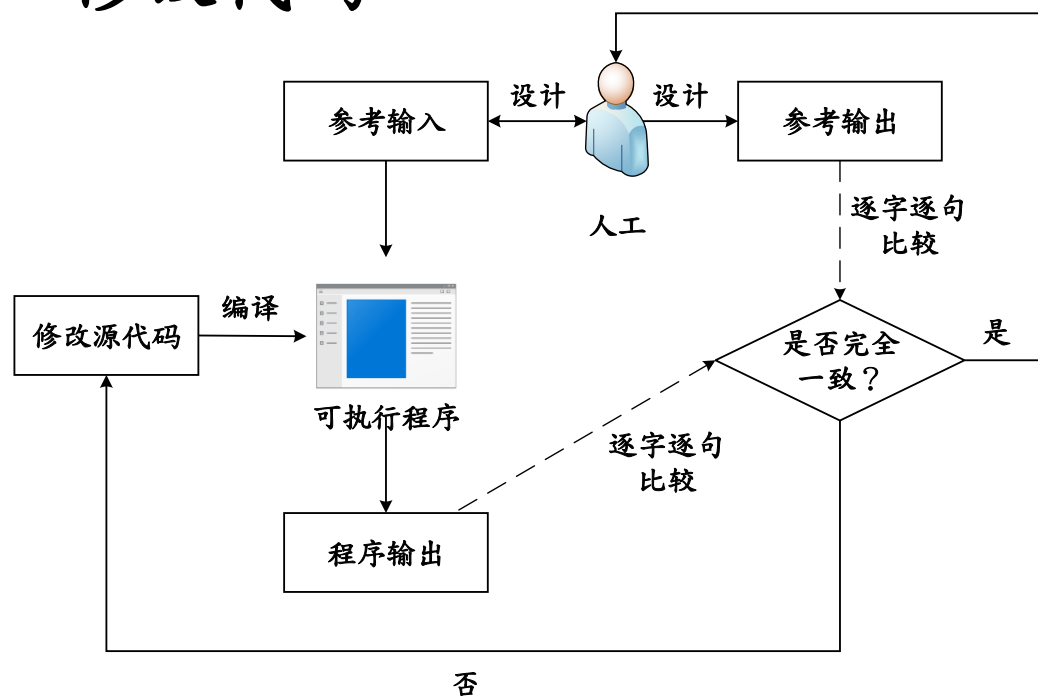
程序实现：程序运行

- 程序运行：计算机严格执行指令得到预期的结果
 - 程序一次实现，可以多次运行
 - 平时：计算机指令一般存储在磁盘文件（可执行程序）中
 - 运行时：将磁盘加载入内存，按时序在CPU调用执行
 - 通过双击图标或在命令行输入文件路径的方式运行



程序测试

- 将样例输入导入程序，对照输出是否为样例输出
- 自制测试数据，对照输出是否符合预期
- 通过调试手段找到错误，修改代码



课程目标

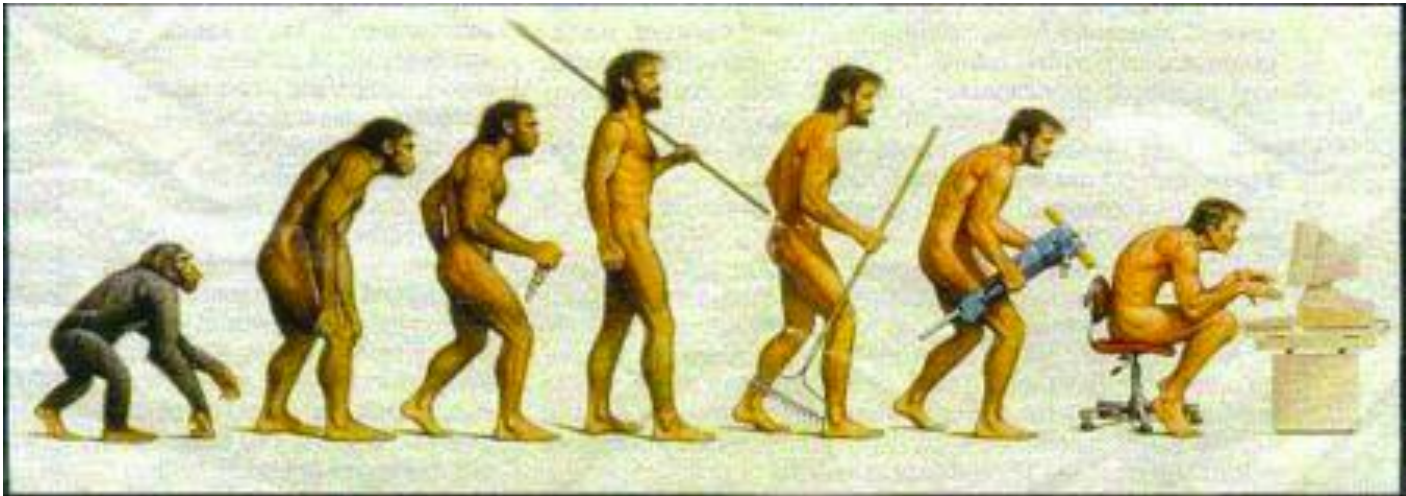
- 培养学生应用软件工程基本知识分析解决问题的能力
 - 了解题目 **需求**，**设计**解题方法
 - 用高级语言语法**实现**为源代码，**编译**成机器指令
 - **运行**、**测试**并修正源代码，重新编译机器指令
- 最终满足团队合作的技术要求

