**2019级期末考试真题**

\* 简答题（4个，一个10分）

  \* 1.进程与线程

\* xxxx段可以共享，栈段不能共享，原因

线程共享进程的代码段、数据段、堆段等，但是不共享进程的栈、寄存器等资源。这是因为栈和寄存器是程序执行动态上下文，它们是随着程序执行不断发生变化，如果允许被共享，则程序执行的上下文会被破坏，程序无法正常执行。

\* 举例什么能导致进程由运行态 => 就绪态

1.进程执行yield()，主动让出CPU。

2.有优先级更高的进程抢先执行。

  \* 2.核心IO子系统、设备驱动程序、中断处理程序分别有什么功能？他们有什么关系？

核心IO子系统：用户可以利用内核提供统一的系统调用格式访问不同的设备，核心IO子系统实现有关设备共性的操作，对于不同的设备，I/O子系统为内核提供统一的访问控制方式。

设备驱动程序：屏蔽了具体设备的细节，为I/O子系统提供统一的访问控制方式（访问接口）。

中断处理程序：用于保护被中断进程的CPU环境，转入相应的中断处理程序进行处理，处理完成后再恢复到被中断进程的现场后返回到被中断的进程。

设备驱动程序为内核I/O子系统提供统一的访问接口，屏蔽了具体设备的细节，而设备驱动程序驱动I/O设备时需要用到中断处理程序。

  \* 3.死锁

\* 解释死锁概念

死锁是指一组处于等待（阻塞）状态的进程，每一个进程持有其它进程所需要的资源，而又等待使用其它进程所拥有的资源，致使这组进程互相等待，均无法向前推进。

\* 死锁处理措施

如果出现了死锁，可采用以下几种方式：

1. 撤销一个或多个处于死锁状态的进程。
2. 将某些进程回滚，相当于从这些进程中抢占资源，依靠checkpoint实现。

\* 现代操作系统一般怎么处理死锁

死锁的预防

死锁的避免

死锁的检测与恢复

  \* 4.文件的磁盘块分配算法

\* 分配方式有哪些

连续分配方式

链接分配方式

索引分配方式

    \* 这些分配方式应用于实际操作系统时有什么改进？改进有什么意义？

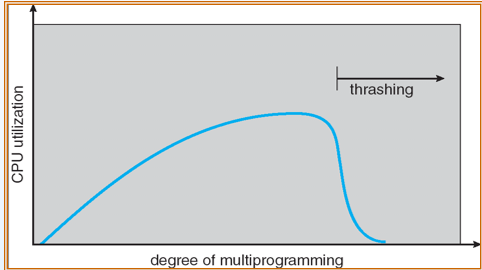
连续分配方式：以一个扩展为单位分配磁盘空间，而不是以磁盘块为分配单位。

链接分配方式：将隐式链接中的Linked域独立出来，构成一个FAT表，专门用于实现磁盘块之间的链接，可以支持文件的随机存取。

索引分配方式：采用混合索引分配，既能存储大文件，又能快速访问小文件。

\* 计算题（35分，15+10+10）

  \* 5.给一个场景（）

（这幅图是我从PPT上截的，不是考试原图）

\* 这是发生了什么问题？（其实是抖动）

\* 给出解决这个问题的两个方法，并说出他们相应的思想。

1. 工作集

根据局部性原理，每个进程最近使用的页框数作为将要使用的页框数的近似值；根据在一个时间窗口内每个进程所访问的页面踪迹，得出该时间段内每个进程所访问的页面集合，即每个进程的工作集；

1. 缺页故障频率策略

先确定可接受的缺页率，如果真实的缺页率太低了，进程将释放一些页，如果真实的缺页率太高了，进程将获得一些页。

  \* 6.进程调度（优先级调度）

    \* 画甘特图，计算平均等待时间

\* 优先级调度算法有什么问题？怎么改进？

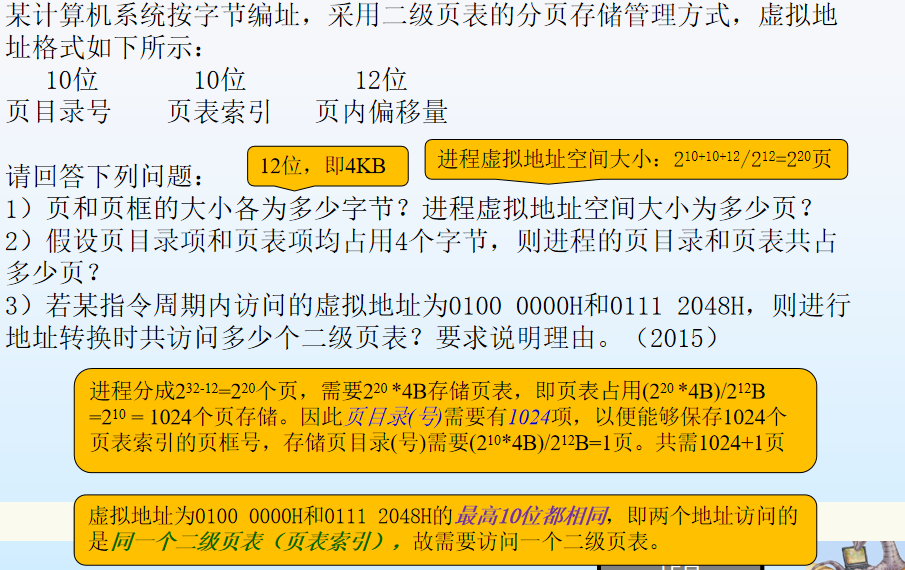
优先级调度算法存在饥饿问题，可采用高响应比算法，避免饥饿。

\* 7.两级页表（考研原题）10分

    \* 给虚拟地址，计算页/页框大小

    \* 求一级页表和二级页表共占了多少页

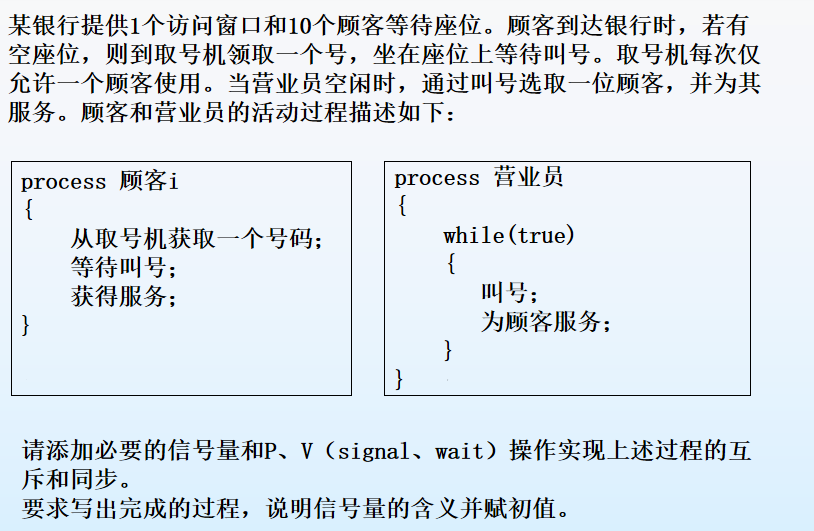
    \* 给你两个16进制虚拟地址，问你访问这两个地址的过程中访问了多少个二级页表？（1个）

