Intro 2 CS

2019年6月15日 15:18

Log：2019年暑假完成，之后有若干次复习。
计算机导论笔记

1. 绪论
    a. 计算机的由来
        i. 算盘只能算是一个数据存储系统
        ii. 帕斯卡、莱布尼茨、巴贝奇的计算机基于齿轮，并且停留在理论上，因为当时的技术不足以制造出这些机器，同时钱也是个问题
        iii. 二十世纪时，马克一号（由艾肯和IBM合作制造）大量使用了电子控制的继电器，它们都过时了，因为与此同时已经有人在研究电子管建造的完全电子化的计算机。
        iv. 如：Colossus：被用来破译德国的情报。这类机器有十余台，由于军方保密而罕为人知。
        v. 不久，宾州大学制造了ENIAC
        vi. 后来肖克利、巴丁、布莱顿发明了晶体管，杰克基尔又发明了集成电路，电子计算机于是体积开始变小
        vii. 史蒂芬？沃兹尼亚克和乔布斯等极客制造了家用计算机，并且在1976年成立了苹果公司。当时人们还是青睐于IBM的计算机。
        viii. 1981年，IBM推出PC，里面的基础软件就是由微软开发的
        ix. 20世纪后期，蒂姆？伯纳德？李提出了WWW（万维网）的构想，然后由此人们开发了搜索引擎
2. 数据存储
    a. 存储器的结构
        i. 计算机的主存储器（main memory）又被称为RAM（随机存取存储器），因为它由独立的、可编址的存储单元组成，可以被任意的顺序访问
        ii. 刷新电路：有些存储器不是由触发器构建的，因而需要附加电路反复补充电荷。这种存储器称为动态存储器DRAM或dynamic memory
        iii. 有时会用SDRAM表示同步动态存储器
    b. 度量
        i. 在存储中，KB=1024 byte
        ii. 在数据传输中，Kb=1000 bit
    c. 海量存储器（mass storage）
        i. 指附加的存储设备，包括磁盘、CD、DVD、磁带、闪存驱动器
            1）联机（on-line）表示接入了计算机，脱机（off-line）表示断开连接
        ii. 不足：一般需要机械运动，所以比起由电子器件构成的主存储器而言要慢
        iii. 分类：
            1）磁学系统：
                a）比如磁盘。表面涂了磁介质可以存储数据。用存取臂和读/写磁头外加旋转磁盘读写。
                b）缺点：慢。
                c）优点：稳定，便宜，容量大。现在一般用来脱机档案数据存储中。在一点点地被弃用
            2）光学系统举例：
                a）CD（compact disk，光盘）。通过在光洁保护层上创建偏差来记录信息。一般600-700 MB。
                b）DVD（digital versatile disk 或 digital video disc），可以有好几个GB。由多个半透明层面构成，精确聚焦的激光可以识别其不同的层面。
                c）BD（blu-ray disk，蓝光光碟）：使用波长更小的蓝色而非红色激光，更加精确聚焦。容量是DVD

的五倍多。

3) 闪存（flash memory）

a) 磁学和光学技术都是通过物理运动存读信息，所以慢并且对物理运动敏感。而闪存通过二氧化硅晶格截获电子，并且能够保持截获的电子很多年，所以可以用来存储脱机数据

b) 反复的擦除刷新会逐渐损坏二氧化硅晶格，所以闪存只能用于比较少改变的设备中，比如数码相机、手机、PDA（personal digital assistant，掌上电脑）

c) 闪存设备被称为闪存驱动器（flash drive），容量可以有好几个GB。应用：

i) U盘。

ii) SD存储卡（secure digital memory card），简称SD卡。

One. 包括很多类型。一般的2GB左右，特殊的比如：

First. SDHC存储卡（secure digital high capacity memory card，高容量SD存储卡）可以有32GB

Second. SDXC存储卡（secure digital extended capacity memory card），容量可以有1 TB

iv. 文件存储以及检索

1) 存储设备上的数据块被称为物理记录，信息的自然划分（比如一个员工的信息由姓名、地址、工号等字段（field）构成）被称为逻辑记录。注意：键字段中的值被称为键（key）

2) 一条物理记录可能可以包含两条逻辑记录，一条逻辑记录有可能可以占据两条物理记录。

d. 用位模式表示信息

i. 文本的表示

1) ANSI采用了ASCII。ANSI代表美国参与ISO，ISO开发了很多ASCII拓展，比如Unicode。Unicode有16位，所以有65536种位模式

2) 注意区分文本编辑器产生的简单文本文件和Word等字处理程序产生的文件

ii. 数值的表示

1) 整数：

a) 二进制补码系统

补码complemental code

一的补码(one's complement) 指的是正数=原码,负数=反码

二的补码(two's complement) 指的就是通常所指的补码

b) 补码系统的好处在于方便计算，正负数可以直接计算

c) 余码计算法（excess notation）

i) 名字由来：比它们在传统二进制系统（没有负数）里的值少，如五位的余16，四位的就余8，三位的就余4

ii) 特点：就是二进制补码系统的符号取反，1表示正数，0表示负数。负数的一、二阶补码还是按照负数的标准来

2) 浮点数

a) 从左往右分为符号位、指数、尾数。

b) 注意：一个浮点数可以有好几种表达，要规范化（normalized）。

c) float:7位有效数字，double：15位有效数字

d) 截断误差：浮点数计算时产生的误差。由于有其存在，浮点数计算时的顺序就变得很重要了。另外，推荐通过进制将浮点数转换成整数。如不用美元而用美分作为单位。

iii. 图像的表示

1) 位图表示

a) 像素（pixel）的集合被称为位图（bit map）

b) RGB编码：红绿蓝各一个字节，表示相应成分的强度。所以每一个像素有三个字节

i) RGB的替代方案：用一个亮度成分（白光数量）、蓝色度、红色度表示

c) 缺点：不能轻易调节到任意大小，否则会出现颗粒状

2) 几何结构的集合

a) 用解析几何技术编码

b) 应用：

i) 字处理系统，如：TrueType（由微软和苹果联合开发）、PostScript（由Adobe开发）

ii) CAD（computer-aided design，计算机辅助设计）

iv. 声音的表示

1) 常用方法：对振幅不断采样记录。

a) 例子：

i) 远程电话通信一般每秒8000次

ii) CD每秒44100次。每次采样得到的数据以16位表示出来（用于立体声录制的话要16*2=32位）。所以录制立体声时一秒要100多万比特，3分钟就200Mb了

2) 乐器数字化接口（MIDI）：记录乐谱

a) 应用：电子键盘的音乐合成器（用来制作视频游戏的声音以及网站的辅助音效）

b) 实现：记录每个音符持续的时间，如：单簧管演奏D音符2秒钟，可以编码为3个字节，而不用每秒44100次采样

c) 缺点：录制的音乐在不同的合成器上的声音可能截然不同

e. 数据压缩

i. 行程长度编码（run-length encoding）

1) 例子：例如指出一个数据由253个1、118个0、87个1……构成

ii. 频率相关编码（frequency-dependent encoding）

1) 一般用赫夫曼编码（Huffman code）

iii. 相对编码（relative encoding），或者叫差分编码（differential encoding）

1) 记录数据流前后单元的区别

2) 例子：视频里前后帧一般差不多，那么只记录它们的差别就可以节省空间

3) 注意：可以是有损编码或者无损编码，取决于记录差别的精确度

iv. 字典编码

1) 将一个构造块编码

2) 例子：Word里有个字典，将很多单词都进行了编码，这样就比单独记录每一个字母的Unicode码要节省空间

3) 变体：自适应字典编码（adaptive dictionary encoding），或者叫动态字典编码

a) 思想：编码中字典可以随着编码而改变

b) 例子：LZW编码（LZW是三个提出者的姓氏）

i) 思想：将编码中更大的构造块加入字典

ii) 例子：编码内容：xyz xyz xyz xyz

一开始字典中x=1,y=2,z=3，然后空格代表一个单词结束。我们将xyz加入字典为5,所以最后编码为123454545.