内容纲要

1	相关名词解释
2	程序员的工作
3	程序设计的意义
4	如何学习C语言
5	计算机基础知识（部分）

信息化技术

- 人和动物的根本区别在于：制造和使用工具
 - 工具：工作时所需用的器具
 - 引申：达到、完成或促进某一事物的手段
 - 对一件事归纳出流程，制成工具让后人使用



计算机硬件模拟人工

- 主板 (Mainboard)

- 控制器 (Controller)

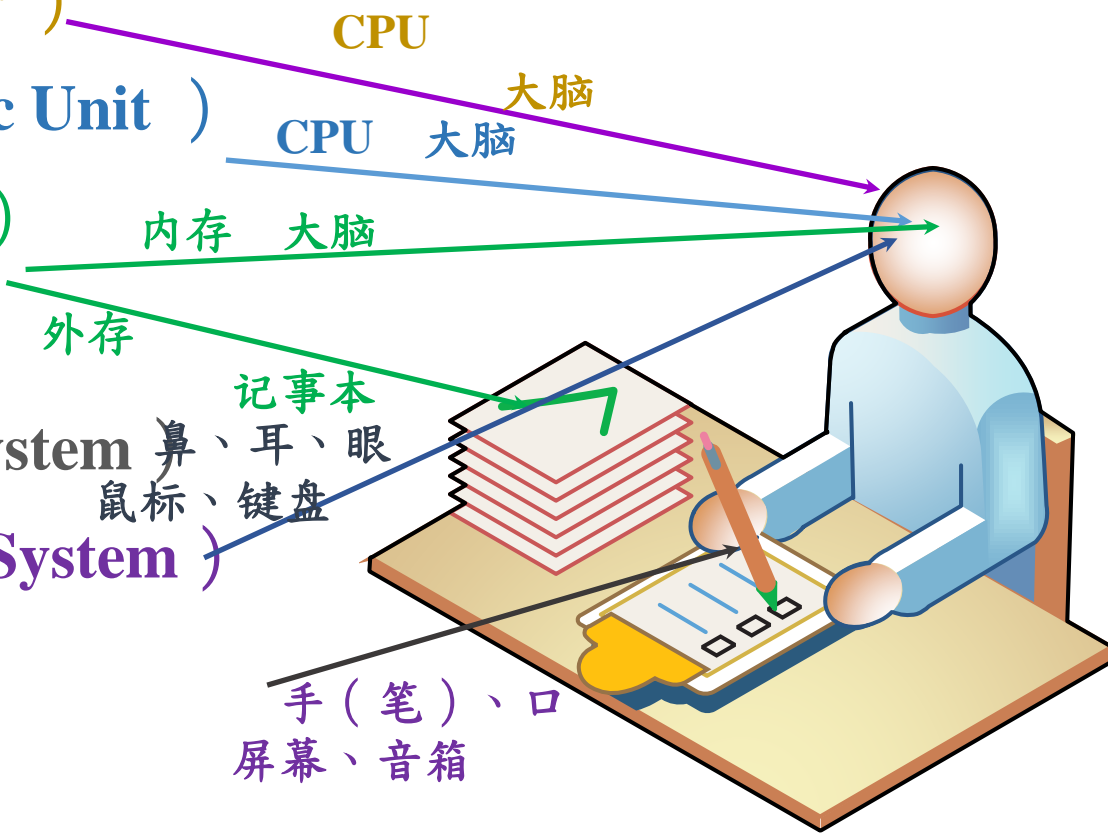
- 运算器 (Arithmetic Unit)

- 存储器 (Memory)

- 输入输出 (I/O)

- 输入系统 (Input System)

- 输出系统 (Output System)



