判断题 10 题×1 分
填空题 20 空×1 分
单选题 10 题×1 分
简答题 3 题共 10 分
综合应用题 5 题 共 50 分（第五章第六章必有，比重也最大）

计算题有：27、38、51、55
费力不讨好的题有：7、34

1.有哪几种 I/O 通讯方式，它们各自的流程是什么？（有三种中断：可编程 IO、中断驱动 IO 和 DMA；把 IO 通道和 IO 处理器加入后讲了七种）
答：
可编程 I/O：处理器给相应 I/O 模块发送命令，然后进程进入忙等待。需要等待很长时间，需要不断检测 I/O，严重降低了性能。
中断驱动 I/O：处理器给 I/O 发送命令之后继续做其它工作，无不必要的等待。但是数据传送都必须完全通过处理器，消耗大量处理器时间。
直接内存存取 DMA：DMA 模块直接与存储器交互，传送完成时 DMA 发送中断信号给处理器。处理器只有在开始传送和结束时才参与。
发生总线竞争时，CPU 等待 DMA 一个总线周期。（不是中断）

2.高速缓冲存储器（cache）的原理和访问过程是什么？有哪些替换算法？（基本原则："替换掉在不久的将来最不可能用到的块"）算法的流程是什么？
答：
原理：　~~时间~~局部性，最近被访问的元素周围的元素在不久将来被访问。
　　~~空间~~局部性，最近被访问的元素在不久将来被访问。
过程：　当处理器试图读取存储器的一个字节或字的时，要进行一次检查以确定这个字是否在高速缓存中。如果在，则该字节从高速缓存传递给处理器，如果不在，则先将由固定数目的字节组成的一块内存存入缓存，然后将该字节从高速缓存传递给处理器。
替换算法基本原则是替换掉**不久的将来最不可能用到的块**。有效的策略是 LRU（最近最少使用）：把缓存**最长时间没被使用过**的块替换掉。

3.操作系统的发展经历了哪些阶段？（第二章没有上，但属于考试范围）（注意实时系统中如何对任务进行处理，系统任务任务堆积的时候怎么处理）
答：①串行处理：没有 OS，程序员直接与硬件打交道；②简单批处理系统；③多道程序批处理系统；④分时系统；⑤实时系统；⑥网络操作系统；⑦分布式操作系统；⑧云计算操作系统；

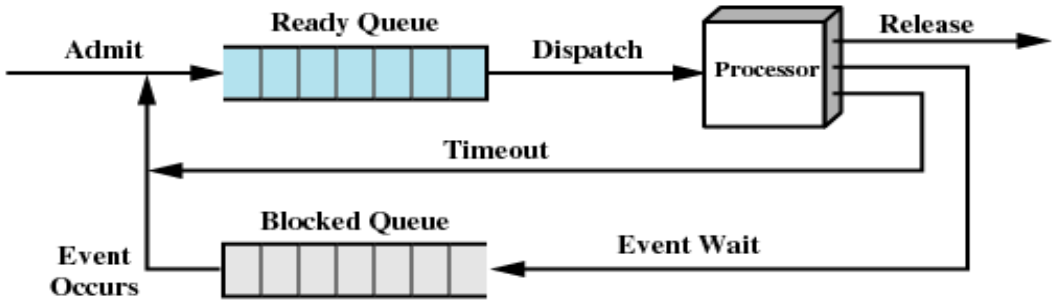
4.OS 的发展过程中有哪些重要的理论进展？（第二章）
答：①进程；②内存管理；③信息保护和安全；④调度和资源管理；⑤系统结构（旧版）

5.现代操作系统的特征是什么？（第二章）
答：
①微内核体系结构：只把一些最基本的功能放在内核中，其它服务由运行在用户模式下的进程提供,内核和服务程序分开。

②多线程：进程划分成可以同时运行的多个线程，在同一个进程中运行多个线程，在线程间来回切换所涉及的处理器开销要比在不同进程间切换的开销少。
③对称多处理：硬件上有多个处理器，这些处理器共享一个主存储器和 I/O 设备，有的处理器都可以执行相同的功能。
④分布式操作系统：一群计算机使用户觉得和使用单机一样。
⑤面向对象设计：用于给微内核增加模块化的扩展，使程序员在不破坏系统完整性下定制操作系统。