iii) 特点：解码时又会重新生成最终的字典

v. 图像压缩

1) GIF（读作Giff或者Jiff）

a) 思想：将RGB编码的三个字节压缩到一个字节的256种颜色的调色盘中。所以有损

b) 注意

i) GIF中某个颜色通常被赋予透明色，所以被赋予颜色的地方的背景色可以透出来

ii) 可以通过LZW技术进一步压缩

2) JPEG

a) 得名于ISO的联合图像专家组（Joint Photographic Experts Group）

b) JPEG基线标准的选择模式：它的有损模式的别名
c) 应用：大多数数码相机将JPEG作为默认压缩技术
d) 注意：JPEG包含多种压缩方法，每一种都有自己的目标。可以有损或者无损。其中，由于有损压缩技术很成功，所以一般很少用它的无损模式。
e) 压缩步骤描述：
i) 由于人眼对亮度的敏感度高于颜色的，所以可以在一个2*2的模块中取四个像素的颜色平均值，亮度不变，信息减少75%。
ii) 将8*8的像素块作为一个单元通过离散余弦转换压缩
f) 效果：至少压缩10倍，甚至可以达到30倍，但是没有明显的质量损失
3) 还有TIFF。一般作为存储照片的标准格式，上面会附有日期、时间、相机设置等信息

vi. 音频与视频压缩
1) MPEG（Motion Picture Experts Group，运动图像专家组）
2) 在视频中：将一部分帧作为1帧，然后应用相对编码。
a) 注意：1帧本身可能会使用JPEG标准
3) 在音频中：即MP3（MPEG layer 3）
a) 利用的人耳特点：
i) 暂时模糊：人在一次巨响后，短时间里听不清轻柔的声音
ii) 频率模糊：某一频率的声音可以掩盖相近频率的轻柔的声音

vii. 图像差错
1) 原因：灰尘、表面的油脂、故障电路
2) 奇/偶校验：取决于最后整个数中的所有码含有的1的个数是奇数还是偶数
3) 汉明码
a) 特点：任意两个正确的位模式之间的汉明距离至少是3

3. 数据操控
a. 计算机体系结构
i. CPU的引脚在计算机主电路板（称为主板，motherboard）上。CPU的尺寸只有邮票的一般，很小，所以被称为微处理器（microprocessor）
ii. 早期的计算机要执行的步骤被内置于控制单元中，所以设计上可以方便地重新布线（通过拔插装置体现）
iii. 冯诺依曼结构使用了存储程序概念（stored-program concept），它视程序与数据为一体。当然冯诺依曼结构不是冯诺依曼提出的，他只是第一个在著作中转述了这一思想
iv. CISC（complex Instruction set computer，复杂指令集计算机）支持者认为：CPU越复杂越能应对日益复杂的软件。CISC的指令集十分功能强大
1) 代表：
a) Intel处理器（或AMD处理器）：由于制造工艺提升，成本大大降低，所以统一了台式机和笔记本电脑，甚至苹果的电脑也开始使用Intel的产品。但是耗电量确实大。
v. RISC（reduced Instruction set computer，精简指令集计算机）支持者认为：这样的计算机快并且便宜
1) 代表：
a) PowerPC处理器（由苹果、IBM、摩托罗拉联合开发），被用于苹果的Macintosh，2006年以后苹果改用英特尔的处理器了
b) ARM（advanced RISC machine）处理器（由高通、德州仪器等制造）：耗电低，被广泛应用于游戏控制器、数字电视、导航系统、汽车部件、移动电话、智能手机等消费性电子产品中
vi. 其他见计算机组成原理课程。补充内容：
1) 掩码（mask）：如想将一个二进制的数的某些位置上的数掩盖掉（masking，即屏蔽），就可以让它和一个精心设计的掩码做与操作。掩码中0的部分就是我们要掩盖的部分。
做或操作也行。也叫掩码。
2) CPU和主存储器通过总线（bus）相连，总线上其实也连着控制器，而每一个控制器都和一个外围设备

(比如CD驱动器、调制解调器、监视器、磁盘驱动器）相连

a) 作为中间设备，控制器一端可能是永久的安装在主板的电路上，也可能是可以和主板的插槽相连；另一端则可能是直接连上外设，也可能连上端口（port），然后外设和端口相连
b) 控制器可能本身就是小型计算机，有自己的存储电路和简单的CPU
c) 最初，控制器和外设是一一对应的，想接入新的外设就要有新的控制器
d) 后来出现了USB（universal serial bus，通用串行总线）控制器和FireWire（火线）控制器
    i) 于是，外设的信号只要转换成USB和FW标准就好，USB和FW控制器只要将计算机内部信号转换成USB和FW标准
    ii) USB由英特尔开发，技术成本低，兼容打印机、扫描仪、数码相机、鼠标等
    iii) FireWire由苹果开发，传输速度快，集中用于高传输速度要求的设备上，比如摄像机和联机海量存储系统
e) 有两种CPU和控制器交互的设计
    i) 存储映射输入/输出（memor-mapped I/O）
        One. CPU直接使用load和store操作，每个控制器响应唯一一组地址，而主存忽略这些地址。
        这样，计算机的输入输出好像就是在存储器里进行的，无缝结和外设
    ii) 在机器语言里提供对控制器的操作（称为I/O指令），如：将寄存器5的内容存储在A3指定的控制器中
f) 直接内存存取（direct memory access，DMA）
    i) 因为控制器也是直接连在总线上，所以可以在总线没有被CPU占用的时候和主存直接交流。
    ii) 优点：不耽误CPU，效率高。比如CPU要求读磁盘的信息并把信息放到主存里，然后控制器去读了，读完直接在总线空闲时交给主存，在此期间CPU可以继续执行别的指令。
    iii) 缺点：出现了冯诺依曼瓶颈。DMA导致CPU和控制器争总线，使得总线的通讯更加复杂

3) 握手（handshaking）：外设和计算机交流（有的系统里是和控制器，有的是和CPU），比如：打印机要告诉计算机现在别传文件过来了，打印不过来，文件会装不下然后丢失了；然后打印机发了一个状态字（status word），表示缺纸或者接受不下等状态
4) 流行的通信媒介
    a) 串行通信与并行通信
        i) 并行（parallel）：不同的信号都有各自的路线
            One. 优点：快
            Two. 应用：计算机的总线
            Three. 缺点：需要复杂的线路
        ii) 串行（serial）：在一条线路上一个接一个地传
            One. 优点：只要一根线路
            Two. 应用：USB和FireWire、以太网
    b) 传统拨号连接：
        i) 实现：先通过调制解调器（modulator-demodulator，缩写为modem，中文戏称作猫）把数字信号转换成和语音模拟信号频率一样的模拟信号，通过电话线传输，然后在另一头通过猫又将模拟信号转为数字信号
        ii) 通信速度：优化后可达57.6 kb/s，而MP3都要64 kb/s
        iii) 特点：便宜，但是很慢
    c) DSL（digital subscriber line，数字用户线路）
        i) 基于的事实：电话线不止能传输语音通信要用的频段。可以用高于语音通信的频段来传输网络数据，就是来传输网络连接用的数字信号
        ii) 升级：用光纤线路
        iii) 通信速度：可达54Mb/s

iv) 竞争对手：用于有线电视的电缆、通过高频无线电广播的卫星链路

d) USB/Firewire可达几百Mb/s

e) 带宽（bandwidth）：大致等同于传输的最快速度。高带宽意味着该通信路径传输信息又快又多

5) 并行处理技术

a) 光速大概1英尺每纳秒。所以如果CPU想取1英尺外的存储单元中的指令，一去一回至少要2ns

b) 多核CPU：一个芯片上可以有好几个CPU共用cache（高速缓冲存储器）

c) 得益于多核处理器，MIMD（multiple-instruction multiple-data）简化了

i) 描述：一个处理器遇到任务时，可以将部分任务放在公共存储器中，让别的存储器来帮忙。于是不同的指令在不同的数据集上进行

d) SIMD（single-instruction multiple-data）：

i) 适用领域：对一大堆数据中的每一项都做一样的操作

4. 操作系统

a. 分类

i. UNIX：服务于较大的计算机系统和PC群

1) Mac OS

2) Solaris

ii. Linux

1) 来源：最初由Linus Torvalds在赫尔辛基大学学习时设计的

iii. Windows

b. 许多操作系统的核心都是在解决长期以来的问题

c. 历史

i. 几个用户共享一台电脑时，操作系统提供签名表，记录预定的时间。一台电脑在某个时间段里面只归一个用户

ii. 一开始有个电脑操作员，负责交接用户的程序，

iii. 批处理（batch processing）：把一些job（一个程序的执行）集中成一个批次，然后统一执行无需交互

iv. job queue（作业队列）：FIFO。外加一些优先级

v. 分时（time-sharing）：

1) 目的：为了让一台电脑同时处理

2) 可能的实现方式：多道程序处理（multiprogramming）

a) 描述：电脑的运行时间被分隔成一片片的时间片（time slice），每个时间片中执行一个任务，时间结束了就换下一个时间片和相应的的任务。由于每个时间片时间都很短暂，所以看起来就像同时在执行

