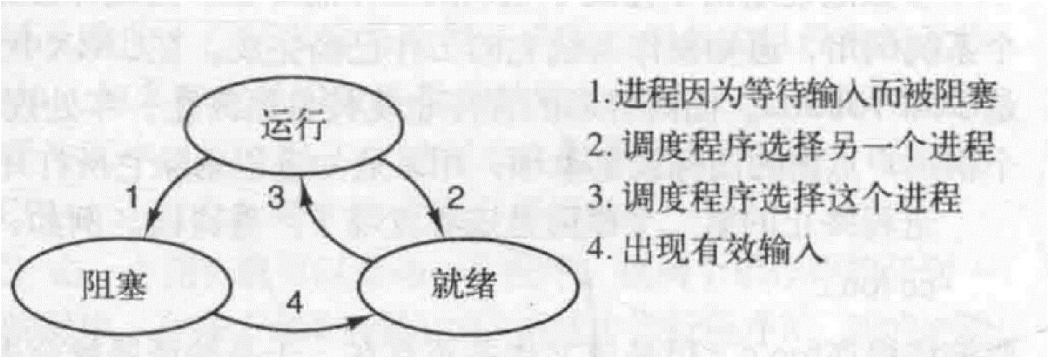
- **三个可能的状态**
  运行态：正在使用 CPU；
  就绪态：在等待队列，随时可运行；
  阻塞态：无法运行，直到某些条件满足，比如等待一个外部的输入。



- (1) 操作系统
  答：操作系统是运行在计算机硬件系统上的最基本的系统软件。它控制和管理着所有的系统　硬件（CPU、主存、各种硬件部件和外部设备等），也控制和管理着所有的系统软件（系统程序和用户进程等），操作系统为计算机用户提供了一种良好的操作环境，也为其他各种应用系统提供了最基本的支撑环境。
- (2) 脱机输入输出方式
  答：为了解决低速输入/输出设备和 CPU 速度不匹配的问题，可将用户程序和数据在外围机的控制下，预先从低速输入设备输入到磁带上，当 CPU 需要这些程序和数据时，再直接从磁带机高速输入到内存；或当程序运行完毕后 CPU 需要输出时，先高速地把结果输出到磁带上，然后在外围机地控制下，再把磁带上的计算结果由输出设备输出。这种输入/输出方式称为脱机输入输出方式。 采用这种方式大大加快了程序的输入/输出过程，提高了效率。
- (3) 进程
  答：进程是程序在一个数据集合上运行的过程，是系统进行资源分配和调度的一个独立单位。
- (4) 死锁
  答：当多个进程因竞争资源而造成的一种僵局，在无外力作用下，这些进程将永远不能继续向前推进，我们称这种现象为死锁。
- (5) 设备独立性
  答：设备独立性是指用户在编制程序时所使用的设备与实际使用的设备无关，即引入了逻辑设备和物理设备的概念。在用户程序中对 I/O 设备的请求采用逻辑设备名，而系统在实际执行时，则是通过逻辑设备表将设备名映射为物理设备名。
- **前台进程（Foreground processes）**:与用户交互并完成相应工作的进程
- **后台进程（Background processes）**：非前台进程，具有某些专门的功能。
- **守护进程（daemons）：**在后台处理各种请求服务的进程。
- **导致进程创建的事件**
  系统初始化
  由正在运行的进程创建
  用户请求创建
  批处理作业的初始化

- **导致进程终止的条件**
  正常退出(自愿):“Exit”/“ExitProcess”-Windows.
  出错退出（自愿)： 如: 输入文件不存在.
  严重错误（非自愿)：如：内存访问越界.
  被其他进程杀死（非自愿)：“Kill”/“TerminateProcess”-Windows.
- **伪并行（Pseudo- parallelism）**：进程间的快速切换给人以并行运行的假象。
- **操作系统维护一个进程表（Process Table)，其中的每一个表项（进程控制块，PCB）对应一个进程**
- **什么是线程?**
  线程是操作系统能够调度的最小单位;
  线程是进程的组成部分;
  每个进程至少包含一个线程;
  一个进程可以包含多个线程;
  线程间可以共享数据和内存。
- **为什么需要线程?**
  - ✓ **响应时间: 多个任务可以同时进行。**
  - ✓ **资源共享:线程间可以共享变量等资源。**
  - ✓ **经济: 线程更容易创建和销毁。**
  - ✓ **在多核架构计算机上有用。**