计算机能做的不能做的

- 计算机能完成的功能

- 接受用户输入
- 运算处理数据：算术运算、逻辑运算等
- 向用户输出

- 计算机不能完成的功能

- 需求不明确的功能
- 除了计算以外的工作（仅限于生成指令）
 - 为用户泡一碗泡面
 - 电梯停靠升降

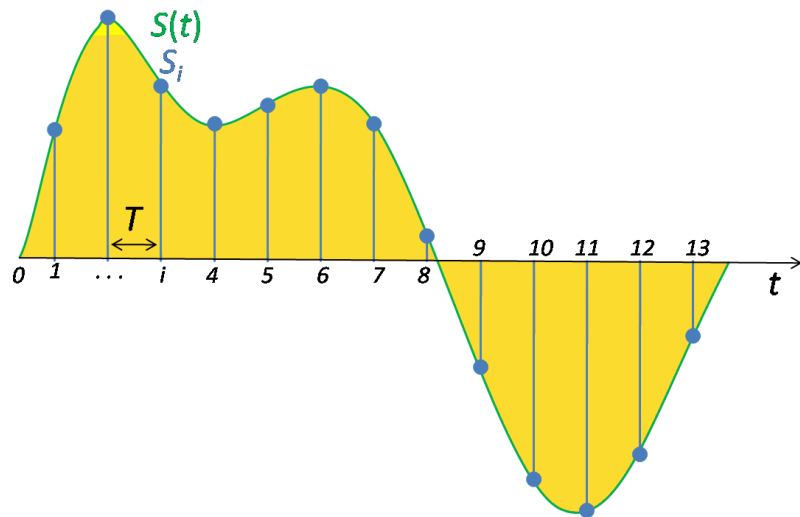
从现实社会到虚拟世界

- 现实世界：连续的

- 时间上的、数值上的，如：挥手，如：声波、光波
- 如果不能找到规律，则需要大量的存储能力

- 计算机世界：离散的

- 时间上：采样；数值上：编码
- 数字化
 - 1号时间、2号时间、……
 - 1号音量、2号音量、……



意义：抽象

- 小学：算术（计算技术），具体

- $5 + 3^2 = 5^2 + 3^2 + 2 \times 5 \times 3 = 64$

- 初中：代数（用字母表示数），表示规律

- $a + b^2 = a^2 + b^2 + 2 \cdot a \cdot b$

- 大学：教计算机解决问题

- 输入输出更宽泛：如一句话。

- 结构更多样

- 中间结果可以暂存 $\sum_{i=0}^N x + i^2$ $\text{sgn } x = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases}$

从现实社会到虚拟世界

- 把一个具体的生活问题，抽象成数学问题
 - 理顺输入与输出
 - 剔除应用场景，保留数值变化关系
- 把数量分成若干个等级
- 写出计算过程
- 完善计算过程
- 熟练以后，就不需要这么麻烦了

程序设计意义

- 利用计算机在重复、记忆的特长，简化人工操作
 - 重复：计算机做同样的事情不会觉得枯燥
 - 记忆：计算机存储可以长达数十年
- 用机器代替人工，大幅提高工作效率
- 以选课系统为例
 - 重复：每个人、每年的选课工作流程是重复的
 - 记忆：每个选课记录要记住至少四年，越久越好

内容纲要

1	相关名词解释
2	程序员的工作
3	程序设计的意义
4	如何学习C语言
5	计算机基础知识（部分）

初学者最常问的几个问题

- 多久才能学成

- 如果你每天都拿出大把的时间来学习，那么两三个月就可以学会C/C++，不到半年时间就可以编写出一些软件。
- 但是有一点可以肯定，几个月从小白成长为大神是绝对不可能的。要想出类拔萃，没有几年功夫是不行的。
- 学习编程不是看几本书就能搞定的，需要**你不断的练习**，编写代码，积累零散的知识点，代码量跟你的编程水平直接相关

初学者最常问的几个问题

- C语言如何学好：算法像数学，语法像英语
- 如何学到最好
 - 语文很重要：需求都搞错了，做什么都是瞎忙
 - 数学很重要：算出来的东西才最可靠
 - 英语很重要：很多重要文献都是用英语出版和交流的
 - 专业课很重要：我辈既务斯业，便当专心用功
 - 其它课程也很重要：做全面发展的好学生

内容纲要

1	相关名词解释
2	程序员的工作
3	程序设计的意义
4	如何学习C语言
5	计算机基础知识（部分）

位和字节的基本知识

- 位 (bit , b) 是二元的不确定性。
 - 有两种事件出现的概率各是1/2，其信息量是1 bit。
- N 位可以表示 2^N 元的不确定性。
 - 一共是 2^N 种情况，和事件的具体意义无关。
- 字节 (byte , B) : $1 \text{ Byte} = 2^3 = 8 \text{ bit}$
 - 因为 2^1 、 2^4 ，都无法表示26个字母，故而寻求 2^8 。
- KB (2^{10}B) 、 MB (2^{20}B) 、 GB (2^{30}B)

进制换算：看信息的角度

- 根据 x_i 和旧进制 N 算出 $x = \sum_{i=-\infty}^{\infty} x_i \cdot N^i, x_i \in 0, 1, \dots, N$
- 然后再根据新的进制 M 计算出 $x'_i = \frac{x}{M^i} \bmod M$
 - 其中 mod 的含义是在整数除法中取余数