b) 注意：多道程序处理也可以用于单用户系统，这时称为多任务（multitasking）

vi. 分隔任务的目标

1) 负载平衡（load balancing）：动态地把任务均等合理地分给各个处理器

2) 均分（scaling）：把大任务分成几个子任务，与可用的处理器数目相适应

3) 计算机网络出现后还要考虑多个用户和多个计算机的任务分配

vii. 嵌入式：用于完成特定任务的设备，比如手机、车载电子设备、医疗设备、家用电器

1) 相关系统：VxWORKS、Windows CE（也就是Pocket PC）、Palm OS（主要用于手持设备）、安卓

d. 软件分类

i. 应用软件（application software，用来完成特定任务的）：如数据库系统、游戏

ii. 系统软件（system software，完成一般的计算机系统都需要完成的任务）：提供了应用软件所需的基础架构

1) 实用软件（utility software）：不包含在操作系统中，用于扩充操作系统的功能，如数据压缩和解压软件、多媒体播放软件

2) 操作系统本身
    a) 用户界面（user interface，UI）
        i) shell：
            One. 描述：输入指令
            Two. 分类
                First. UNIX有Bourne shell、C shell、Korn shell
                Second. 微软有DOS shell cmd.exe
                Third. 苹果的OS X保留了Terminal作为实用软件，它承袭了UNIX shell
        ii) GUI（graphical user interface）
            One. 分类：
                First. UNIX用户可以有X11
            Two. 重要组件
                First. 窗口管理程序（window manager）
                    1. 原理：负责分配、跟踪、联系窗口，生成GUI样式。获取用户鼠标点击图标（icon）的位置，传给相应程序
                    2. 补充：Linux用户甚至可以选择窗口管理程序
                        1. KDE
                        2. Gnome
    b) 内核（kernel）
        i) 文件管理程序（file manager）
            One. 作用：协调海量存储器的使用
            Two. 详细说明：记录了每一个文件的位置、不同用户的访问权限、剩余空间的位置
            Three. 目录路径：只有Windows是用反斜杠\区分
            Four. 每次软件申请访问文件时，先看看权限
        ii) 设备驱动程序（device driver）
            One. 作用：和控制器或直接和外设通信
            Two. 每次软件要使用外设时，直接找相应的设备驱动程序，而不用自己去处理技术细节
        iii) 内存管理程序
            One. 作用：协调管理主存
            Two. 在多用户、多任务时，负责合理分配限制内存空间，跟踪回收内存
            Three. 在设计虚拟内存时，还要负责页面调度（paging）（在内存和海量存储器之间切换程序和数据），造成有额外内存的假象
        iv) 调度程序（scheduler）
        v) 分派程序（dispatcher）
iii. 注意：这种分类很模糊，比如网络通信软件在其研发期时，被视为应用软件，现在成了实用软件；微软曾经面临的反垄断诉讼案核心也在于浏览器和媒体播放器究竟是操作系统的一部分还是微软用来压制对手的实用软件

e. 系统启动
    i. 启动（booting）计算机包括执行引导程序和启动操作系统
    ii. 引导（boot strapping，简称booting）：由于主存是易失性存储，所以每一次开机都需要boot。开机时ROM（只读存储器，一般用闪存制作）中含有的boot程序引导CPU将操作系统从海量存储器中加载复制到主存中
    iii. 一旦操作系统被放入主存，引导程序就引导CPU跳转到这个存储区
    iv. 操作系统的原始存放位置千奇百怪
        1) 在嵌入式中是闪存制造的ROM
        2) 在大公司或大学的小型工作站中，可能要通过网络从远程机器上复制

3) 在PC中是海量存储器

v. 固件（firmware）

1) FlashROM中除了引导程序，还有一些别的例行程序，实现基本的IO。比如从键盘上接受信息以和用户交互，把信息显示在屏幕上以报错或者交互

2) 因为既不是芯片那样被固化了的硬件，也不是海量存储器中可以轻易更改的软件，所以称为固件

3) 被广泛使用的固件系统有：

a) PC中一直在用的BIOS（basic I/O system，基本输入/输出系统）

b) Open firmware（SUN公司的，现在是Oracle公司的一个产品）

c) 嵌入式的CFE（common firmware environment，通用固件环境）

vi. 不在PC中直接用FlashROM装操作系统的原因

1) 效率低。所以只有嵌入式这样做

2) 操作系统经常更新

f. 多道程序设计下的协调机器的活动

i. 使用多道程序设计的目的

1) 提高效率，因为等待外设或用户输入都很浪费时间

ii. 进程（process）：某个程序的执行

1) 它的状态被称为进程状态（process state）

iii. 进程表（process table）：用来跟踪记录每一个进程的状态

iv. 就绪（ready）：某个进程已经准备好了，随时可以开始

v. 等待（waiting）：可能是在等用户敲下键盘、获取海量存储或者是其他进程的反馈

vi. 不同进程每次都占用一个时间片，然后进程间的切换称为进程切换（process switch）或者上下文切换（context switch）

vii. 中断（interrupt）：用来终止一个时间片

viii. 中断处理程序（interrupt handle）：当CPU收到中断信号时，会首先完成当前的机器周期，保存它在当前进程的位置，然后就开始执行中断处理程序

ix. 有的CPU提供机器语言以重新恢复某个进程上次中断时的状态

g. 处理进程间的竞争

i. 矛盾：可能好几个进程同时需要一个不可共享的资源，比如都要打印机

ii. 初步处理方案：使用信号量（semaphore）作为标志，表示该资源是否已经被占用了（置位set或清零clear）

iii. 初步方案的不足：为了获取这个标志的值，可能需要好几条机器指令，然后由于我们使用了多道程序设计，可能会进程间不同步，如：进程A发现打印机空闲后没来得及更改打印机的状态就结束了时间片，恰好随后的进程B发现打印机是空闲的然后占用了它，等到进程A重新得到了时间片，它根据上一次结束时遗留的信息误以为打印机是空闲的，然后也去占用打印机

iv. 解决方案：

1) 很多机器语言中都有中断屏蔽指令和中断允许指令

a) 在中断屏蔽指令和中断允许指令出现之间的时刻，中断信号无效，所以进程就不会被干扰

2) 使用一种复合指令：测试并置位指令（test-and-set）。这一条指令就可以完成检索标志并且置位它的两个操作。由于中断前时必须完成当前的指令，所以这两个指令不会分开来在两个时间片中执行

v. 补充：一个信号量守护一段互斥（mutual exclusion）的程序，这段程序我们称为临界区（critical region）

vi. 死锁（deadlock）

1) 出现的条件

a) 竞争不可共享的资源

b) 一个进程接受了某些资源后，稍后还将请求其他资源

c) 资源一旦被分配，就无法强制回收

2) 分析：只要这三个条件有任意一个不满足，则死锁不会发生
3) 解决方案：
a) 清除（kill）某些进程
b) 死锁避免方案
i) 要求一个进程一次性请求所有的所需资源，这样就不会得陇望蜀
ii) 使用假脱机（spooling）技术，将不可共享的资源变成好像是可以共享的，比如：给打印机一个task queue，把要打印的东西放在海量存储器里，于是每一个请求者都觉得打印机可以共享，可以满足自己的需求
iii) 假脱机变体：不同进程都可以请求读同一个文件，但是一般不能写同一个文件。可以使用假脱机，让不同进程写文件的不同的区段。当然这样也有很多技术细节要考虑

vii. 安全性
1) 来自外部的攻击
a) 审计软件（auditing software）
i) 作用：记录分析系统中的各种活动对错误或异常活动进行处理
ii) 例子：
One. 一个普通用户平时一直是只使用word和excel，某天突然用起了系统中非常专业的一个软件，这说明可能有非授权用户访问了这个账号
Two. 嗅探软件：一个程序模拟了一个登入界面，欺骗用户把账号和密码输入进去。而审计软件就可以用来防止这种嗅探软件
2) 来自内部的攻击
a) CPU中有若干专用的寄存器用来记录一个进程存储区域的上下界
b) 非特权用户（不是administrator或super user）不能修改这些寄存器的值

5. 组网及因特网
a. 网络分类
i. 按作用域
1) 局域网（LAN，local area network）
2) 城域网（MAN，metropolitan AN）
3) 广域网（WAN，wide AN）
ii. 按开放度
1) 开放式（open）网络
2) 封闭式（closed）网络，也叫做专用（proprietary）网络
iii. 按拓扑结构
1) 星型拓扑
a) 在无线网络中广泛使用
b) 中央计算机称为AP（access point，接入点）
2) 总线型拓扑
a) 以太网
b) 中央位置被称为集线器（hub）
3) 区别：各个计算机是通过总线直接通信还是通过中央计算机间接通信
4) 二者区别有时很小。比如hub很小时总线型网络像是星型网络
b. 协议（protocol）
i. CSMA/CD（carrier sense，multi-access with collision detection，带冲突检测的载波侦听多路访问）
1) 如果不同的计算机在相近时刻发送报文，它们都能发现彼此冲突，然后都暂停了一段随机长度的时间就再次发送报文
2) 用于以太网
ii. CSMA/CA（carrier sense，multi-access with collision avoidance，带冲突避免的载波侦听多路访问）