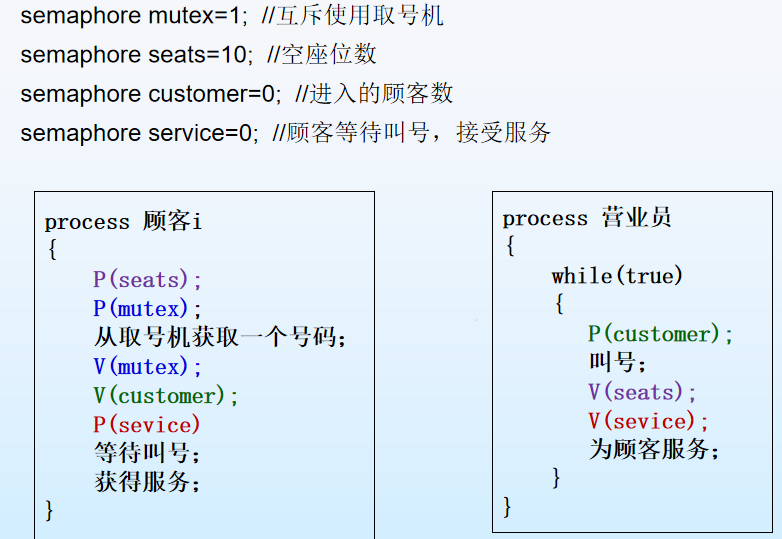


\* 算法编程题（25分，10+15）

  \* 8.银行顾客座位(银行提供1个访问窗口和10个顾客等待座位。顾客到达银行时，若有空座位，则到取号机领取一个号，坐在座位上等待叫号。取号机每次仅允许一个顾客使用。当营业员空闲时，通过叫号选取一位顾客，并为其服务。顾客和营业员的活动过程描述如下)

    \* 设置相应的信号量并初始化，并解释信号量的含义，并通过PV操作（或wait/signal操作）写出整个流程





  \* 9.用linux设计一个类似的shell程序。有三个命令，分别说明命令的含义及实现思路，并给出核心代码的C语言实现

    \* ./a.out&（父进程即shell不wait这个子进程 => 就是装入这条命令的子进程 了，而是由管理进程来wait）

      \* ```c

        if(pid == 0) {

          // 子进程

          execlp(a.out, ...);

        } else {

          // 父进程 shell

          printf("命令提示符\n");

        }

        ```

    \* ./a.out > t.txt（标准输出重定向，把a.out的标准输出重定向到t.txt中）

      \* ```c

        if(pid == 0) {

          // 子进程在装入新的进程前要完成重定向

          close(1);

          open(t.txt, CREATE | WRITE);

          execlp(a.out, ...);

        } else {

          waitpid(pid);

          printf("命令提示符\n");

        }

        ```

    \* ./a.out | b.out | c.out（多级管道，a的输出传到b的输入，b的输出传到c的输入。abc之间是拓扑同步的，通过管道来传递数据，其实也是重定向标准输入输出来实现的，对a来说重定向了输出，对b重定向了输入和输出，对c重定向了输入)

      \* ```c

        int pipeAtoB[2] = pipe();

        int pipeBtoC[2] = pipe();

        // 下面以创建a和b间的管道为例

        pid\_t pidA = fork();

        if(pidA == 0) {

          // a进程

          // pipeAtoB[0]为读，pipeAtoB[1]为写

          close(pipeAtoB[0]);

          execlp(a.out, ...);

        } else {

          // 父进程

          pid\_t pidB = fork();

          if(pidB == 0) {

            // b进程

            close(pipeAtoB[1]);

            close(pipeBtoC[0])

**2020级期末考试真题**

简答题

1、

（1）用户态，核心态是什么

用户态和核心态是CPU运行的两种状态。

1. 这种区分对现代操作系统的意义

在用户态，CPU不能执行特权指令，在核心模式中，CPU可以执行其所有指令，包括特权指令。而特权指令则是指可能对操作系统或其它程序引起损害的指令，这样区分能保证系统的稳定性和安全性，也能防止用户影响其他进程的运行。

1. printf(“str”) 是否进入核心态，为什么（是）

会，因为printf函数实际上是调用了write系统调用，而用户提出系统调用后会陷入核心态，让操作系统代为执行。

（4）double x = cos(13) 是否进入核心态，为什么（否）

不会，因为cos(13)不涉及到系统调用，所以也不会进入核心态。

2、

1. 抖动的定义

抖动：系统在页面置换上所花费的时间多于进程的运行时间，导致CPU的利用率急剧下降。

1. 如何利用工作集算法解决抖动问题

根据局部性原理，每个进程最近使用的页框数作为将要使用的页框数的近似值。在一个时间窗口内每个进程所访问的页面踪迹，得出该时间段内每个进程所访问的页面集合，即每个进程的工作集,将该工作集的大小作为每个进程需求内存的下限。如果所有进程需要的页框数之和大于系统可用的页框数，需挂起或终止部分进程，以释放内存，将它们分配给其它的进程。

3、

基于请求页式虚拟内存实现，回答下面问题

1. 虚拟存储器的思想

虚拟内存指的是仅把程序的一部分装入内存便可运行作业，具有请求调入功能和置换功能，能从逻辑上对内存容量加以扩充的一种存储器系统。

1. 页表中有哪些页表项，对应的作用是什么

有效位和物理块号。物理块号的作用是表明页表项当前页号对应的物理块是哪一块，用于逻辑地址转化为物理地址。有效位则是标识页表项是否有效，如果有效则可以进行地址转化，如果无效可发出缺页错误，让OS进一步判断是地址越界还是不在内存中。

1. 如何防止进程访问不属于自己的地址空间

如果进程访问到了不属于自己的地址空间，操作系统会产生地址越界中断。

4、

1. 驱动的功能

设备驱动程序：屏蔽了具体设备的细节，为I/O子系统提供统一的访问控制方式（访问接口）。

（2）操作系统“一切皆文件”思想的优缺点

优点：

1、统一的文件操作接口和编程风格

2、方便的设备管理，易于扩展和定制

3、简化权限管理，提高安全性

4、操作方式统一，便于实现设计

缺点

统一的访问方式无法使用有些设备的特性，这些特性可能会提高设备的性能。

大题

1、

1. 一个例子，应用银行家算法

银行家算法：

合法性检查：即判断Request向量是否小于Need向量。

可用性检查：即判断Request向量是否小于Available向量。

假分配，先假设分配。

然后进行安全性检查，如果能通过则分配资源，不能通过则让进程等待。

（2）死锁的必要条件，银行家算法破坏了哪一个（些）条件（破坏了循环等待）

2、

基于索引分配文件块算法的文件系统

char c;

int fp = open("file.txt", flag);

c = read(fd, &c, sizeof(c));