6.操作系统作为计算机资源的管理者，它管理哪些资源？（所有资源都归它管，全部都要列出来）
答：内存、处理器、IO 控制器、IO 设备和外存储器、进程、文件

7.什么是进程？进程和程序的主要区别是什么？进程状态变迁图（五状态、七状态，记住圈、线、线上的字），**进程状态变迁的触发条件是什么？挂起状态的进程有哪些特点？**（在硬盘上，不能执行，它们进程切换会是 IO 操作）
（阻塞/挂起的进程为什么要激活成阻塞进程？操作系统检测到它等待的事件马上要发生，由于挂起阻塞状态切换到阻塞状态是一个 IO 的过程，时间比较长。为了它等待的事件发生后可以尽快执行，需要把它调入内存中，即先切换到阻塞状态，从阻塞状态到就绪状态到运行状态是内存的过程，速度就比较快。所以需要把阻塞/挂起状态切换成阻塞状态。）
答：进程是一个正在执行的程序；进程=程序代码+数据+PCB；（一个是静态的一个是动态的，通常情况下没有对它们区分，只有特别拎起来才做区分）
五状态模型：新建态（未获内存分配）、就绪态（只差 CPU）、运行态（得到 CPU 控制权）、阻塞态（等待某些事件）、退出态



(a) Single blocked queue

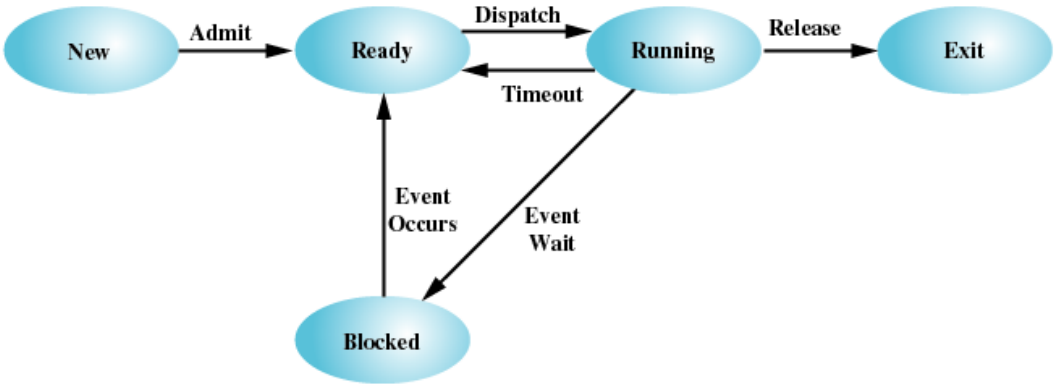
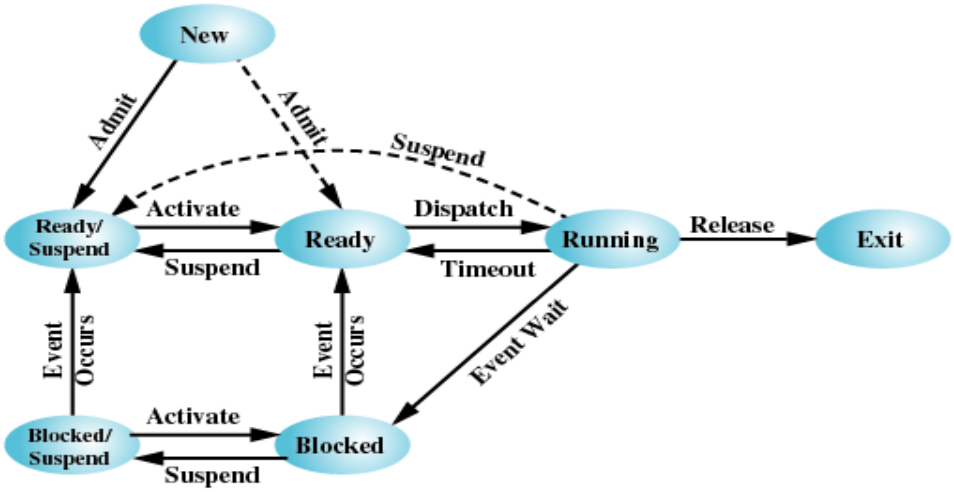