1) 当一台计算机想发送报文时，它必须先等一等，如果这段时间里信道都处于空闲状态，则发送，否则就再等一等。这样可以保证占用信道的都是等待时间最久的。如果冲突还是发生了，就暂停了一段随机长度的时间再重新发送报文
2) 用于星型网络，比如WiFi（无线保真）
3) 隐藏终端问题（hidden terminal problem）：在星型网络中，终端计算机都能向中央计算机发送报文，但是它们无法检测到别的终端是否发送了报文给中央计算机
4) 如何确认信道空闲：一些WiFi中要求每台计算机想使用信道时都发一条请求报文给AP，如果AP空闲，就会回复表示空闲，否则不回复

c. 网络互连
   i. 总线互连
      1) 中继器（repeater）
         a) 作用数量：只能是2条总线
         b) 不能筛选报文，只会无脑传递，最多放大一下报文
      2) 网桥（bridge）
         a) 作用数量：只能是2条总线
         b) 可以筛选报文，只有当报文的地址是另一边的计算机时才会传输过去。比中继器高效
      3) 交换机（switch）
         a) 类似车轮的辐条，可以连接不止两条总线
   ii. 连接以太网和WiFi网
      1) 路由器（router）
         a) 名字由来：它收到报文后会向适当的地方转发报文
         b) 计算机连接接入点，接入点连接路由器
         c) 每一台计算机都有两个地址：它在原始网络（不用路由器时的网络）中的地址和在互联网（Internet）中的地址。每个路由器都维护了一张转发表，上面记载了不同目的地地址的信息应该往哪发
      2) 网关（gateway）
         a) 一个网络（internet，首字母要小写）和因特网（Internet，首字母大写）连接的点
         b) 虽然网关有多种形式，但是一般指路由器
         c) 有时指的是路由器和AP（接入点）。因为这两个东西经常安装在一个单元里

d. 进程间通信（interprocess communication）的方法
   i. 客户机/服务器（client/server）模型
      1) 例子：打印服务器、文件服务器
   ii. 对等（peer-to-peer，简称为P2P）模型
      1) 例子
         a) 两个人用微信聊天（发送即时消息）
         b) 分发文件（对等体集合被称为蜂群（swarm））：盗版高发地带
      2) 注意：对等是指两个进程通过网络来进行通信，而不是指网络本身的特性。所以我们可以说一个进程通过了对等模型通信，不能说它通过对等网络通信。对等和服务器模型都是通过网络进行的
   iii. 分布式系统
      1) 集群计算（cluster computing）
         a) 优点：
            i) 高可用性（high-availability）：不会一下子所有的计算机都瘫痪，一般总有一些可以用
            ii) 负载平衡（load-balancing）：负载可以自动地在集群中调节转移
      2) 网格计算（grid computing）
         a) 和集群相比耦合度比较低
         b) 需要专门的软件来发布数据和算法

i) 威斯康星大学的condor系统
ii) BOINC（Berkeley‘s open infrastructure for network computing）
c) 各种办公用的PC和家用的PC都可以自愿称为网格的一员，在空闲时提供算力
3) 云计算（cloud computing）
a) 云指的是网络上大量的可用的计算机资源
b) 就像当年电网的发展使得工厂和企业不用维护自己的发电机一样
c) 担忧：隐私和安全性得不到保障
iv. 因特网（Internet）
1) 结构
a) 总体而言，由两层ISP（Internet service provider，网络服务提供商）来完成网络的构建和维护。
b) 然后由access ISP（网络接入服务提供商，比如一些公司、大学）负责作为个人用户和ISP之间的中介。
c) access ISP向个人用户（终端设备（end system）或主机（host））提供因特网接入，比如通过WiFi
e. 因特网编址
i. 每一台计算机都有一个IP（Internet protocol）地址，这个地址是由ICANN（因特网名称与数字地址分配机构）向因特网服务提供商提供的
ii. IP地址采用了点分十进制计数法（dotted decimal notation）。本来地址是二进制的，然后转换成十进制时以点隔离。如：192.7.177.12就是四个字节，每一个字节是一个数字
iii. 助记名：
1) ICANN指定了一些注册商，注册商在ICANN那里注册域名
2) 顶级域名（top-level domain，简称为TLD）：如edu、info（表示无限制使用）、museum、org（非盈利机构）、cn、ca、au
3) 拥有域名的机构可以向左任意扩展名称，比如创一个子域（subdomain）
4) 例子：r2d2.compsc.nowhereu.edu表示顶级域名为edu、域名为nowhereu、子域为compsc、名为r2d2的计算机
5) 域名系统查找（DNS lookup）
a) 就是通过域名系统解析域名：一个人如果通过助记地址给别的电脑发信息，就会通过域名服务器（name server）的DNS（domain name system，域名系统）转换成IP地址
6) 这个拥有域名的实体如果有资源维护它的域名服务器的话，它就成了access ISP，为它域内的所有名称提供解析服务
7) 一个机构如果很小所以不打算自己维护自己的域名服务器的话，就可以和ISP签订合同，付费让ISP来做
f. 因特网应用
i. 电子邮件
1) 如果要提供邮件服务，一个域要有邮件服务器，当一个人发一封邮件给这个域时，邮件会首先保存在邮件服务器里面，然后如果收件人要查看的话就展示出来
2) 邮箱@后的所有内容都是表示接受这封邮件的机构的地址
ii. 文件传输协议（file transfer protocol，FTP）
1) 传输文件时，FTP使用客户机/服务器模型
2) 服务器（称为FTP站点）可以通过口令限制文件的访问权
3) 如果一个FTP站点不限制文件的访问权，那它应该把通用口令设置为anonymous（匿名），因此这个站点就被称为匿名FTP（anonymous FTP）站点
iii. 远程登陆与SSH（secure shell，安全shell）
1) SSH：给传输的数据加密和验证

iv. 四种VoIP（voice over Internet protocol）
    1) VoIP软电话：通过P2P软件允许两个或者多个PC共享一个电话，只要扬声器和麦克风就可以为用户提供通话。
        a) 比如Skype：它也可以让用户和传统电话通信系统连接起来。但是为了接受电话，用户必须保持Skype时刻连通，因而PC的一些资源就会被用来支持其他人的Skype通话
    2) 模拟电话适配器（analog telephone adapter）
        a) 允许用户将传统电话连接到access ISP的电话服务上
        b) 经常被和数字电视捆绑在一起
    3) 嵌入式VoIP电话
        a) 把传统的基于铜线的电话系统换成基于以太网的TCP/IP网络上
        b) 很多大型组织都这样做以减少电话开销，同时还能增强功能
    4) 4G电话完全就基于IP，本质上就是一台主机
v. 无线电话的演进：
    1) 1G：通过空气传输模拟语音信号，就是没有铜线罢了
    2) 2G：使用数字信号对语音编码，能够高效地使用无线电波，还能发短信
    3) 3G：速度更快了，支持视频通话等对宽带要求比较高的活动
    4) 4G：速度更快，能使用IP协议完全进行分组交换的网络
vi. 因特网广播
    1) 常规思路：使用多点传播（N-unicast）技术，谁要看就给谁发送。1000个人要看就向不同目的地发1000份
    2) 问题：这样的话和这个服务器相邻的因特网邻居直接就炸了，因为所有这些信息必须靠服务器的邻居转发
    3) 改进方法1：使用P2P模型。一旦一个对等体收到数据，就把数据发给其他对等体
    4) 改进方法2：使用多路广播（multicast）。这时广播的目的地址称为组地址，一个客户端想要接受这个信息时就等于是要订购这个组，然后离它最近的这个路由器就会把这个需求传到因特网上，接着所有的路由器都会把带有这个组地址的信息往这个客户端传。
       补充：这种方法还是比较新的，还在逐渐流行之中。因为它对路由器提出了新要求

g. 万维网
i. 引入概念：
    1) 超文本（hypertext）：这个概念的意义随着使用慢慢发生了变化，原来指的是含有指向其他文档链接的文本文档，现在等价于超媒体（hypermedia），就是说含有指向其他图像视频等的文档就行，像维基百科那样
    2) 浏览器：客户端给客户提供的一个界面
    3) 服务器软件包（通常称作万维网服务器，Web server）：响应浏览器访问服务器计算机中文档的请求
    4) 浏览器和服务器之间的传输协议：HTTP（hypertext transfer protocol，超文本传输协议）
    5) URL：统一资源定位符。万维网中每一个文档都有唯一一个地址，作为超文本的链接
ii. URL命名规则
    1) 例子 http://sunting_comp.sustc.edu.cn/desktop/lbwnb.html 表示孙挺的这台电脑中桌面的一个叫卢本伟牛逼的HTML文件
    2) URL也可以不写文档名字，比如 http://sunting_comp.sustc.edu.cn/desktop 就只是表示孙挺电脑的桌面
    3) 另外，很多浏览器都默认用的是HTTP，所以开头不写HTTP也可以
       sunting_comp.sustc.edu.cn/desktop/lbwnb.html 也表示卢本伟牛逼这个文件
iii. 当超文本在网络中实现时，就形成了万维网（world wide web）。这主要是英国的Tim Berners-Lee 想出来的，他在1990年开发出第一个实现万维网的软件。
     2017年，他因“发明万维网、第一个浏览器和使万维网得以扩展的基本协议和算法”而获得2016年度的图