- **竞争条件（Race conditions）：多个进程访问一个共享数据，而数据最后的值由进程访问的先后顺序决定。**
- **临界区：对共享内存进行访问的程序片段。**
- **临界资源**：多道程序系统中存在许多进程，它们共享各种资源，然而有很多资源一次只能供一个进程使用。一次仅允许一个进程使用的资源称为临界资源。
- **好的竞争条件的解决方案**满足的四个条件

① 没有进程同时在临界区
② 不假设 CPU 的速度和数量;
③ 临界区外运行的进程不得阻塞其他进程;
④ 不会导致有进程永远在临界区外面等待。

- **管程**是一个由过程、变量及数据结构组成的集合。
- 进程可以随意调用管程中的过程，但是不能在管程之外声明的过程中访问管程内的数据结构。
- **忙等待**
  - 一个进程一直占用 CPU 测试循环的条件，直到测试条件为真。
  - 浪费 CPU;
  - 导致优先级反转问题

  如 H 进程优先级高，L 进程优先级低 ，假设 L 处于临界区中，H 这时转到就绪态想进入临界区，H 需要忙等待直到 L 退出临界区，但是 H 就绪时 L 不能调度，L 由于不能调度无法退出临界区，所以 H 永远等待下去。
- **从忙等待到阻塞...**
- 睡眠是一个系统调用，使得调用者阻塞自己直到另外的进程唤醒该进程。
- 唤醒调用唤醒指定的进程。
- **饥饿**：一个可运行的进程尽管能继续执行，但被调度器无限期地忽视，而不能被调

度执行的情况。

- **调度程序（Scheduler）：**用于决定哪个进程使用 CPU 的操作系统模块。
- **调度算法:**调度模块使用的算法。
- 为了防止一个进程运行太长时间，计算机有周期性的时钟中断激活调度程序（比如 20 毫秒）。
- **抢占式的调度:** 运行的进程被临时中断，使得其他进程有机会使用 CPU。
- **调度算法的目标：**
  公平性：每个进程获得相同的 CPU 使用时间。
  效率性：使 CPU 处于忙碌状态，做有用功。
  响应时间：最小化从发出命令到得到响应的时间。
  周转时间：最小化从提交任务到任务完成的时间。
  吞吐量：最大化每小时完成的任务数量。

- 死锁：
  一个程序集处于死锁状态当集合中的每个进程都在等待一个资源，而该资源又被集合中的另一个进程占有。
- 一个资源指的是一种能够被申请、被使用和被释放的对象。
- **可抢占资源**
  可以从拥有它的进程中抢占而不会产生任何副作用（e.g.内存)。
- **不可抢占资源**
  若从拥有它的进程中抢占将导致致命错误或相关的计算失败（e.g. 光盘刻录机)
- **死锁的条件**
  互斥：进程拥有的资源不可共享;
  占有和等待：一个进程拥有某个资源并请求新的资源;
  不可抢占：分配给进程的资源不可强制抢占，只能被拥有它的进程显示释放。
  环路等待：存在一条环路，环路中每个进程都在等待下一个进程所占有的资源。
- **处理死锁的策略**
  1、忽略该问题。例如鸵鸟算法，该算法可以应用在极少发生死锁的的情况下。为什么叫鸵鸟算法呢，因为传说中鸵鸟看到危险就把头埋在地底下，可能鸵鸟觉得看不到危险也就没危险了吧。跟掩耳盗铃有点像。
  2、检测死锁并且恢复。
  3、仔细地对资源进行动态分配，以避免死锁。
  4、通过破除死锁四个必要条件之一，来防止死锁产生。
- **四种用于存储管理的算法:**
  首次适配: 从头搜索直到找到一个足够大的空闲区。
  下次适配: 从上次搜索结束的位置开始搜索直到找到一个足够大的空闲区。
  最佳适配: 搜索整个链表以找到最小的足够大的空闲区。
  最差适配: 搜索整个链表以找到最大的空闲区。

- 问题：整个程序所需的内存远大于内存空间
- 解决方案:
  - ✓ 程序员把程序划分成多个片段，称为 **覆盖** – 但覆盖系统非常复杂,人工操作。
  - ✓ **虚拟内存** – 操作系统仅保留当前正使用内存的程序的一部分程序，全部工