Figure 3.6 Five-State Process Model

七状态模型



(b) With Two Suspend States

Figure 3.9 Process State Transition Diagram with Suspend States

导致进程新建的原因：1、新的批处理作业 2、交互登录 3、OS 因为一项服务而创建 4、由现有进程派生

导致进程终止的原因：1、正常完成 2、算术错误 3、数据误用 4、超时时限 5、时间超出 6、操作员或 OS 干涉 7、无可用内存 8、I/O 失败 9、父进程终止 10、越界 11、无效指令 12、父进程请求 13、保护错误 14、特权指令

新建-》就绪：OS 接纳/允许该进程进入内存，获得调度资格，多道程序涉及道数+1

就绪=》运行：调度或分配

就绪=》退出：强制退出（系统、用户）、父进程终止

运行=》就绪：1、时间片到 2、主动释放 3、被抢占

运行=》阻塞：1、等待 IO 2、等待资源 3、等待网络通讯 4、等待进程同步 5、等待系统历程调用

运行=》退出：1、正常退出 2、异常退出：内存溢出、无效指令、越界访问、堆栈溢出、越权访问、算术错误、父进程
阻塞=》就绪：等待的事件发生
阻塞=》阻塞挂起：无就绪进程，至少有一个阻塞进程被换出让出空间
阻塞挂起=》就绪挂起：等待事件发生
就绪挂起=》就绪：内存中无就绪进程或者就绪挂起优先级高于所有就绪进程
就绪=》就绪挂起：优先挂起的是阻塞进程，迫不得已如此转换。OS 确定优先级高的阻塞进程马上就绪
新建/运行=》就绪挂起：没有足够内存
阻塞挂起=》阻塞：少见，有多余空间，优先级高且等待的事件要发生

挂起状态进程的特点
在硬盘上，不能执行，变换操作是 IO 操作
1.进程不能立即执行
2.进程可能是或不是正在等待一个事件（阻塞/挂起、就绪/挂起）
3.为阻止进程执行，可以通过代理把这个进程置于挂起状态，代理可以是进程自己，也可以是父进程或 OS
4.除非代理显式地命令系统进行状态转换，否则进程无法从这个状态中转移

8.进程控制块（PCB）的作用是什么？
答：PCB 包含了充分的信息，这样就可以中断一个程序的执行，并且在后来回复执行的过程中就好像进程未被中断一样。
作用：1 进程调度；2 资源调度；3 中断处理；4 性能监控和分析。

9.中央处理器的工作状态有哪些？（三个状态：空闲状态，工作状态分两种：一种做调度，一种做用户进程运行。）（有效时间指的仅仅是运行用户进程的时间）
答：空闲
用户模式：用户程序在用户态下运行，不能访问被保护的内存区域和执行特权指令。
内核模式：监控程序在内核态下运行，可能可访问被保护的内存区域，并可执行特权指令。调度。

10.操作系统的执行有哪些方式？
答：
①无进程内核：在所有进程之外执行 OS 内核，OS 代码作为一个在特权模式下工作的独立实体被执行，进程的概念仅仅适用于用户程序
②在用户进程中执行：OS 软件在用户进程中，OS 代码与数据在共享地址中空间，执行 OS 代码时，切换到内核模式，同一进程中执行，只进行模式切换不切换进程，用户代码无法干涉 OS
③OS 作为一组系统进程实现，多处理器或多机环境十分有用。有模式切换也有进程切换。

11.线程有哪些类型，有哪些运行方式？
答：
①纯粹的用户级线程：所有的线程管理工作都由应用程序完成，内核不可见线程，以进程为调度单位

②纯粹的内核级线程：有关线程管理的工作都由内核完成，内核管理进程和线程的上下文信息，调度基于线程
③混合方案：线程创建完全在用户空间中完成，线程的调度和同步也在应用程序中进行，但多个用户级线程被映射到内核级线程上
运行方式有：派生、阻塞、就绪、运行、结束。（其实就是五种状态，和进程的五状态是一样的）

12.什么是微内核？
答：内核的精简版，只给内核分配最基本的功能和调度，所有服务在用户模式下运行。

13.什么是进程切换，步骤是什么？（进程切换和中断处理主要都是三步：保护现场、恢复现场、转入中断处理和执行）
答：
进程切换是在某一时刻，一个正在运行的进程被中断，操作系统指定另一个进程为运行态，并把控制权交给这个进程
1.保存处理器上下文，包括程序计数器和其他计数器
2.更新当前处于运行态进程的 PCB，包括进程状态的改变
3.把进程的 PCB 移到相应队列
4.选择另外一个进程执行
5.更新所选择进程的 PCB
6.更新内存管理的数据结构
7.恢复处理器在被选择进程最近一次切换出时的上下文。