例如：将10进制的9027转换为8进制数

$$9 \cdot 10^3 + 0 \cdot 10^2 + 2 \cdot 10^1 + 7 \cdot 10^0 = 9027$$

$$9027 / 8^4 \bmod 8 \approx 2 \bmod 8 = 2$$

$$9027 / 8^3 \bmod 8 \approx 17 \bmod 8 = 1$$

$$9027 / 8^2 \bmod 8 \approx 141 \bmod 8 = 5$$

$$9027 / 8^1 \bmod 8 \approx 1128 \bmod 8 = 0$$

$$9027 / 8^0 \bmod 8 \approx 1128 \bmod 8 = 3$$

答案：21503 （熟练后跳步）

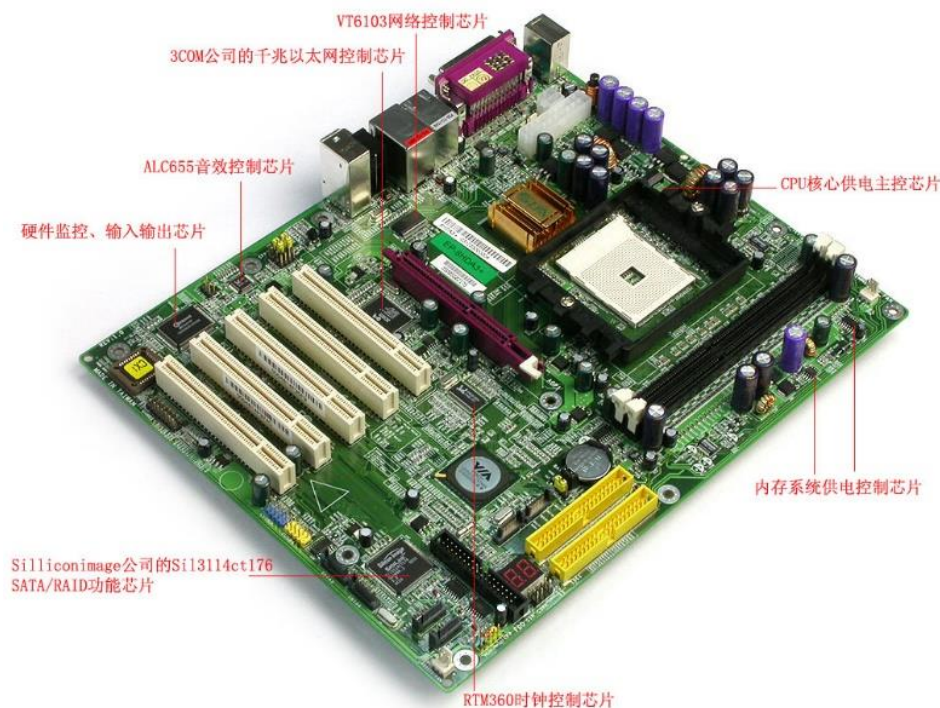
计算机硬件与软件

- 硬件是物理设备。
 - CPU、内存、主板、电源、硬盘、鼠标、.....
 - 显卡、网卡、声卡、.....
 - 音箱、耳麦、打印机、.....
- 软件是逻辑产品。
 - 操作系统软件和应用软件
 - Windows、Office (Word)、Visual Studio、.....

硬件：主板（Motherboard）

- 主板（Motherboard）

- 计算机（如：台式机或笔记本）内部的一块电路板，布满大小不一的电子器件，包括CPU、内存条、网卡、各种插槽和接口等。



硬件：CPU

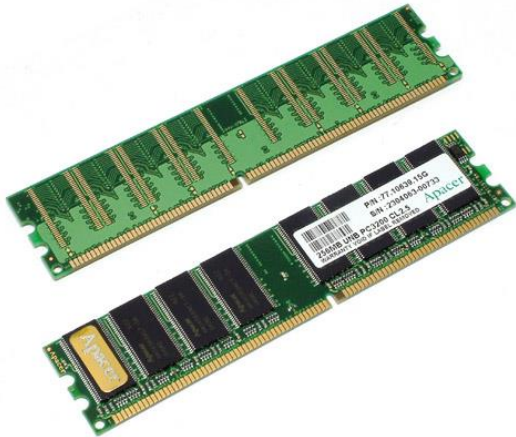
- CPU (Central Processing Unit) “中央处理器”
 - 大脑，负责计算、思考、处理数据、控制其他设备等
 - 主频、外频、总线频率、一级缓存、二级缓存、三级缓存
 - 指令集、多核心、虚拟化.....
- CPU的频率
 - 设CPU运算时的工作的频率为2.7GHz，
则其1秒钟运算 2.7×2^{30} 次。
 - 当然也受缓存频率的影响。



硬件：内存

- 内存

- 内存是下面的长方形“卡片”，也叫内存条。
- 影响内存价格的重要因素是：容量、频率（速度）
- 内存容量的最小单位是Byte



硬件：内存

- 虚拟内存 (Virtual Memory)