作交给计算机去做。

- **分页** 实现虚拟内存的一种技术。
- **虚拟地址** 是一个由程序生成的地址
- **MMU** (内存管理单元) 把虚拟地址映射成物理地址
- 虚拟地址空间按照固定大小划分为(虚拟) **页面**。物理内存中对应的单元称为(页) **框。**
- 一个"在/不在"位记录页面在内存中的实际存在情况。
- 调用一个没有被映射的页面会使 CPU 陷入到操作系统。
- 这个陷阱称为**缺页中断（或缺页错误）。**MMU 找到一个很少使用的页框且把它的内容写入磁盘（如果它不在磁盘上）。随后把需要访问的页面读到刚回收的页框中，修改映射关系，然后重新启动引起陷阱的指令。

- 大部分的程序都会访问一小部分的页
- 提高方案: 给计算机装配一个小的硬件设备，称为 **转换检测缓冲区** (**TLB**) 或 **相联存储器**，以无需通过页表即可把虚拟地址映射成物理地址。

- 通常地，每个进程都会关联一个页表。这种方法的其中一个缺点是每个页表都可能含有数以百万计的表项。
- **倒排页表**被用于解决这个问题。每个物理实际页框对应一个表项。
- 每一个表项由存在于真实内存的页的虚拟地址，以及占有这个页的进程的相关信息组成。

- 页面在请求的时候才调入—**请求调页**。
- 在进程运行的任何阶段，它都只访问较少的一部分页面，这种现象称为**局部性访问**。
- 进程最近一段时间访问的页面集合称为**工作集**。
- 若进程每执行几条指令就发生一次缺页中断，这种现象称为**颠簸**。
- 在进程运行之前，系统保持进程的工作集在内存中，这种方法称为**工作集模型**。
- 全局分配策略对所有可运行的进程动态分配页框。局部分配策略只对单个进程分配页框。
- **段是逻辑上独立的地址空间。**
  - ✓ 段可能有不同的大小
  - ✓ 它们的大小可能会动态变化
  - ✓ 地址空间使用二维内存地址，由两部分组成: (段 #，段的偏移量)
  - ✓ 段可能有不同的保护
  - ✓ 允许进程之间共享过程和数据
- 把每个物理块中的指针集中记录到一个表，放在内存中索引，该表称为文件分配表 **FAT** (**File Allocation Table**) 。
- 备份: 从灾难中恢复、挽救错误的操作;
- 备份的一些考虑
  - ✓ 备份整个文件系统;
  - ✓ **增量转储**: 每次只存储修改过的文件;
  - ✓ 先压缩再存储;
  - ✓ 对活动文件系统做备份;
  - ✓ 安全问题;
- 两种备份策略:

- ✓ **物理转储**：从第 0 块开始，将全部块复制存储；

优点：简单、快速；

缺点：无法增量存储，不能满足特定文件的恢复请求；

- ✓ **逻辑转储**：从一个或几个指定的目录开始，递归地转储自某个基准日期之后更改的文件和目录；

- **数字签名：**一种类似写在纸上的普通的物理签名，但是使用了公钥加密领域的技术实现，用于鉴别数字信息的方法。
- **安全性的目标与威胁**

  数据机密性：数据暴露

  数据完整性：数据被篡改

  系统可用性：系统瘫痪或拒绝服务

  排外性：系统被病毒控制

- **程序控制 I/O** (**PIO**) or Polling：控制 CPU 和外围设备之间数据传送的一种方法。数据传送的工作主要由 CPU 完成。
- **中断驱动 I/O**