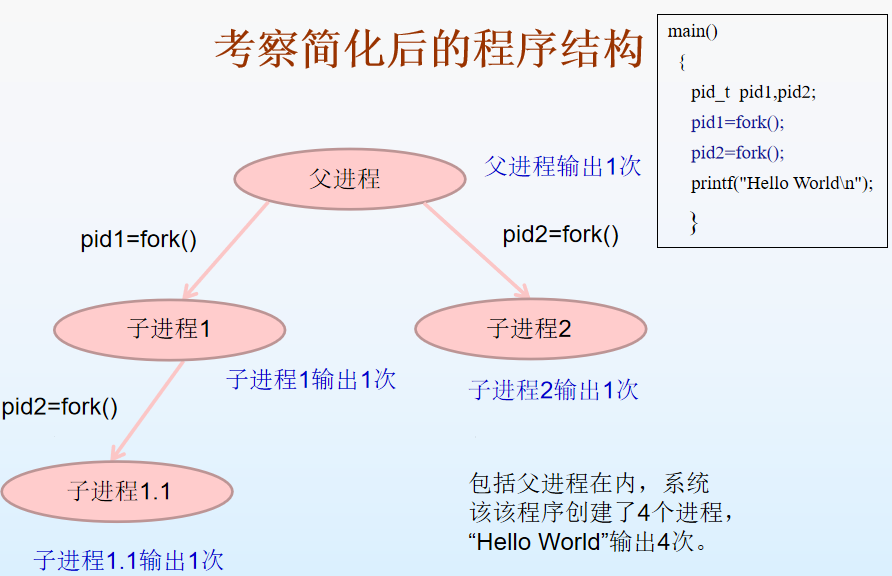
1. 从open打开文件，到变量c获得数据，操作系统做了什么？



1. 上述过程用到了文件系统的哪些数据结构

文件目录表，全局打开文件表，FCB，进程局部打开文件表（进程用户标识符表）

1. 结合进程树，说明代码会输出几个“Hello World”



4、（2016年考研真题，题干为某进程调度程序采用基于......）

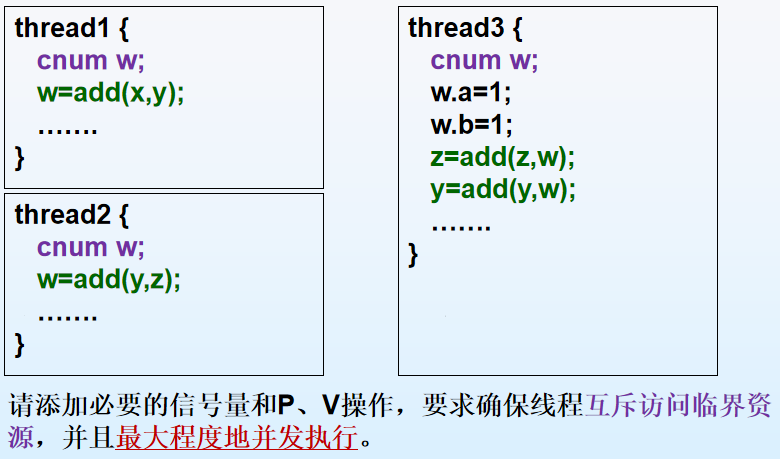
1. 为什么采用固定的优先级调度策略会引起进程饥饿

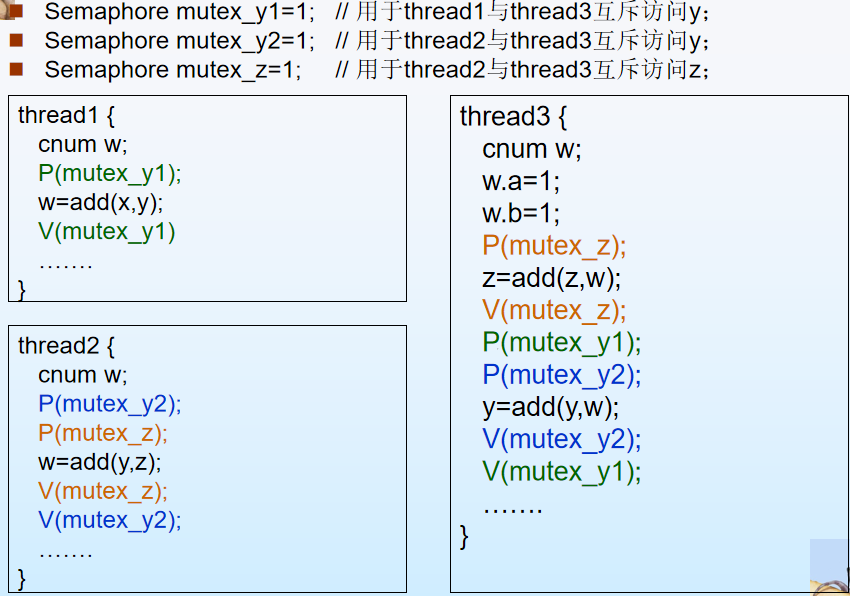
如果优先级固定的话，优先级低的进程可能一直得不到响应，就会导致饥饿。

1. 根据cpuTime和waitTime设计动态优先级算法 waitTime的作用 (当运行态时cpuTime按时间加1，waitTime置0，就绪态时cpuTime置0，waitTime按时间加1)