灵奖。

iv. HTML（hypertext makeup language，超文本标记语言）

1）本质：一个符号系统

2）用途：一个文档里包含了多媒体资源的话，怎么排版显示这些图片之类的就是新的问题，所以发明了HTML

3）它基于XHL（eXtensible markup language，可拓展标记语言）标准

a）但是由于它在XML标准巩固之前就开发出来了，所以它不完全遵守XML

b）HTML有个版本叫做XHTML，它完全遵守XML

c）可以通过XML开发出不同的标记语言，可以将它们组合起来获得复杂应用

v. 客户端（client-side）活动和服务器端（server-side）活动

1）就是客户端和服务器端提出一些要求，进行交流

2）控制客户端活动的方法

a）JavaScript（由网景公司开发）

b）先将网页传给浏览器，然后将小应用程序（applet，用Java编写）根据HTML源文档的要求传输给浏览器（SUN公司开发的）

c）用Flash展示多媒体（现归Adobe公司）

3）控制服务器端活动的办法

a）使用servlet（SUN公司开发）

b）JSP（Java server page，Java服务器页面）

c）ASP（active server page，活动服务器页面）

d）PHP（最初代表 personal　home　page（个人主页），现在指PHP hypertext processor（PHP超文本处理程序））

h. 因特网协议

i. 因特网软件的分层：应用层、传输层、网络层、链路层

ii. 传输过程：

1）应用层在报文上加上目的地址和端口号（port number），端口号用来确定目的地址中应用层的哪一个单元负责接收这个报文，端口号都是唯一的

2）然后传输层接过应用层加工好的报文，将报文拆成一个个独立的小片段，这些小片段被称为分组（packet）

拆分的原因

a）为了防止传输时堵塞，就像在铁路交叉口，一排排小汽车在等长长的火车通过那样。这些分组独立传输，可能会通过不同的路线到达目的地。

当传输层给这些分组加上片段序列号以后，就把它们交给了网络层

3）网络层决定报文下一个传输的中间地址是什么，然后就把报文交给链路层

4）链路层负责实际的传输，处理那些传输上的细节问题

a）如果网络是以太网，链路层就用CSMA/CD协议

b）如果网络是WiFi网，链路层就用CSMA/CA协议

5）传到了中转站以后，这个中转站的链路层将报文给了中转站的网络层

网络层通过转发表比对一番报文的目的地址：

a）如果自己就是目的地，则把报文交给交给了指定端口号上的应用层软件，然后等着其他报文都来了，就按照片段序列号拼接起来

b）如果自己不是目的地，就把报文还给链路层，并指定下一站。

c）为了提高速度，路由器中的链路层和网络层紧密相连，转发时间以微秒计

6）因为这些过程中，很多过程是瞬间完成的，所以因特网的整个响应时间以毫秒计

i. TCP/IP协议簇

i. 来源：

尽管ISO制定了一个OSI（open system interconnection，开放系统互连）作为开放式网络的参考模型，但是OSI标准是基于7层结构的，而4层结构在OSI之前就成为了因特网的事实标准，尽管没有人统一强推，但是大家都这样做

所以TCP/IP协议簇才是真正的标准

ii. 名字由来

1) TCP（transmission control protocol，传输控制协议）和IP（Internet protocol，网际协议）只是这些协议中的其中两个，所以不要因为UDP（user datagram protocol，用户数据报协议）和TCP不一样就认为UDP不属于这个协议簇

iii. UDP和TCP比较

1) 它们都是传输层的协议，是TCP/IP协议簇在传输层上的不同版本
2) TCP比较严谨，它比UDP多出的步骤有
   a) TCP发报文前，会向目的地的传输层发一个报文告诉目的地的传输层，让目的地的传输层确认接收。这就建立了一个连接。

      UDP因为没有这样的连接所以被称为无连接协议
   b) TCP会要求发送源和目的地的传输层通过 确认 和 分组重发 确保报文的所有片段都成功到达目的地

      因此，TCP被称为可靠的协议，而UDP被称为不可靠的协议
   c) TCP还有流量控制（flow control）和拥塞控制（congestion control）以防目的地应接不暇或者传输途中发拥塞
3) TCP：可靠但是慢，所以可以用于传输邮件
4) UDP：快但是不可靠，所以用于DNS查找和VoIP

iv. IP

1) 是网络层的标准
2) 网络层的任务：
   a) 转发（forwarding）：通过因特网传递分组
   b) 路由（routing）：维护更新转发表，以反映出条件的改变（比如一个路由器出问题不能用了，或者一个因特网区域太拥堵了需要避开）
3) 跳数（hop count）：也称作生存的时间（time to live）：发送源的网络层将这个跳数加到分组上，然后传递时路过的每一个网络层都会把这个跳数-1
   a) 意义：避免分组在系统中无休止地循环
   b) 初始值：一般64就够了
4) IPv4到IPv6
   a) IPv4规定的互联网地址是32位的，但是发现不够用了，于是IPv6给升级到128位了
   b) IPv6也能改进促进多路广播

j. 安全性

i. 恶意软件（malware）

1) 从内部攻击
   a) 病毒（virus）
      i) 蠕虫（worm）：主要是疯狂复制，然后搞得网络负载过大而崩溃
      ii) 特洛伊木马（Trojan horse）：伪装成游戏之类的实用安装包，诱骗用户下载，然后有时一完成下载就开始破坏用户电脑，有时会等特殊的日期之类的再开始破坏
   b) 间谍软件（spyware），也称为嗅探（sniffing）软件
      i) 收集所在的电脑的信息，然后发给攻击者。
      ii) 有时候一些商业公司可能会用嗅探软件来收集用户信息，建立目标用户档案
      iii) 电子黑饵（phishing）：跟钓鱼或老式电话诈骗差不多，广撒线。只是通常是利用电子邮件完成

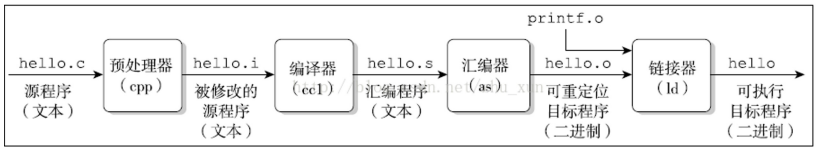
2) 从外部攻击
   a) DoS（denial of service，拒绝服务）
      i) 事先到网上抓一匹未设防的计算机作为帮凶，向其中植入攻击软件
      ii) 然后只要给出一个信号，这些计算机就会向目标计算机发出大量要求回复的信息，破坏这台计算机的工作。
      iii) 这种活动经常导致一些公司的商业活动被中断
   b) 发垃圾邮件（spam）
      i) 这不足以压倒计算机系统，但是足以压垮接收垃圾邮件的人

ii. 防护与对策
1) 防火墙（firewall）
   a) 安装在网络中某一重要节点上
   b) 当安装在网关处时
      i) 阻止向某一特定目的地址发送信息
      ii) 阻止接收已知的有问题的来源发送的信息（用来终止拒绝服务攻击）
      iii) 欺骗（spoofing）
         One. 阻止所有源地址为区域内地址的信息进入，因为这说明这些信息想假装自己是区域内的成员
      iv) 如果一台计算机不在用作服务器，那么它上面的防火墙应当阻止所有用于这些应用的通信，否则它上面的“漏洞”可能会被入侵者利用来建立联系，利用间谍软件建立一个秘密的服务器，从而获取嗅探结果
   c) 变种：垃圾邮件过滤器（spam filter）
      i) 可能要用到训练式的过程来判断，和概率论、人工智能之类的领域结合
   d) 代理服务器（proxy server）
      i) 作为客户机和服务器之间的媒介，屏蔽来自服务器的不利行为
      ii) 特点：实际的服务器没办法知道代理服务器是否是真的客户机
2) 审计软件
   a) 和操作系统里的审计软件差不多，用来察觉非正常行为
3) 防病毒软件（antivirus software）
   a) 用来杀毒，就是删除病毒和被病毒等感染的文件

iii. 加密
1) 许多因特网应用已经和加密技术进行结合，形成了这些应用的安全版本，比如：
   a) FTPS（FTP的安全版本）
   b) SSH（远程登入服务的安全版本）
   c) HTTPS（HTTP的安全版本）
      i) 很多时候用于金融机构
      ii) 核心是SSL（secure sockets layer，安全套接字层）
2) 公钥加密
   a) 一般用RSA算法（见离散数学）
   b) 公钥系统的小问题：确保公钥是正确的，而非冒名顶替者用来钓鱼骗取你的私钥的
   c) 解决方法：建立一些可信任网点，称为认证机构（certificate authority），用来记录网站的公钥，以证书（certificate）的形式来提供
   d) 鉴别（authentication）：用公钥来生成原文，称为数字签名（digital signature）。给对方私钥，让对方把你发的密文解一下。因为只有你才能产生这种密文，所以你是你