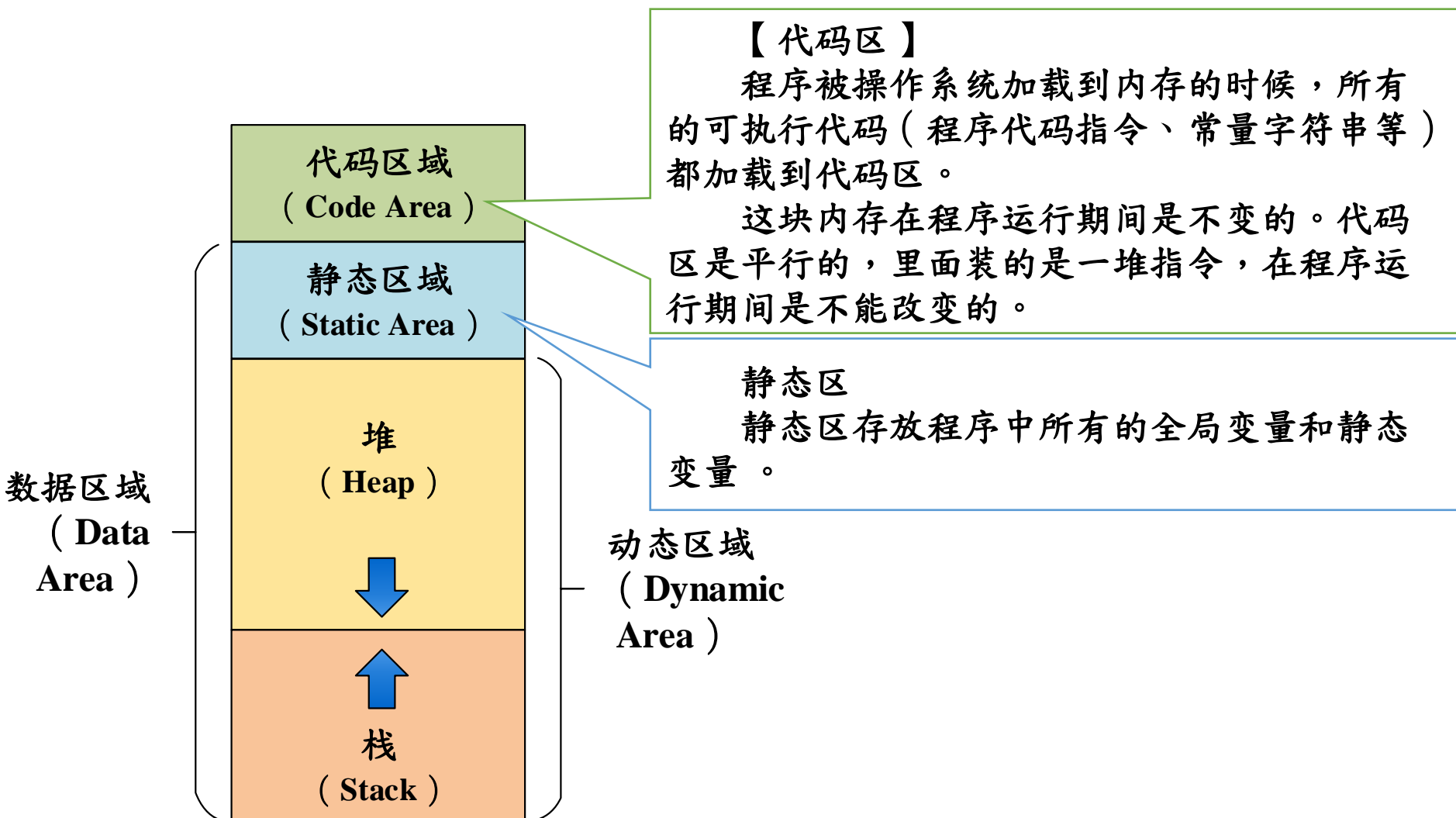
- 如果我们运行的程序较多，占用的空间就会超过内存（内存条）容量。
- 操作系统为我们解决了这个问题：当程序运行所需空间大于内存容量时，会将内存中暂时不用的数据写回硬盘；需要这些数据时再从硬盘中写入内存，并将另外一部分不用的数据写入硬盘。硬盘中就会有一部分空间用来存放内存中暂时不用的数据，叫做虚拟内存。