设计一个类似高响应比算法即可。

5、





**2021级期末考试重点(非真题)**

简答六个

1. 线程和进程

进程本质上是一个运行中的程序，是一个程序执行的实例。

线程是进程的运行实体，一个进程至少需要一个线程，可以拥有多个线程，这些线程运行在所属进程的上下文中，线程不能脱离进程而独立存在，线程可以看成是一个特殊的进程。

进程是分配资源的基本单位，而线程是调度的基本单位。

1. 程序的结构（内在）

程序的结构包括：代码段、数据段、堆区、栈区、全局变量区。

代码区：存放程序的可执行指令，所有的执行都在代码段发生。

常量区：程序中的常量数据，全局的const变量也在此处。

堆区：存放程序的堆，也即动态分配内存时内存的来源。

栈区：存放程序的运行栈，包含临时数据，如局部变量、函数参数值、返回地址等，以供过程调用时保存和恢复上下文。

全局数据区：存放静态数据，比如：静态局部变量、静态全局变量、全局变量等。

初始化的全局变量和静态变量在一块，未初始化的全局变量和未初始化的静态变量在一块。

1. 一切皆文件（文件思想）

操作系统将所有外部设备统一视作文件，用户可以像操作文件一样使用这些设备，无需了解其具体存在形式。

Linux系统将所有硬件设备、目录、进程和网络连接等都当作文件处理，统一了操作方法，使得Linux系统有很高的灵活性和可扩展性。

优点：

1、统一的文件操作接口和编程风格

2、方便的设备管理，易于扩展和定制

3、简化权限管理，提高安全性

4、操作方式统一，便于实现设计

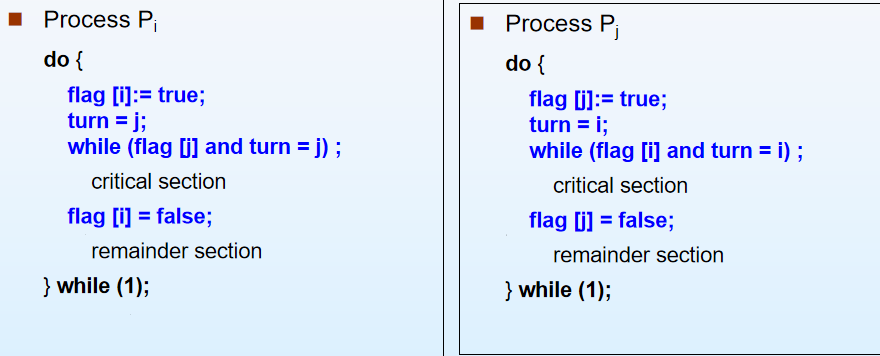
缺点

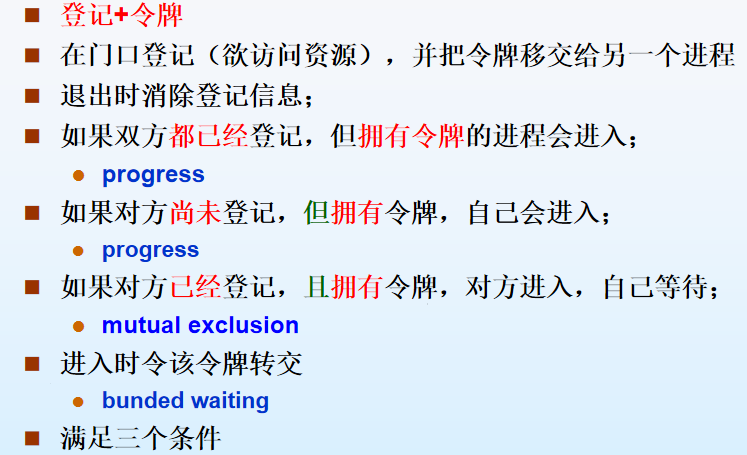
统一的访问方式无法使用有些设备的特性，这些特性可能会提高设备的性能。

竞争条件（race condition）：多个进程并发访问和操作同一数据并且执行结果与访问发生的特定顺序有关，称为竞争条件。

4.互斥同步四种方法 访问临界资源 伪代码

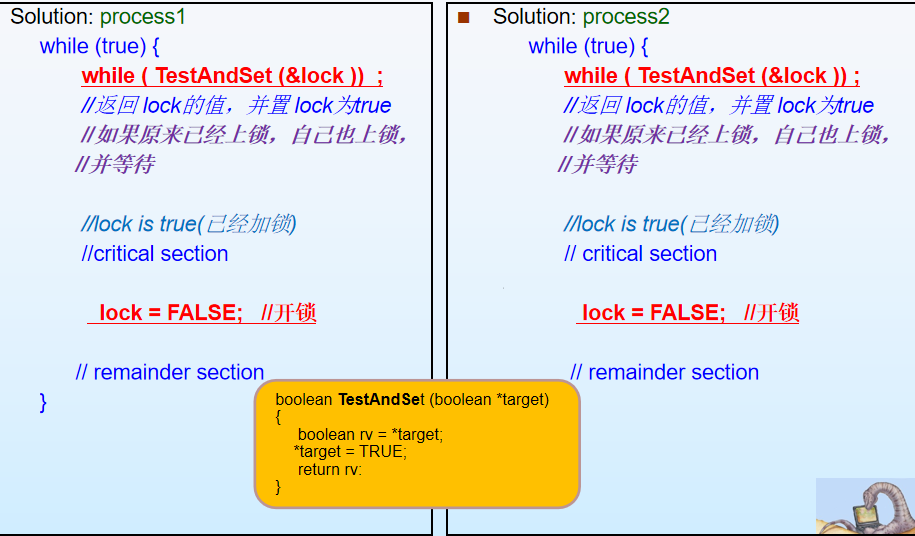
（1）软件实现（Peterson）：设置flag[]和turn。



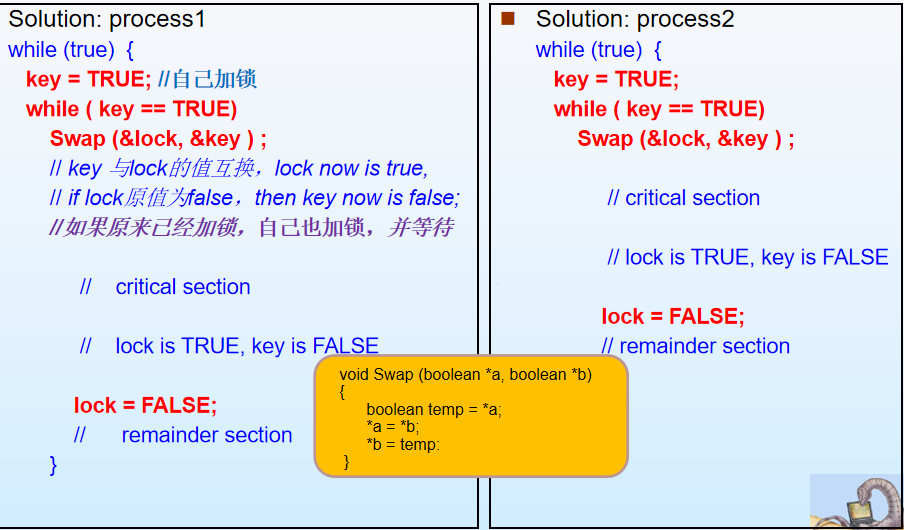


（2）硬件实现：TestAndSet指令和Swap指令。

TestAndSet指令



Swap指令

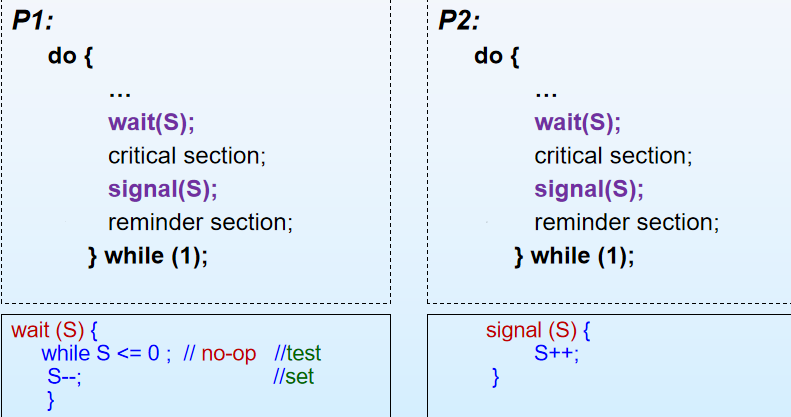


（3）信号量

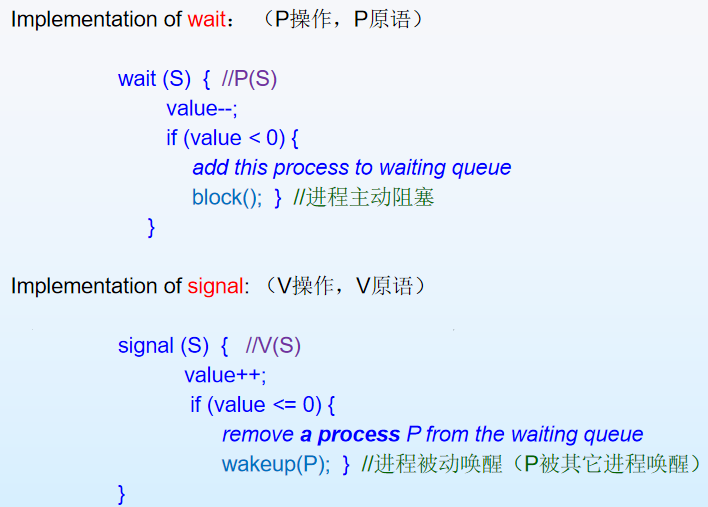
整型信号量：整型变量（自旋锁）

缺点是循环等待，占用CPU。

优点是进程在循环等待，不需要进行上下文切换，减少了系统开销，如果自旋锁等待的时间较短，自旋锁是有效的。