6. 算法
   a. 主要内容见数据结构与算法分析、算法设计与分析这两门课。这里做一些常识性内容的补充
   b. 定义：算法是一组可终止的、有序的、无歧义的、可执行的步骤的集合

c. 原语（primitive）：为了描述算法所用的严格定义的语言
d. 伪码（pseudocode）：用比较像程序语言的语言描述算法
e. 变量名命名规则
    i. 帕斯卡样式（Pascal casing）：每个单词的首字母都大写。
        1）名字来由：帕斯卡语言中比较流行这种样式
    ii. 驼峰样式（camel casing）：除了首字母，其他单词的首字母都大写
f. flowchart：流程图。就是高中学的那玩意，能够直观地体现算法思路过程

7. 程序设计语言
a. 建立在学过Java和C++的基础上进行补充
b. 分类：
    i. 机器语言就是一大串01编码
    ii. 将机器语言通过汇编器（assembler）用助记符系统表示就形成了汇编语言（assembly language）
        1）仍然要从机器角度考虑
        2）不可移植（要考虑新机器的寄存器配置和指令系统）
    iii. 第三代程序设计语言
        1）几乎机器无关（machine independent）。机器无关指不依赖特定的计算机特性
        2）执行方法：
            a）使用翻译器（translator），也称为编译器（compiler）
                i）特定：先把代码翻译成机器语言，再执行
            b）使用解释器（interpreter）：一边翻译一边运行，实际上就是根据程序的高级形式来使用
        3）翻译与解释
            a）Java
                i）把代码先编译成字节码文件，就是.class文件（这里的编译并不是像传统编译那样编译成机器语言），.class文件也叫做类文件，这仍然是平台无关的
                ii）然后再放到JVM里面逐条执行（用解释器），翻译成特定机器上的机器码
                iii）由于字节码并不针对任何机器，所以Java程序无需重新编译就可以在多种不同的计算机上执行
            b）JS早期将源码编译成语法树，然后直接执行语法树（后序遍历：先左再右最后根），但是直接解释语法树并不高效
            c） 

一个完整的编译系统与 一个用C编写的程序hello.c的编译过程
        4）语言拓展：一些常用语言通常有标准化版本，但是编译器的设计者可能会提供一些不在标准里的特性，一旦一个程序员用了这些拓展，他的代码就不能和其他编译器兼容了
c. 程序设计范型（programming paradigm）
    i. 命令型范型（imperative paradigm）
        1）FORTRAN：FORmula TRANslator。现在可以支持面向对象范型
        2）COBOL:Common Business-Oriented Language.据说很多银行的上古代码就是用Cobol写的，然后遇到千年虫问题了就GG
    ii. 函数式范型（functional paradigm）
        1）比如LISP在1960年就有了，1970年有了ML
        2） 有一个个函数完成操作
    iii. 说明性范型（declarative paradigm）
        1）描述问题即可，如：prolog。

2) 比较注重逻辑程序设计（logic programming）

iv. 面向对象范型（object-oriented paradigm）

1) C++、Java、Visual Basic、C#

d. 传统的设计概念

i. 脚本语言（scripting language）

1) 用来执行管理任务，表明要做什么，

2) 使用：PC用户也许会写一个脚本来指定一系列程序的执行，比如从数码相机中读取照片，通过日期对照片建立索引等

3) 历史：脚本语言可以追溯到上世纪60年代的作业控制语言，当时在批处理作业中它被用来指定操作系统

4) 不良影响：直到今天，很多人都认为脚本语言是指导其他语言的语言，这有局限性

5) 例子：

a) Perl和PHP在控制服务器Web应用和VBScript中很常见（VBScript是Visual Basic的非标准语言，由微软开发并用于Windows的特定环境下）

ii. Visual Basic

1) 面向对象。Windows用户可以通过它来开发自己的GUI应用程序

2) 不止是一种语言，还是一个完整的软件开发包，有各种预先定义的组件（如：按钮、复选框、文本框、滚动条等）

e. 翻译过程

i. 先用词法分析器分析一下字符流，整理出实体——标记（token）

1) 比如：153不应该被解释成1、5、3

2) 词法分析器会跳过所有的注释

ii. 再用语法分析器找出实体单元之间的逻辑关系和各种成分

1) 固定格式语言（fixed-format language）：为了简化语法分析，有的语言要求语句以特定的方式定位，比如python（利用缩进）

2) 一般都是自由格式语言（free-format language）

iii. 通用机器语言：

1) 来由：有的时候，比如动画网页，软件必须传到远程机器上执行

a) 如果发源码，到了目的地再翻译，那还要造成额外的延迟

b) 如果发机器码，那还要照顾到对方计算机的版本，要写不同的源码

c) 所以SUN和微软的Java和C#被翻译后得到的不是真实的机器语言，但是这种语言可以很迅速地被翻译成机器语言

2) 在Java中，这种可快速翻译的语言就是字节码（类文件，或者叫.class文件）

3) 在C#中，叫做.NET通用中间语言

4) 即时编译（just-in-time compilation，简称为JIT）：通用机器语言在执行前被快速翻译

iv. 语法分析时，我们可以通过grammar（文法）建立语法图（syntax diagram）或者语法分析树（parse tree）

v. 最后使用代码生成器生成机器语言的程序，并在此过程中进行汇编上的优化

f. 并行处理

i. 比如玩雷霆战机，很多飞船都要作为独立单元同时运动，我们同时激活（activation）多个战机程序，这就涉及到并行处理（parallel processing）或者并发处理（concurrent processing）了

ii. 这种并行，在Java里叫做线程（thread），类似于多任务操作系统里的进程

iii. 进程间通信：线程之间也要有通信。正在研究之中，几种处理方式都在操作系统那一节里提到了，比如设置通信量守护一片区域

8. 软件工程

a. 软件工程学科

i. 它和别的工程有很大本质上的不同，要注意
ii. 区别：
    1）利用构件来构建系统的能力弱
        a）一般构件都是用于特定领域的，很少有通用的构件，复杂的软件系统历来都是从头做起
    2）缺少度量技术——度量学（metrics）
        a）软件的复杂度的估算方法还不太成熟，评价软件质量的方法也不成熟
        b）机器有耗损，软件没有
iii. CASE（computer aided software engineering，计算机辅助软件工程）
    1）项目设计系统（用来辅助经费预算、项目调度、人员分配等）
    2）项目管理系统（用来辅助监控项目的开发进度）
    3）文档工具（用来辅助编写和组织文档）
    4）原型与仿真系统（用来辅助开发原型系统）
    5）界面设计系统（用来辅助GUI开发）
    6）IDE

b. 软件生命周期
    i. 维护软件的人员常常不是开发者，所以不得不研究底层的程序和它的文档，不然改动只会带来更多问题。很多时候，改着改着就觉得重写可能更快（通常是真的）
    ii. 在开发期间稍微努力，就可能导致修改时有极大的便利
    iii. 建房子时，地基建好了很多时候就不太可能做大修改，但是在软件工程中，需求不会因为软件的构建而停止

c. 软件工程方法
    i. 瀑布模型（waterfall model）
        1）单方向执行，事先完成全部分析，然后再开始执行
    ii. 增量模型（incremental model）
        1）先做出一个雏形，然后把具体实现一步步往上加
    iii. 迭代模型（iterative model）
        1）和增量模型很像，但是这个是不断改进优化，甚至重写
    iv. 增量模型和迭代模型都是原型开发（prototyping）
        1）增量模型中，是演化型原型开发（evolutionary prototyping）
        2）迭代模型中，是抛弃式原型开发（throwaway prototyping）
            a）快速原型开发（rapid prototyping）通常属于抛弃式原型开发
                i）迅速做出一个简单的模型，以便理清系统要有哪些功能、怎么和用户交互、各部分的关系、帮助销售期间向潜在的客户推销
        3）开放源码开发（open-source development）
            a）增量和迭代开发的一种变种方法
            b）例子：Linus Torvald 开发Linux系统
            c）描述过程
                i）一个开发者先把自己开发来满足自己需求的代码和相关说明文档公布
                ii）其他用户感觉不错，就拿过来改了改（纠错、优化、增强），以满足自己的需要
                iii）然后这些用户又把改动报告交给了原作者，原作者把这些改动整合到自己发布的软件中
                iv）实际上，有时候一个软件包在一个星期内就有可能经过好几次改动升级
    v. 敏捷方法（agile method）
        1）由瀑布模型转化过来，降低需求分析的严格性，响应需求变更
            a）代表例子
                i）极限编程（extreme programming，简称为XP）
                One. 少于12人的开发团队，在共同的办公场地自由交换想法，互相协作，每天不断地分析需求、设计、实现、测试，以增量的方式开发软件