硬件：内存

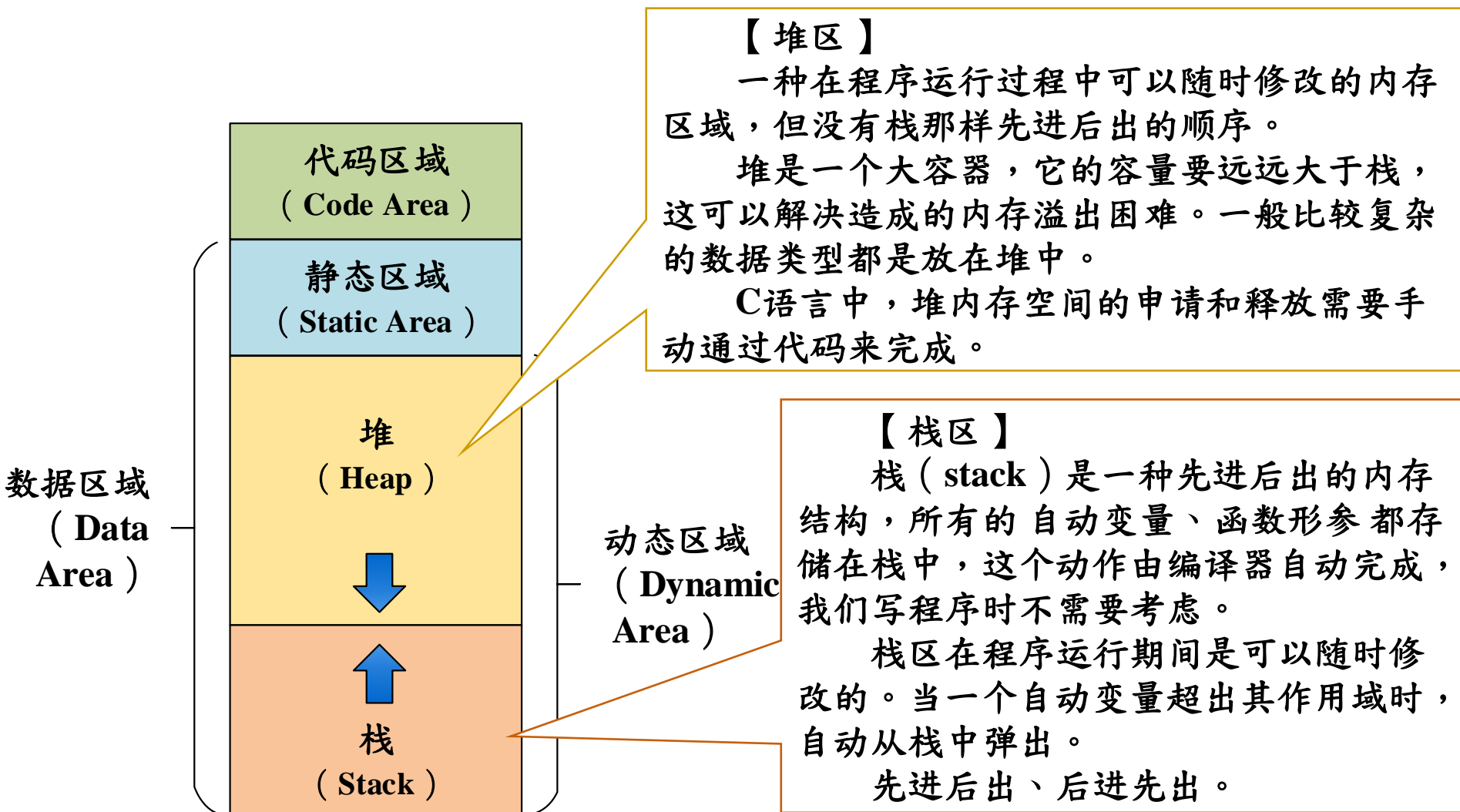
- 类似地

- CPU大部分操作数从寄存器（最快）读取
- CPU向存储器读取数据时，先访问高速缓存（Cache）
- 如果不存在所需数据，则逐级访问至内存
- 如果内存也不存在所需数据，则从硬盘加载至内存，存入高速缓存读取。