记录型信号量：整型变量和进程链表（非盲等）



（4）管程

提供更高层的方便用户同步机制，系统将其映射到底层的信号量及wait、signal操作，编译器保证管程互斥。

5.系统的分类 批处理 分时系统 原理

批处理系统（单道/多道）：为了解决人机矛盾及CPU和I/O设备之间速度不匹配的矛盾。系统对作业的处理是成批进行的，在系统中形成一个自动连接的连续的作业流，系统自动依次执行每个作业。

分时系统：操作系统将CPU时间划分为时间片，以时间片为单位轮流把CPU分配给在内存中的作业。

实时系统（硬/软）：为了能在某个时间限制内完成某些紧急任务而不需要时间片排队。有明确的时间约束，能及时处理有过程控制反馈的数据。(硬：保证关键任务按时完成，绝对严格时间内完成处理；软：能接受偶尔违反时间规定)

1. 重定向原理 输入输出不走标准输入输出（管道）实验要看

重定向的原理：系统创建进程的时候，会为每个进程自动打开三个标准设备：stdin、stdout 和 stderr，并为其分配三个文件描述符0、1、2，每个文件描述符与一个打开文件相对应。而重定向实际上就是**改变文件描述符表中描述符的指向，让描述符指向用户想要重定向的文件，使对一个文件的操作变成对另一个文件的操作。**

七个大题每题十分

1. 程序的运行，给一个代码段从编译开始到运行操作系统做了什么，程序运行流程，从操作系统的角度介绍C程序运行流程（外设交换数据等）

**程序运行流程：**编译、链接、装入、执行。

**编译：**由编译程序将源码编译成若干目标模块。

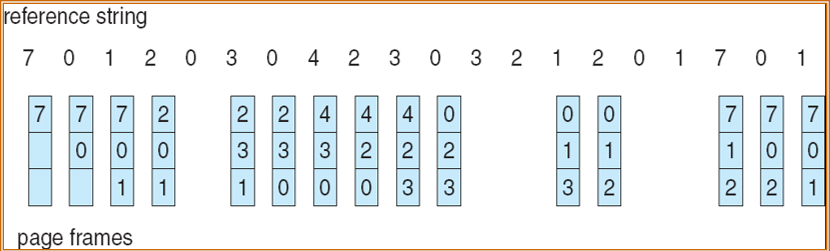
**链接：**由链接程序将一组目标模块及所需库函数链接在一起形成完整装入模块。（静态链接：运行之前完成链接，将所有的程序模块链接起来，形成一个可执行文件，运行时直接装入内存；动态链接：运行时仅链接需要的模块；运行调用时进行链接；）

**装入：**由装入程序将装入模块装入内存运行。（绝对装入：装入所有模块才执行；动态时装入：只装入主控模块或部分模块即执行，其他模块调用时装入。）

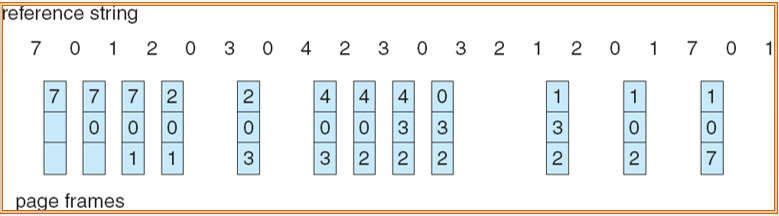
**执行：**会创建一个进程并为其分配资源等待调度，涉及到进程调度和内存管理。如果程序中还跟输入输出有关（比如printf和scanf等），那么还涉及到外设交换数据等。

1. 页面调度，算法（五问）

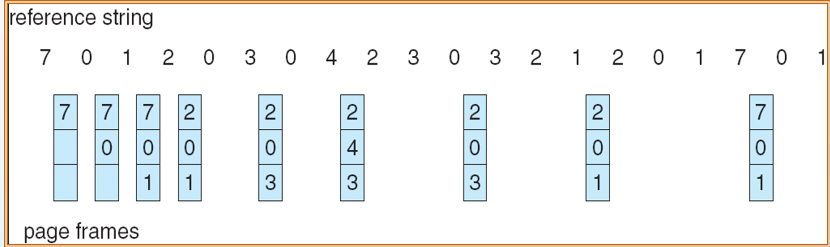
FIFO（先进先出）：优先淘汰最早进入内存的页面，即淘汰在内存驻留时间最久的页面（存在Belady’s Anomaly，即分配的帧越多，缺页率越高）注意FIFO中跟命中无关，只跟进入内存的时间早晚有关。