d. 模块化（modularity）
    i. 耦合（coupling）
        1) 模块之间的联系。要尽可能小
        2) 是度量软件系统复杂度的指标
        3) 包括控制耦合和数据耦合
    ii. 内聚（）
        1) 模块内部的关联程度。要尽可能大
        2) 包括：
            a) 逻辑内聚（logical cohesion）
                i) 内聚度较低
                ii) 指一个模块中的某些活动其实看上去功能差不多
                iii) 例子：某个模块是用来负责系统与外界通信的，它里面有很多用来通信的功能，但是这些通信功能可能有各种各样的目标，比如有的用来获取数据，有的用来报告结果
            b) 功能内聚（functional cohesion）
                i) 内聚度较高
                ii) 整个模块都是用来实现每一个功能
e. 行业工具
    i. 数据流图（dataflow）
        1) 表示数据流向
    ii. 数据字典（data dictionary）
        1) 记录每一个数据项的各种属性
            a) 标识符
            b) 数据项里有效条目的构成情况（数据类型、取值范围）
            c) 数据项的存放地点，比如放在哪一个数据库里面
            d) 软件会在哪些情况引用这些数据项（哪些模块需要这个数据项的信息）
        2) 目标
            a) 加强潜在利益者和软件工程师之间的沟通
            b) 确保数据的数据类型是我们想要的
                i) 例子
                One. 有些数字不是数字类型的，而可能是字符串类型的，通过数据字典我们可以比较容易的发现这个问题
            c) 确保整个系统的一致性。借助数据字典经常可以发现冗余和矛盾
                i) 例子
                One. 一个数据在库存记录中称为PartNumber，而在销售记录中可能改称为PartId
                Two. 人事部门可能会用Name表示一个员工，而在库存记录中可能被用来表示一个零件
    iii. 统一建模语言（unified modeling language，UML）
        1) 基于面向对象范型发展而来的工具集
        2) 用例图（use case diagram）：表示某个人用到了哪些模块
        3) 类图（class diagram）：表示不同类型的项之间的逻辑关系，比如医生照看病人，外科医生继承了医生这个类
        4) 交互图（interaction diagram）：表示不同个体之间的通信，如：A打了一个球后请求judge（裁判）来判断这个球，然后裁判给B请求，让B把球打回去。在这个while循环结束后，score也要变化
f. 软件测试
    i. 帕累托法则（Pareto principle）：二八定律
        1) 运用：与其把所有的模块都进行相同的、不彻底的测试，不如对那些容易出问题的模块进行彻底的测试

ii. 基本路径测试（basis path testing）
    1）确保每一条指令都至少执行一次
iii. 白盒测试（glass-box testing）：测试人员在测试时需要用到软件的内部结构
iv. 黑盒测试（black-box testing）：站在用户的角度进行测试，不关心软件本身的工作原理
    1）边界值分析（boundary value analysis）：根据每一项数据的边界值来测试软件
    2）β测试（beta testing）：在版本还没有正式走向市场前，把它交给特定的一些用户使用
        a）beta版软件的存在会在市场上造成对软件的一种期待，可以增加推广和销量
    3）α测试（alpha testing）：在开发者那里进行测试

g. 文档编制
i. 用户文档（user documentation）
    1）软件中有一部分，官方网站上也可能有一部分，实体书里还可能有一部分
ii. 系统文档（system documentation）
    1）描述软件的内部构成，便于以后维护
    2）源码是其中的主要部分，要有大量注释
    3）还有设计文档的记录，包括了软件的需求规格说明和显示了这些需求是怎么实现的，这样就不会在维护时破坏软件的完整性
    4）大公司一般都有统一的软件编写规定，对缩进、命名等进行规定
iii. 技术文档（technical documentation）
    1）用来说明这个软件系统是怎么安装的，比如说调整参数、安装更新、反馈问题渠道
    2）在只有一个用户的PC上，技术文档和用户文档没什么区别，但是在多用户的环境中，区别就大了，因为这时候系统管理员负责所有软件的维护，而用户只管把软件当成一个抽象工具使用就好了

h. 人机界面
i. 从人的角度考虑各种GUI之类的东西的设计，运用人体工程学和心理学之类的学科进行设计
ii. 例子
    1）键盘为什么不按字母顺序设计呢?
        a）把热门字母错开，防止它们相邻，因为早期的打字机容易卡到
        b）放慢打字速度，因为手指不得不反复移动（毕竟热门字母都不相邻）
        c）所以有些键盘改变了这种字母排列顺序，据说可以让熟练的打字员提高68%的打字速度

i. 软件所有权和责任
i. 专利在这个时候并不靠谱，因为申请时间过长。等你申请到了，软件产品可能已经淘汰了

9. 数据库系统
a. 由于我已经学过了数据库系统这门课，所以以下内容为补充
b. 散列文件（hash file）
i. 通过位模式（键值）取模直接获得桶的位置（相当于一个元素是动态数组的数组）
ii. 模数是素数比较好，因为这样可以减少碰撞（collision，两个数散列以后获得同样的值）
iii. 如果很多数碰撞到一个桶里了，就称为群集（clustering），要不得
iv. 要规避群集，就要保证空桶多一些，我们使用负载因子（load factor）表示这个比例
v. 当负载因子保持在50%（要记录的数占总数的一半）以下，散列的性能就比较好，如果接近75%，那么通常大概要重建桶以扩容

c. 数据挖掘
i. 我们不要联机的数据库，而要静态的数据集合，称为数据仓库（data warehouse），这是数据库的快照，我们用数据仓库来分析
ii. 使用的技巧
    1）类型描述（class description）：找出描述一组数据项 的属性
        a）例子：发现购买二手车的人的特点
    2）类型识别（class discrimination）：找出区分两组数据项的属性

3) 聚类分析（cluster analysis）：比如发现某部电影的观众主要集中在4-10岁、25-40岁
4) 孤立点分析（outlier analysis）：找出不合群的数据项
   a) 用途：发现信用卡偏离了正常的消费模式，可能发生了盗用
5) 序列模式分析（sequential pattern analysis）：试图确定随时间变化的行为模式，比如预测股票的变化

iii. 数据库系统技术的发展和数据挖掘相辅相成
1) 因为数据库技术的广泛使用，数据仓库以多角度看待数据的能力，称为多维数据集（data cube，cube表示多维的意思）

10. 计算机图形学
a. 处理二维平面的包括2D图形学（2D graphic）和图像处理（image processing），前者侧重把二维的各种图像（比如○、矩形、文字）转化为像素模式，后者侧重于分析图像，模式识别，试图理解图像的内容
b. 3D图形学（3D graphic）
i. 负责把三维图形转化为图像
ii. 步骤
1) 建模（modeling）
   a) 在计算机图形学中，布景被称为场景（scene），道具被称为物体（object）
   b) 形状
      i) 3D中物体通常被描述成一些平面片（planar patch），这些多边形（一般是三角形）的平面片形成了多边形网格（polygonal mesh）
      ii) 方法1：用精确的解析几何方程表示，然后再用多边形网格近似。所以很多廉价的计算机动画中的人物角色经常用锥形、圆柱体之类的拼凑
      iii) 方法2：贝塞尔曲线（Bezier curve）和贝塞尔曲面（Bezier surface），就是用几个控制点表示曲线或者曲面在哪个地方弯曲了
      iv) 方法3：蛮力手工建模。用类似笔的设备触摸屏幕获得点集，然后合成
      v) 方法4：程序化模型（procedural model）。
         One. 运用算法产生出所需结构的单元。有时一些复杂的地形或云烟等难以用几何建模或者手工自动化建模，那就用程序化
         Two. 要运用分形几何，比如建立复杂的山脉就可以运用雪花曲线之类的思想
   c) 表面特征
      i) 纹理映射（texture mapping）
         One. 类似于贴墙纸
         Two. 在相对平坦的表面的效果比较好
   d) 寻求逼真的效果
      i) 这个要对每一种东西单独研究，从动力学、生理学等各个角度进行分析
      ii) 例子：布料、头发、羽毛、皮肤的建模
   e) 整个场景的建模
      i) 随着“相机”的移动，图像发生变化，场景中的各个物体都要改变
      ii) 也可以利用角度的改变来形成3D电影（同时获得左右眼的内容），这种3D电影一般要戴特别的眼镜，但是裸眼3D也在逐步发展
2) 渲染（rendering）
   a) 运用解析几何计算场景中的物体在投影平面（projection plane）上的透视投影（perspective projection）
   b) 从投影中心（center of projection）或叫做视点（view point）延伸出投影线（projector），形成的锥体称为视体（view volume）
   c) 光与环境的交互
      i) 要考虑环境光、反射、散射、折射