内存四区

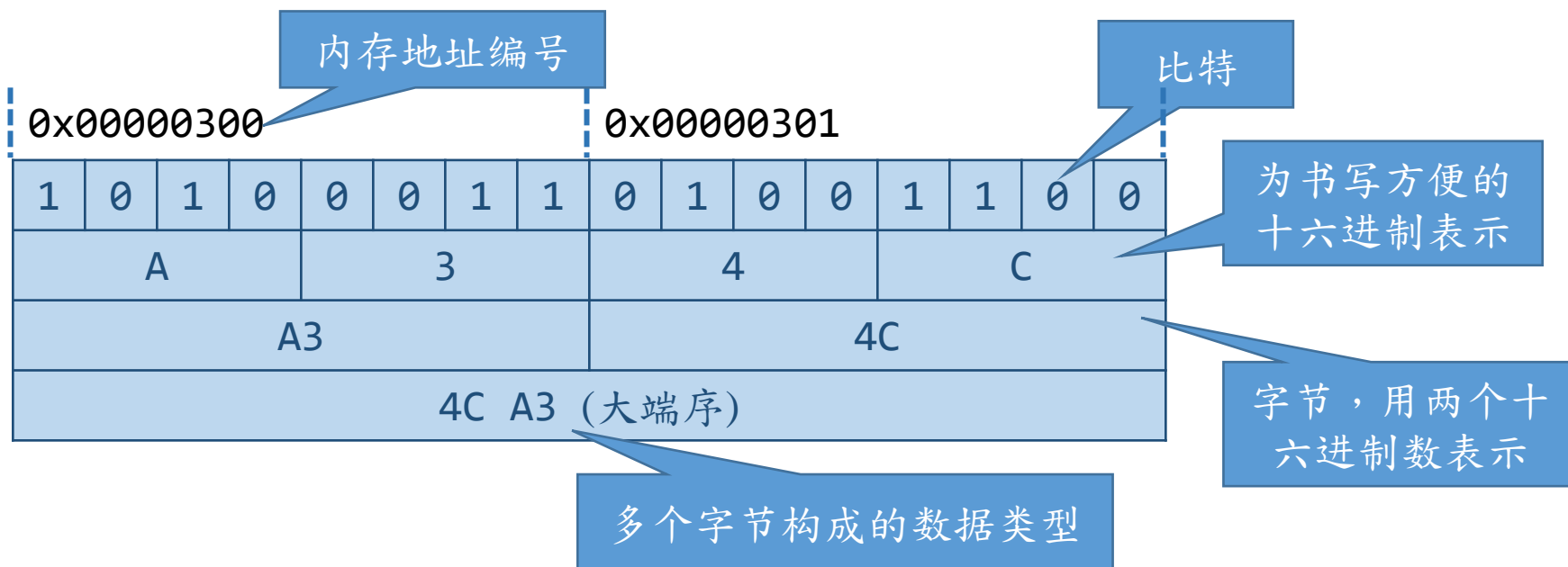


内存四区



硬件：内存的编址

- 内存的基本单位：Byte
- 把编号和物理意义分离开
- 一条4G内存共有0x00000000~0xFFFFFFFF个编号



硬件：硬盘

- 硬盘：用于保存数据，断电不会丢失。
 - 硬盘是外部存储
 - 不直接插在主板上，而是通过导线与主板相连
 - 与机械硬盘相比，固态硬盘读写速度有很大提升
- 文件系统：磁盘上组织文件的方法。
 - 包括NTFS、FAT32等。
 - 命名规则、时间戳、路径（绝对、相对路径）
 - 文件名+扩展名



硬件：I/O总线

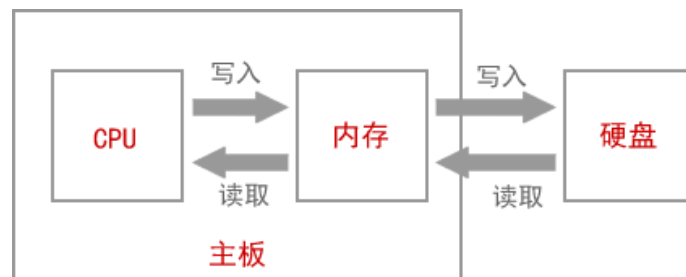
- I/O（输入/输出）总线和接口

- I/O操作

- 将输入设备输入的信息存入内存指定区域，或将内存指定区域的内容输出到输出设备。

- I/O控制器

- 接收来自CPU的I/O操作指令后，独立控制I/O设备的操作，直到I/O操作的完成
 - 不同设备I/O控制器结构和功能不同，复杂程度也不同



编程相关的软件

- 操作系统的分类
 - Microsoft Windows
 - Linux或UNIX
 - Android, iOS, Mac OS, ...
- 操作系统的作用
 - 管理和分配软硬件资源
 - 提供友善的人机界面

编程相关的软件

- 处理器管理

- 多任务管理：每个打开的计算机程序通过时间片轮流的方式被CPU执行

- 存储管理

- CPU大部分操作数从寄存器（最快）读取
- CPU向存储器读取数据时，先访问高速缓存（Cache），如果不存在所需数据，则逐级访问至内存；如果内存也不存在所需数据，则从硬盘加载至内存，存入高速缓存读取。

文件：文件系统

- 文件是一种抽象的概念，它提供了一种把信息保存在磁盘等外部存储设备上，并且便于以后访问的方法。
 - 这种抽象体现在用户不必关心具体的实现细节。
- 文件的属性
 - 大小、创建/修改/访问时间、只读、隐藏、存档权限
- 文件的操作：打开，关闭，读，写
- 操作系统一般采用多级目录结构，即树状目录结构或层次目录结构，其形状好似一颗倒立的树。

文件：文件系统

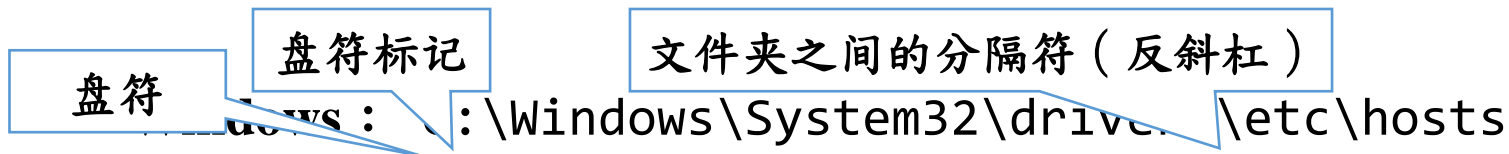
- 文件的分类

- 普通文件：ASCII文件，里面包含的是一行行的文本。二进制文件，通常具有内部的逻辑结构，被应用程序使用。
- 目录文件：用来管理文件系统的组织结构的一种系统文件，这是一种专用的特殊文件。
- 在UNIX中，还有两种特殊文件，其实就是输入输出设备，在UNIX中将输入输出设备看成是一种文件。包括字符特殊文件和块特殊文件。