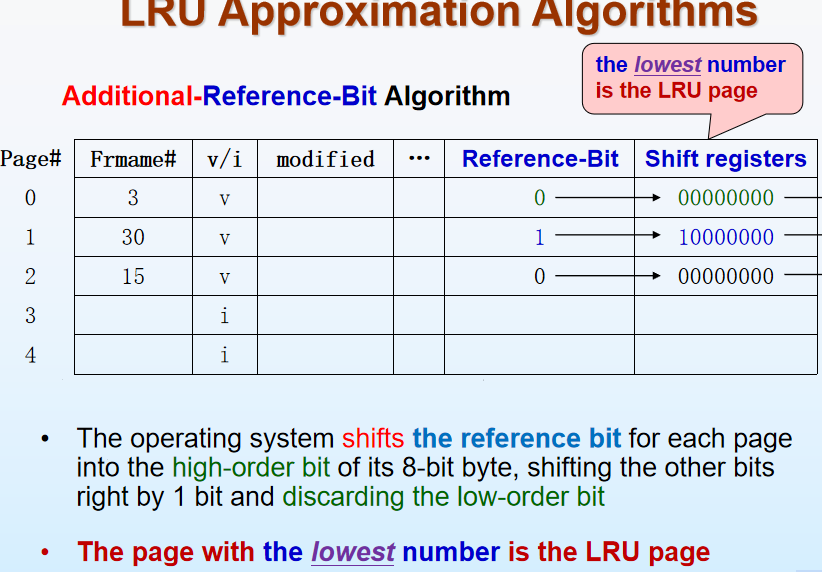
LRU（最近最久未使用）：选择最近最长时间未访问过的页面予以淘汰。



OPT（最佳）：最长时间内不再被访问的页面。



LRU近似算法（LRU算法实现起来开销很大，故采用近似算法代替）





3.死锁 银行家算法细节 需求矩阵 会不会死锁

请求指令流A（1，1，1），系统允不允许分配，这个1，1，1是系统能不能分配给指令流A再分配，不改需求矩阵，需求矩阵不变

概念：一组处于阻塞（等待）状态的进程，每个进程持有其他进程所需要的资源而又等待使用其他进程所拥有的资源，致使这组进程相互等待均无法向前推进。

原因：竞争资源，进程间推进顺序不当。

必要条件：互斥，不剥夺，请求并保持，循环等待。

处理策略：死锁预防，死锁避免，死锁检测和恢复。

银行家算法，需求矩阵。

银行家算法：

合法性检查：即判断Request向量是否小于Need向量。

可用性检查：即判断Request向量是否小于Available向量。

假分配，先假设分配。

然后进行安全性检查，如果能通过则分配资源，不能通过则让进程等待。

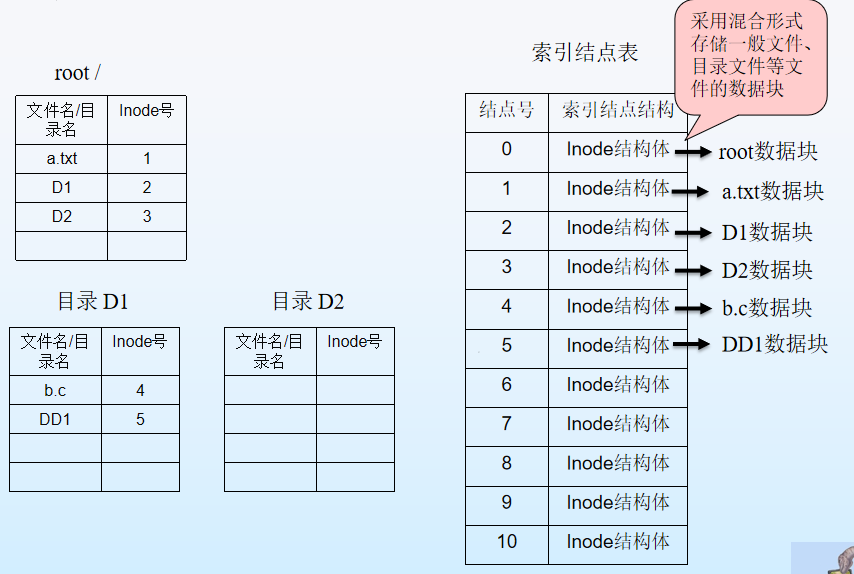
4.用户态内核态 分析代码能不能进入内核态 为什么这么设计 原则原理

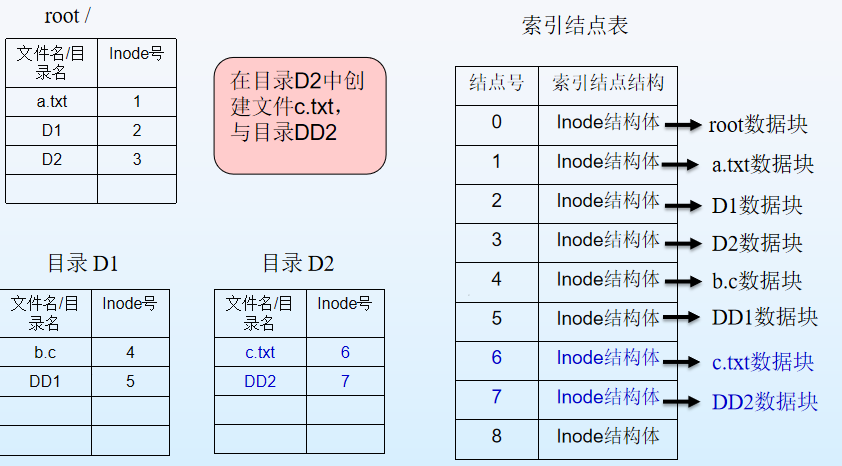
用户态和内核态是操作系统运行的两种状态。

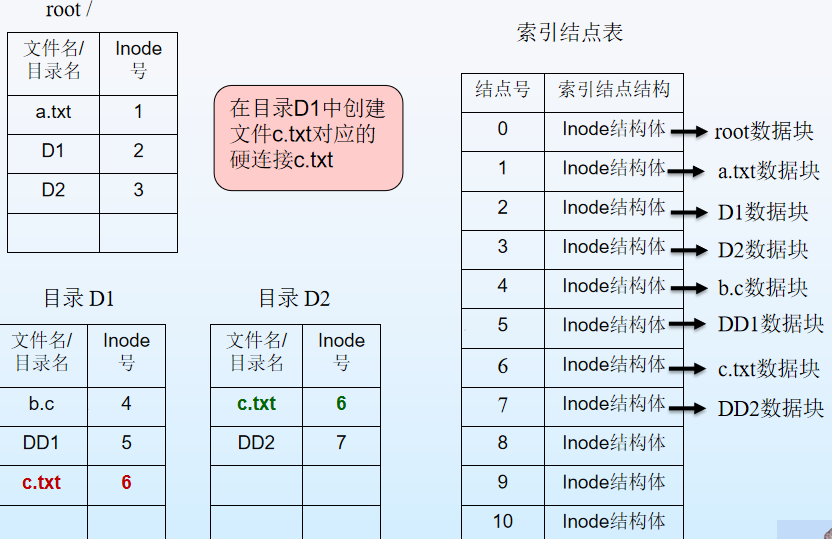
在用户态，CPU不能执行特权指令，在核心模式中，CPU可以执行其所有指令，包括特权指令。而特权指令则是指可能对操作系统或其它程序引起损害的指令，这样区分能保证系统的稳定性和安全性，也能防止用户影响其他进程的运行。