ii) 渲染流水线不考虑物体之间的相互影响，也没有阴影的问题
iii) 裁剪掉那些实体之外的东西后，进行扫描转换（scan conversion），或者叫做光栅化（rasterization），就是进行最后的成像
iv) 解决前景/背景问题
One. 画家算法（painter's algorithm）
First. 先扫描离“相机”最远的物体，然后一步步扫描更近的
Second. 缺点：不能处理那些纠缠的情况
Two. 单独考虑每一个像素点
d) 着色（shading）
i) 要考虑点表面的朝向从而确定镜面反射、散射、环境光
ii) 算法
One. 平面着色（flat shading）：对多边形网格着色
First. 缺点：看起来不平滑
Two. Gouraud 着色
Three. Phong 着色
Four. 凹凸映射（hump mapping）：增加额外的一个变量，本身表面的朝向纹理
e) 渲染-流水线硬件
i) 显卡/图形卡（graphic card）：与总线相连，专门解决渲染问题
ii) 通用的图像硬件接口OpenGL（open graphic library）、Direct3D
iii) 问题
One. 渲染-流水线只能实现局部照明模式（local lighting model），物体之间是独立的
Two. 但是要想全局的，复杂度太高，目前实时的技术做不到，所以游戏只能局部照明
f) 处理全局照明
i) 光线跟踪（ray tracing）：沿着到达视点的光线的反方向追踪，递归计算，如果追踪到了光源则结束，否则继续
One. 问题：只能追踪反射光，所以渲染出来的东西都好像是有光泽的
Two. 改进办法：使用分布式光线跟踪（distributed ray tracing）：同时跟踪从这点出发的多条光线，甚至是折射光，同时考虑光线吸收
g) 一种在建筑设计的图形软件中取代传统渲染-流水线的方法是辐射度（radiosity）。考虑平面片之间的辐射光能
3) 我们最终的成像就是图形窗口（image window）上的内容，就好比摄像机的屏幕
4) 最终的图像被放在帧缓冲区（frame buffer）

c. 动画
i. 先做出一个故事板（storyboard），显示出一些关键性的场景草图
ii. 如果是传统2D动画（如20世纪20年代的迪士尼工作室），动画大师做出一些关键帧以后，由助理动画师填补中间的帧，现在则可以自动生成中间帧，所以助理动画师不复存在
iii. 如果是3D动画，凭借故事板作为蓝图，构建出3D虚拟世界，然后不断地“拍摄”
iv. 为了增加动画的连贯性：
1) 超取样（supersampling）：将多幅稍微移动的图像重叠成单帧
2) 改变移动物体的形状使得它们仿佛是在沿一个方向延伸

d. 运动学（kinematic）和动力学（dynamics）
i. 比如《玩具总动员》中的牛仔脸部有将近100个关节以做出各种各样的表情口型
ii. 也可以自动生成动画，就像赋予动画人物意志一般，我们只是负责调参
iii. 动作捕捉（motion capture）：由实际的生物模型之类的指示各个关节怎么动
iv. 也可以由动画制作者创造性地给出

11. 人工智能

a. 研究路线
    i. 工程路线：面向性能，侧重于开发展示智能行为的系统
    ii. 理论路线：面向模拟，从计算角度研究人类的智能
b. 图灵测试（Turing test）
    i. 内容：一个人和一个测试对象通过打字机系统交流，如果人不能分辨出和他交流的是人还是机器，那么图灵测试就算成功
    ii. 图灵预言到2000年一台机器有30%的机会通过一个5分钟的图灵测试。预言准确。
    iii. 现在图灵测试不再被认为是对智能的度量，因为一个奇怪的智能显式可能用相对简单的方式产生，比如当机器不知道对方说的是什么时，可以回复说：“继续”、“这很有意思”之类的
c. 强弱人工智能
    i. 弱：对机器编程，然后机器展现出智能
    ii. 强：机器通过自己编程然后获得智能，就是意识到了那种推测能力
d. 理解图像：基于什么标准认定图像的内容？以及各种图像处理
e. 语言处理：句法分析、语义分析、上下文分析、信息检索、信息提取、语义网（知道car和automobile同义）
f. 推理
    i. 产生式系统（production system）
        1）状态集合：包含各种可能状态，比如起始状态、目标状态
        2）产生式集合：状态转移
        3）控制系统：选择哪一个下一个状态
        4）转换成了图论
        5）例子：专家系统
    ii. 搜索树（search tree）：从当前状态开始BFS，直到遇到目标
    iii. 启发式搜索（heuristic search）
        1）运用DFS，每次都选当前看起来最有希望的路：最佳适应算法
        2）要求：
            a）启发值要合理有用
            b）启发值计算不能太复杂
g. 学习
    i. 模仿（imitation）：模仿记录人类演示的行为
    ii. 监督学习（supervised training）：这一连串的实例称为训练集（training set），智能体（agent）对它们进行归纳，开发出一种适用于新案例的算法
    iii. 强化学习（reinforcement）：使智能体可以自己判断任务的成败，然后自我调参
    iv. 计算机系统甚至也可以发现一些新东西，比如Bacon系统就重新发现了欧姆定律、开普勒三大定律、动量守恒定律，一些用于天文学的系统甚至发现了新恒星
h. 遗传算法（genetic algorithm）
    i. 与生物中的遗传学术语对应关系
        1）染色体（chromosome）：一个试探解，未必能够成功
        2）基因（gene）：试探解中的每一步改变
        3）基因重组：先确定哪些染色体是当前最佳的父母候选人（这一步可能是这个算法里最困难的），然后后代都是父母随机基因重组的产物
        4）变异：后代可能出现一些随机的变异
    ii. 使用遗传算法的方法称为：进化规划（evolutionary programming）
i. 人工神经网络
    i. 描述：
        1）将处理单元视为一个神经元
        2）每一次都将所有输入的加权值与阈值进行比较

3) 如果大于阈值，则输出1，类似于神经元的兴奋状态，反之为0（抑制状态）

ii. 训练：调权值，可以使用遗传算法的技巧

iii. 联想记忆：就是有几个神经元被激活后，带动了好几个关联的神经元都被激活了，表示为应该精妙的网络

j. 机器人学

i. 史上最畅销的机器人：iRobot Roomba扫地机器人

1) 遇到障碍就会以简单的规则避开

ii. 涵盖了人工智能的所有研究范围

12. 计算理论

a. 图灵机（Turing machine）

i. 一个控制单元，对一条磁带进行读写，然后能做各种判断。其实就是一台状态机。

ii. 哥德尔早就在1931年就揭示了计算系统的局限性

iii. 在图灵提出图灵机的同一年，波斯特也提出了与图灵机等价的模型，现在称为波斯特产生式系统

b. 丘奇-图灵论题（Church-Turing thesis）：图灵机能算出来的问题和能算出来的问题等价，这些问题称为图灵可计算（Turing computable）问题

c. 一个不可计算的问题：停机问题（halting problem）

i. 问题：把一个函数本身的代码进行编码（比如用Unicode）转化为位模式，然后把这个位模式输入给这个函数，如果这个函数能在有限时间里运行出来，那么我们就称这个函数是自终止的（self-terminating）

ii. 反证法：这里有点像递归。

1) 假设存在这么一个函数H

2) 我们在H下面添加一段代码，while x!=0 {}，其中x=0表示这个函数是自终止的，形成了函数L

3) 然后请问L是不是自终止的呢?

a) 如果H回复是，那么x=0，L陷入死循环，说明L不是

b) 如果H回复不是，那么说明H已经计算出来了L不是，x=1，L结束了

c) dilemma。自相矛盾

4) 所以不存在H这样的函数

iii. 另外：请问人类大脑是否包含了比比执行算法过程更多的东西？如果没有，就说明人没有自由意志

d. NP问题的一种有趣的表述

i. 比如旅行商问题（traveling salesman problem）

ii. 一种非确定性解法表述：

1) 选一条可能的路径，然后算一下路径总距离是否小于要求。如果是，就宣布找到了；否则宣布没找到

2) 这个表述的第一步就不是确定的，所以我们称为非确定性问题

3) 王琦给出的表示是能在多项式时间内验证一个解的正确性，易知和这个表述等价