· 死锁产生的必要条件：

简答题例 1： 简述死锁产生的原因及必要条件。

答：死锁是指多个进程因竞争资源而造成的一种僵局，若无外力作用，这些进程将永远不能再向前推进。

· 产生死锁的原因可归结为两点：

（1） 争资源。

（2） 进程推进顺序非法。

· 在具备下述四个必要条件时，就会产生死锁。

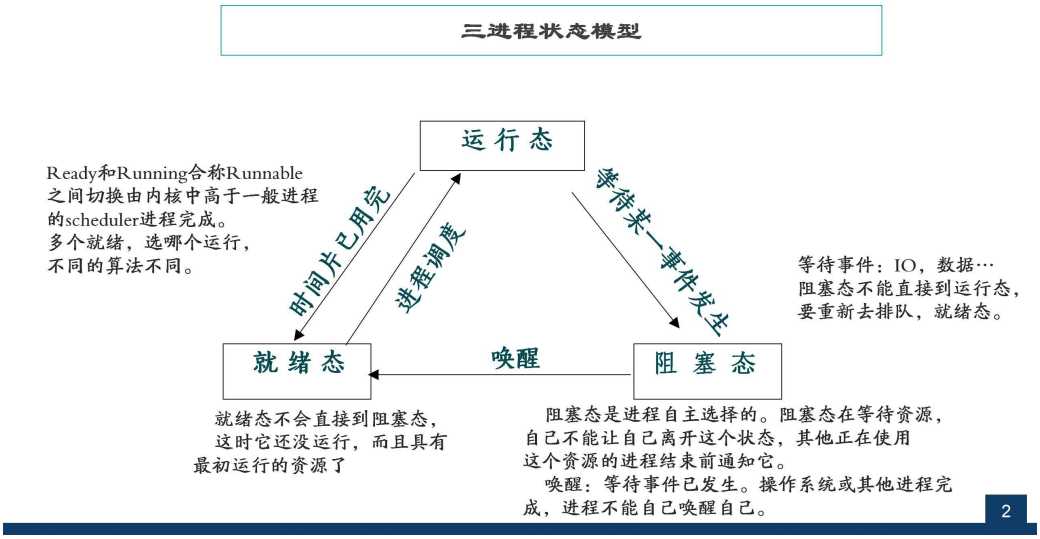
（1） 互斥条件

（2） 请求和保持条件

（3） 不剥夺条件

（4） 环路等待条件

· 进程状态变化及原因：



· 尹玲：重点，注意每个状态的意思，状态切换有哪些，切换由何完成

就绪态 → 运行态：当处理机空闲时，进程调度程序必将处理机分配给一个处于就绪态的进程 ，该进程便由就绪态转换为运行态。

运行态 → 阻塞态：处于运行态的进程在运行过程中需要等待某一事件发生后（例如因I/O请求等待I/O完成后），才能继续运行，则该进程放弃处理机，从运行态转换为阻塞态。

阻塞态 → 就绪态：处于阻塞态的进程，若其等待的事件已经发生，于是进程由阻塞态转换为就绪态。

运行态 → 就绪态：处于运行状态的进程在其运行过程中，因分给它的处理机时间片已用完，而不得不让出（被抢占）处理机，于是进程由运行态转换为就绪态。

而【阻塞态 → 运行态】 和 【就绪态 → 阻塞态】这二种状态转换不可能发生。

- 页面置换算法：（B 站看课）

- SPOOLing：

例题：简述 SPOOLing 技术的基本思想和 SPOOLing 系统的组成。

答: SPOOLing 技术基本思想是利用大容量高速磁盘作为暂存数据空间，通过程序模拟脱机 I/O 技术，以实现联机情况下的高效 I/O 工作。（1 分）

SPOOLing 系统的组成：输入/输出缓冲（1 分）；输入井/输出井（1 分）；输入进程/输出进程（1 分）

·分区分配算法：

给定存储器的划分，依次为 100KB、450KB、250KB、300KB、和 600KB，现有 4 个进程分别依次为：212KB、417KB、112KB、426KB.为了在给定的存储空间中安置进程，现有三种算法：首次适应算法、最佳适应算法和下次适应算法。在这三种算法中，哪一种算法更能充分利用存储空间。

解：

首次适应算法：无法满足，212K 进程放入 450K 的空闲区中，还剩余 238K；417K 的放入 600k 的空闲区中；112K 放入 238 中，426k 的进程无法满足

最佳适应算法：可以满足，212K 放入 250K 空闲区中；417K 放入 450K 空闲区中，112K 放入 300K 空闲区中，426K 放入 600K 空闲区中。

下次适应算法：无法满足，212 放入 600K 空闲区中，剩余 388K；417K 放入 450K 空闲区中，最后导致 426K 的进程请求无法满足。所以，最佳适应算法利用内存最充分。

·补充：

·首次适应算法（First Fit）：该算法从空闲分区链首开始查找，直至找到一个能满足其大小要求的空闲分区为止。然后再按照作业的大小，从该分区中划出一块内存分配给请求者，余下的空闲分区仍留在空闲分区链中。

特点：该算法倾向于使用内存中低地址部分的空闲区，在高地址部分的空闲区很少被利用，从而保留了高地址部分的大空闲区。显然为以后到达的大作业分配大的内存空间创造了条件。

缺点：低地址部分不断被划分，留下许多难以利用、很小的空闲区，而每次查找又都从低地址部分开始，会增加查找的开销。

·最佳适应算法（Best Fit）：该算法总是把既能满足要求，又是最小的空闲分区分配给作业。为了加速查找，该算法要求将所有的空闲区按其大小排序后，以递增顺序形成一个空白链。这样每次找到的第一个满足要求的空闲区，必然是最优的。孤立地看，该算法似乎是最优的，但事实上并不一定。因为每次分配后剩余的空间一定是最小的，在存储器中将留下许多难以利用的小空闲区。同时每次分配后必须重新排序，这也带来了一定的开销。

特点：每次分配给文件的都是最合适该文件大小的分区。

缺点：内存中留下许多难以利用的小的空闲区。

·最坏适应算法（Worst Fit）:最坏适应算法是将输入的作业放置到主存中与它所需大小差距最大的空闲区中。空闲区大小由大到小排序。

特点：尽可能地利用存储器中大的空闲区。

缺点：绝大多数时候都会造成资源的严重浪费甚至是完全无法实现分配。

关于三种放置策略的讨论：首次适应算法、最佳适应算法、最坏适应算法的队列结构。