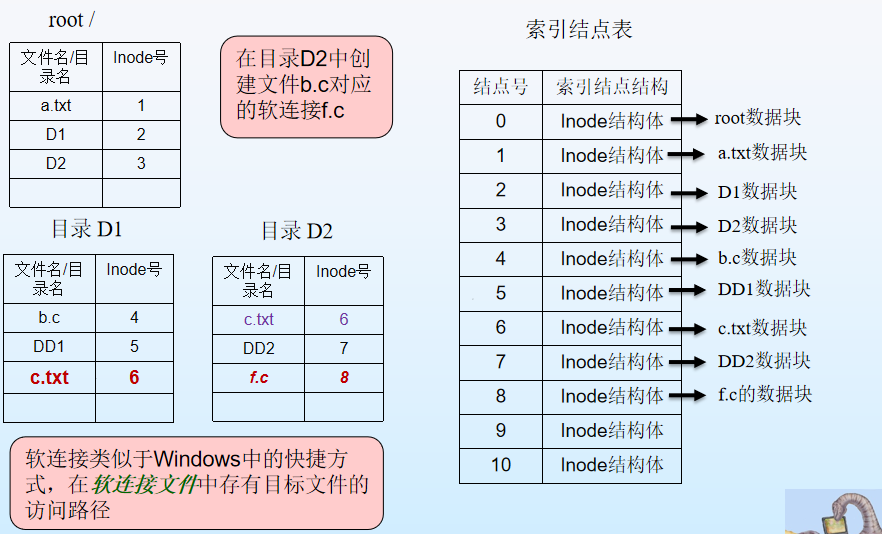
常见的内核态进行事件：代码涉及到系统调用（所有系统调用都是陷入内核态让操作系统代为执行），发生缺页错误，产生软中断，除0异常，I/O指令。