文件：路径

- 绝对路径

- 从根目录开始到具体文件或文件夹的层次表示方法

Windows : C:\Windows\System32\drivers\etc\hosts

Callouts: 盘符 (C), 盘符标记 (Windows), 文件夹之间的分隔符 (反斜杠) (\)

- Linux : /etc/hosts

- 相对路径

- 当前路径

根目录符 (斜杠)

文件夹之间的分隔符 (斜杠)

文件夹名称

最底层文件 (夹) 名称

- 默认路径：操作系统定义的默认路径序列

文件：路径

- 相对路径

- 从当前目录开始到具体文件或文件夹的层次表示方法

- 假设当前路径为 "C:\Program Files\Internet Explorer"

上一级目录

再上一级目录

- Windows : ..\..\Windows\System32\drivers\etc\hosts

- Windows : \Windows\System32\drivers\etc\hosts

- 假设当前路径为 /etc

- Linux : ./hosts

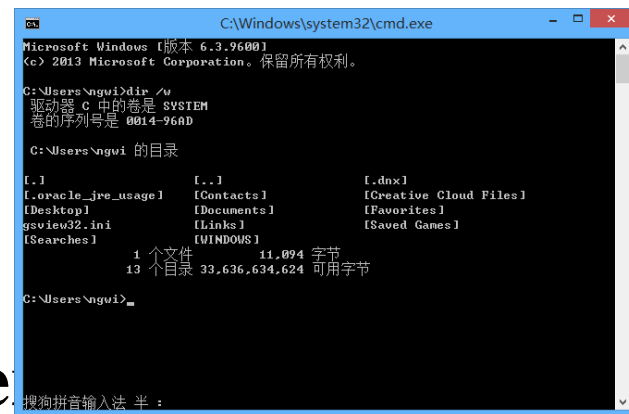
分区根目录

同一级目录

命令行基础

- 命令行用户接口 (Command User Interface)

- 通过输入命令行字符串的形式控制计算机
- 计算机通过显示字符反馈运行结果
 - 没有界面就少了一些崩溃的风险
- Windows：运行，输入cmd，回车。



```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 6.3.9600]
(c) 2013 Microsoft Corporation. 保留所有权利。

C:\Users\ngui>dir /w
驱动器 C 中的卷是 SYSTEM
卷的序列号是 0014-96AD

C:\Users\ngui 的目录

[.]                [...]          [.dnx]
[.oracle_jre_usage] [Contacts]        [Creative Cloud Files]
[Desktop]           [Documents]    [Favorites]
gsview32.ini        [Links]         [Saved Games]
[Searches]          [WINDOWS]

                  1 个文件          11,094 字节
                  13 个目录    33,636,634,624 可用字节

C:\Users\ngui>
```

- 图形用户接口 (Graphical User Interface)

- 鼠标单击相应控件
- 计算机在界面展示响应的图形或者窗口反馈运行结果

命令行基础

- 常用命令行 (Windows系统)

- 系统命令

- 切换盘符： C:
 - 切换当前文件夹： `cd C:\Windows`
 - 文件列表、删除文件、重命名、复制文件、新建文件、删除目录等：
 - `dir del ren copy md rd more pause exit`
 - 如果命令或参数中含有空格等特殊字符，应在前后加上双引号

- 可执行程序：exe，com，bat，cmd类型文件

- `format` (格式化，慎用！)，`xcopy`，`ipconfig`，.....

命令行基础

- 通过直接输入可执行文件名、指定参数来运行命令
 - 系统会在当前路径下查找，如果查无可执行文件，则在默认路径（通过PATH命令查看）下依次查找，如果找到则执行，否则提示不存在此文件。
 - Linux下可执行文件应具备执行权限
 - 示例

可执行程序文件名

第1个参数，如果参数中带有空格，应加上双引号，表示一个整体

第3个参数，一般紧接在开关后的参数，表示开关的配置信息

```
gcc "shift code.c" -o shift
```

第2个参数，可执行程序一般以减号开头的参数表示开关

编程相关的基础

- 全角和半角

- 全角：1 2 3 4 5 (占2个英文字符宽度，合1个汉字宽度)
- 半角：1 2 3 4 5 (占1个英文字符宽度)
- 全角和半角字符并不相同。(如：逗号，和 ，等)

- 字符



- ASCII
- Unicode

盲打：加快编程速度

- 多用键盘快捷键少用鼠标
- 盲打：在电脑上打字时不看键盘。
 - 盲打是打字员的基本要求，打字速度快，必须学会盲打。
 - 打字的每个手指分工正确，不要死记硬背
 - 键盘的八个基本键
 - A、S、D、F（左手）
 - J、K、L、；（右手）
 - 空格（双手拇指）



1

计算机基础

附加课程



廈門大學
XIAMEN UNIVERSITY



信息学院
(国家示范性软件学院)
School of Informatics

黃 煒
博士, 副教授
Dr. Wei Huang