  将 CPU 从繁忙等待中解脱出来

  （1）CPU 初始化 IO 并启动第一次 IO 操作；

  （2）CPU 去忙别的活；

  （3）当 IO 完成时，CPU 将被中断；

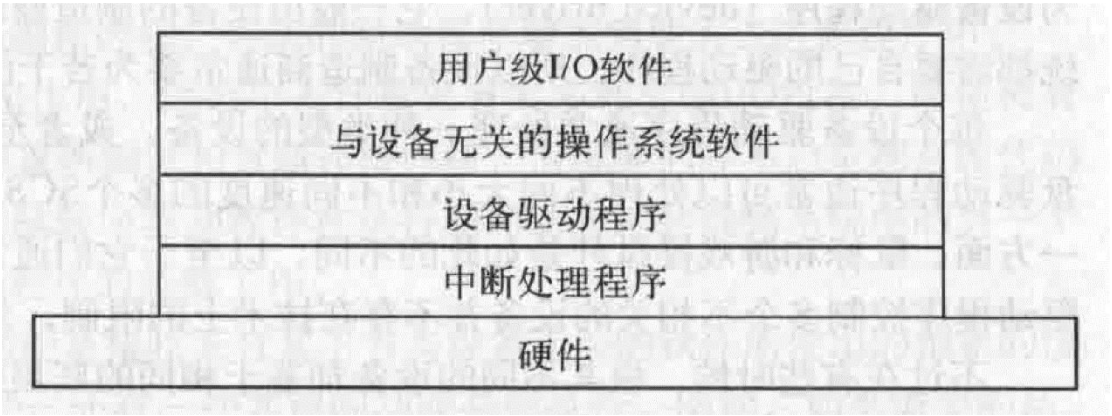
  （4）CPU 处理中断；

  （5）CPU 恢复被中断的程序；
- **使用 DMA 的 I/O**

  利用 DMA 打印字符串的过程

  (a) 程序调用系统调用打印字符串

  (b) 中断服务过程
- **I/O 软件层次**



1. 什么是多道程序设计？为何要引入多道程序设计？

答：多道程序设计是指同时把多个作业（程序）放入内存并允许它们交替执行和共享系统中的各类资源；当一道程序因某种原因（如 I/O 请求）而暂停执行时，CPU 立即转去执行另一道程序。操作系统在引入多道程序设计技术后，使得系统具有了多道、宏观上并行、微观上串行的特点。

引入多道程序设计是为了减少 CPU 时间的浪费，增加系统吞吐量，提高系统效率。

2. 操作系统的设计目标有哪些？操作系统的特性是什么？

答：目标——（1）提供一个计算机用户与计算机硬件系统之间的接口，使计算机系统更易使用；（2）有效控制和管理计算机系统中的各种硬件和软件资源，使之得到更有效的利用；（3）合理地组织计算机系统地工作流程，以改善系统性能。

特性——（1）并发性　两个或两个以上事件在同一时间间隔内发生。(2）共享性　指系统中地硬件和软件资源不再为某个程序所独占，而是提供多个用户共同使用。（3）虚拟性指把一个物力上的实体变为若干个逻辑上的对应物，前者是实际存在的，后者是虚的，只是给用户的一种感觉。(4)　不确定性　有两种含义:一是 程序执行结果是不确定的 二是 多道程序环境下程序的执行是以异步方式进行的,即程序的执行时间和多道程序的执行顺序是不确定的。

3. 何谓线程？试述虚拟处理机的概念。

答：线程是比进程更小的能够独立运行的基本单位。它的引入有效地提高了系统内程序并发执行的的程度，也进一步提高了系统的吞吐量。

虚拟处理机，是采用多道程序设计技术，使得计算机可以同时处理多个作业，使用户感觉到每一个作业在一个独立的 CPU 上运行，这个 CPU 是模拟出来的，称之为虚拟处理机。

4. 何谓临界区？给出临界区的使用准则。

答:进程在并发执行中可以共享系统中的资源，但对临界资源的访问必须互斥进行。我们把一个进程访问临界资源的那段代码称为临界区。临界区使用准则如下:

(1) 空闲让进——无进程处于临界区时，若由进程要求进入临界区应立即允许进入。

(2)忙则等待——当已有进程进入临界区时,其他试图进入各自临界区的进程必须等待,以保证　　诸进程互斥地进入临界区。

(3)有限等待——有若干进程要求进入临界区时,应咱有限时间内使一进程进入临界区,即它们不应相互等待而谁都不进入临界区。

(4)让权等待——对于等待进入临界区地进程必须释放其占有地 CPU。

5．何谓虚拟存储器？有何特征？

答：基于程序局部性原理，一个作业在运行之前没有必要全部装入内存，而仅将当前要运行地那部分页面或段先装入内存就可以启动运行，其余部分则存放在外存。当所访问地信息不在内存时，再由系统将所需要地那部分内容调入内存。从效果上看，计算机系统好像为用户提供了一个比实际内存大得多地存储器。这个存储器称为虚拟存储器。特点有四个

离散性：在内存分配时采用离散分配方式。

多次性：一个作业运行时分成多次装入内存。

对换性：作业在运行时可以将需要的内容调入内存，也可以将内存中暂时不需要的程序或数据调至外存。

虚拟性:从逻辑上扩充了内存容量,使用户感觉到的存储容量远远大于实际的内存容量。