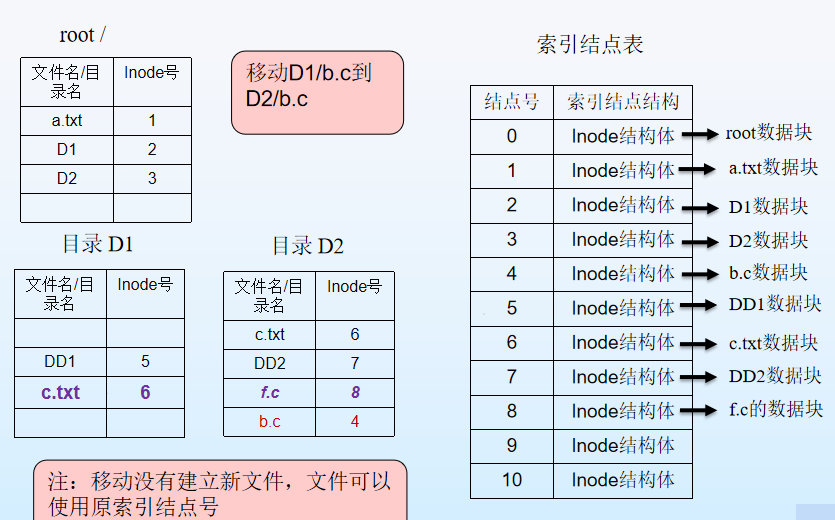
5.文件系统，文件索引图，加文件删文件，软硬链接，复制移动，剪切。

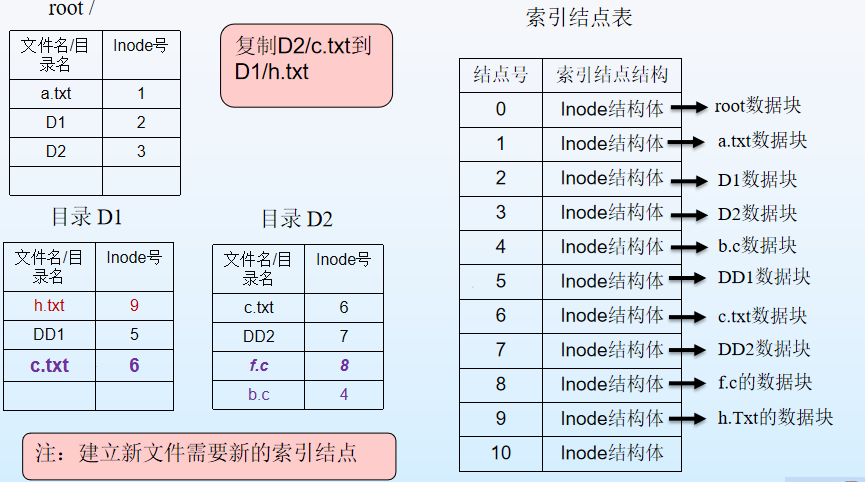


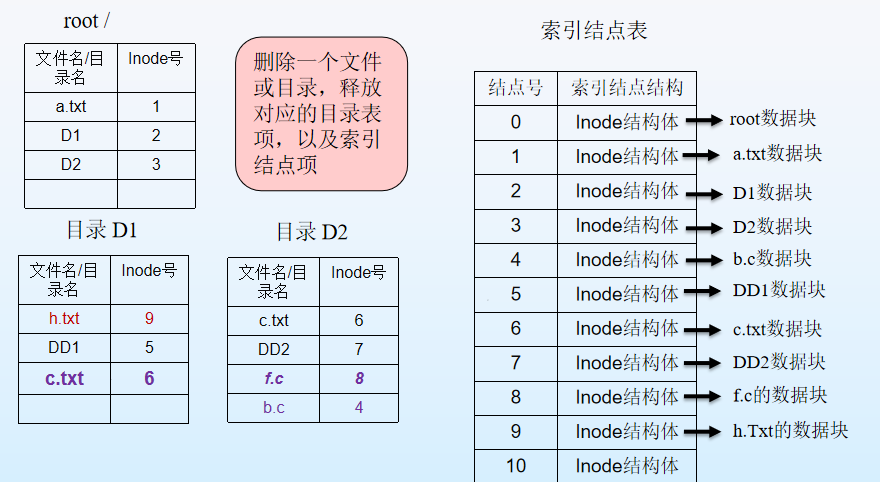










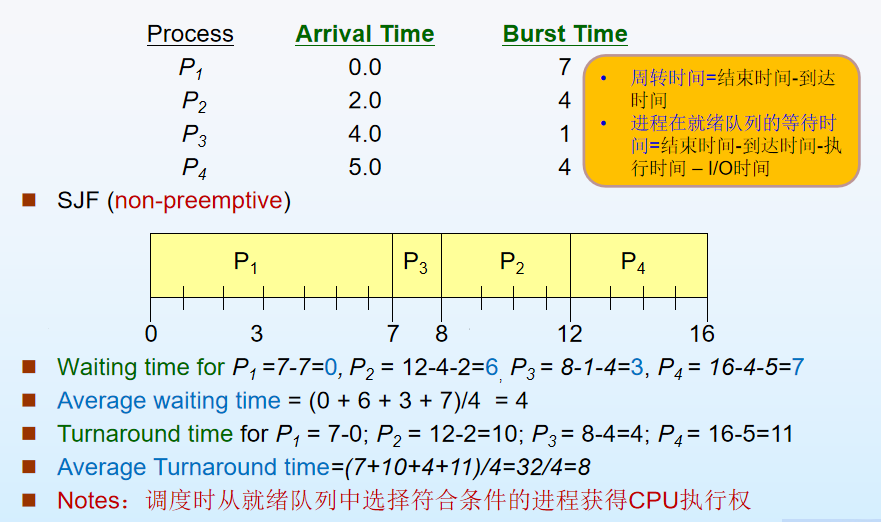


6.线程调度算法优缺点 甘特图 状态转移图



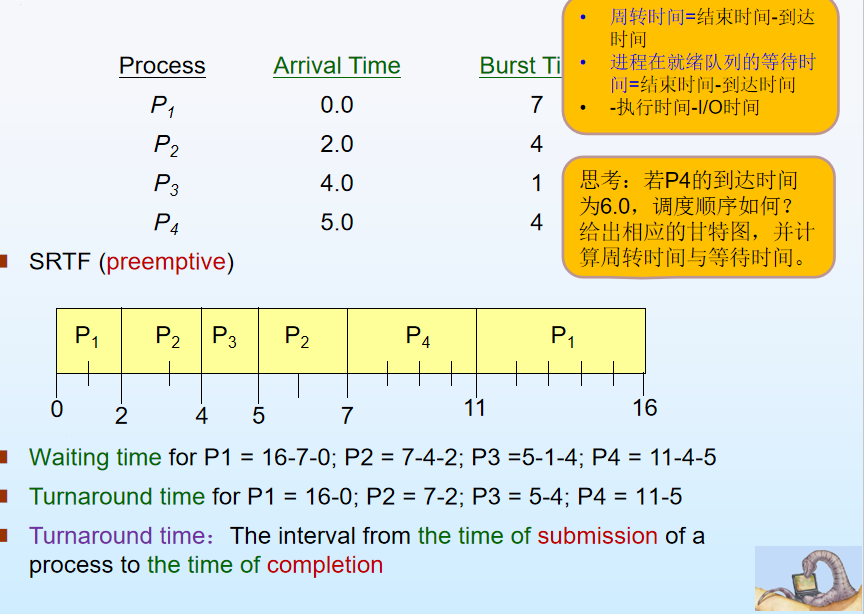
FCFS：先来先服务调度，先到的先被调度，有利于长作业，对短作业不利，算法简单，但效率低，系统吞吐量小，进程的到达顺序对调度性能影响很大。

SJF：最短作业优先调度，每次选择CPU执行时间最短的那个进程，非抢先。系统吞吐量大，平均周转时间短，有利于短作业，不利于长作业，可能出现饥饿。



SRTF：基于SJF的抢先式调度，开始的时候查看进程的就绪队列，选择时间最短的进程获得CPU的执行权。当一个新的进程进入就绪队列，将正在运行的进程的剩余时间与该新进入就绪队列进程的CPU Burst Time进行比较，以确定是否能够抢先。

在所有的进程调度算法中，SRJF调度算法平均等待时间与平均周转时间都最短。

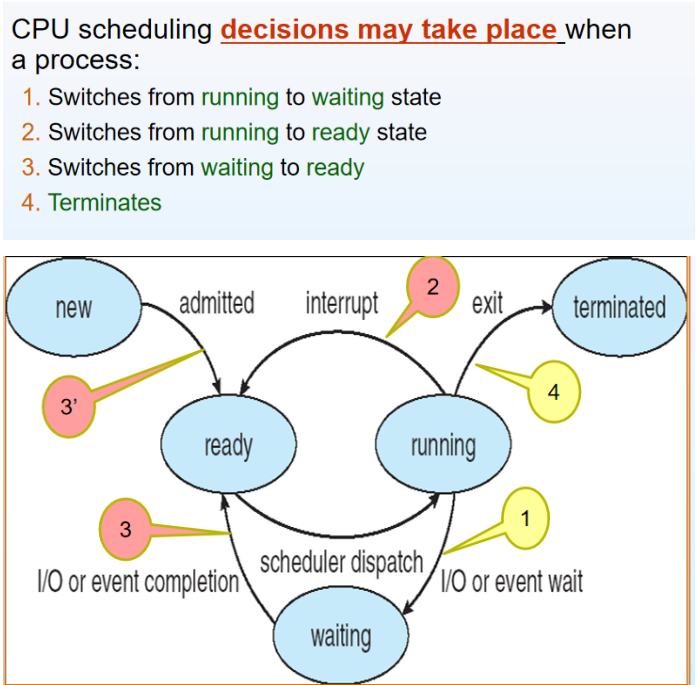
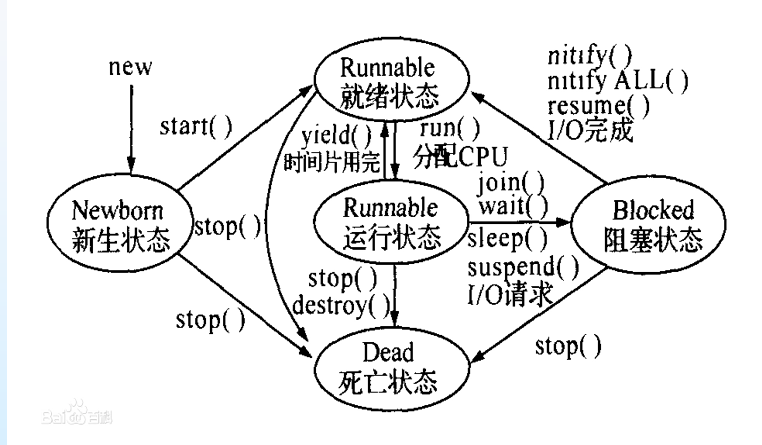


优先级调度：抢先/非抢先，数字小的优先级高，会产生饥饿。

高响应比：响应比=（等待时间+要求服务时间）/要求服务时间，可以解决饥饿。

该算法既照顾了短作业，又考虑了作业到达的先后次序，也不会使长作业长期得不到服务，防止出现SJF中长进程出现”饥饿”现象。

RR（时间片轮转）：抢先，公平，适用于分时系统，进程切换开销大，不区分任务紧急程度。



7.同步，考代码，补充代码 信号量互斥等等，类似上学期