14.什么叫临界资源，什么叫临界区？
答：临界资源：一次只能被一个进程使用的资源，不可共享；
临界区：一段**代码**，有排他性。临界区就是使用临界资源的那部分程序。

15.信号量的物理意义是什么？大于 0、小于 0 的物理意义是什么？
答：
s.count>=0 可以执行 semWait（s）而不需要阻塞的进程数（如果期间没有 semSignal（s）被执行）
s.count<0 阻塞在 s.queue 队列中的进程数

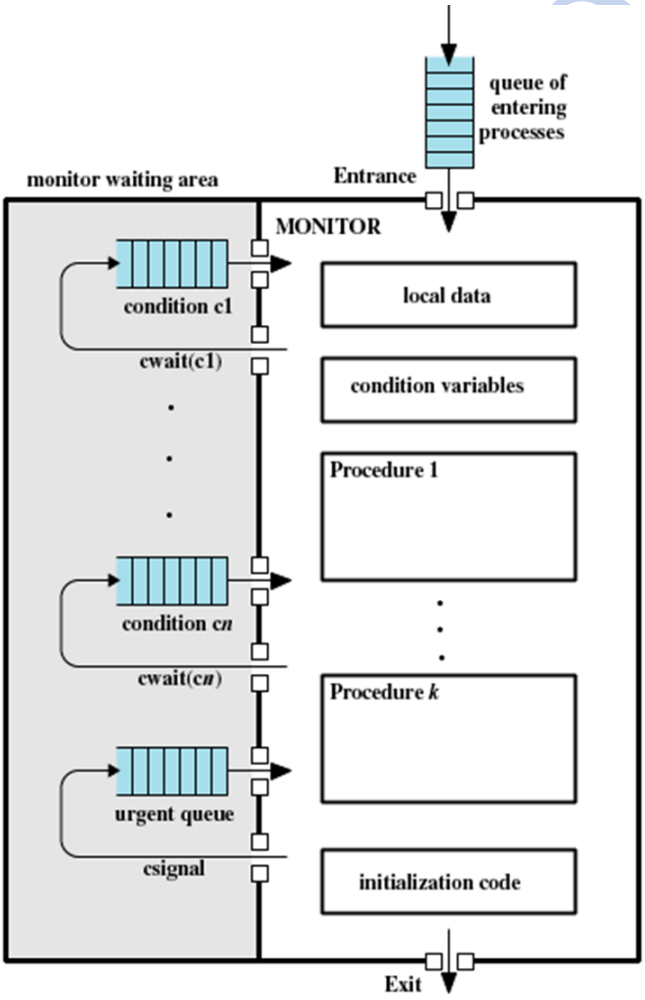
16.信号量的值与资源数量、进程数量的关系是什么？
答：①对于一般信号量，信号量的值如果是正的，可以表示当前可用资源的数量，也就是当前发出 semWait 操作后能立即继续执行的进程数量。资源总数减去信号量的值可表示正在占用资源的进程数。信号量的值如果是负的，则表示正在等待资源等待解除阻塞的进程数量。
②对于二元信号量，信号量只能体现资源的有无。信号量为 1 时表示有资源，可能有进程执行，信号量为 0 时表示没有资源，一定有进程执行在，可能有进程被阻塞。

17.信号量如何控制程序的运行？
答：两个或多个进程可以通过简单的信号进行合作，一个进程可以被迫在某一位置停止，直到它接收到一个特定的信号。具体表现为使用 semSignal 和 semWait 操作通过信号量传送(semSignal)和接收(semWait)信号。（注意都是接收和传送的主体都是信号量）

结合对信号量的三个操作：1.信号量可以被初始化成非负数；2.semWait 操作使信号量-1。若值为负数，则执行 semWait 的进程被阻塞。否则进程继续执行。3.semSignal 操作使信号量+1。若值小于或等于 0，则被 semWait 操作阻塞的进程被解除阻塞。(值大于 0 说明不存在被该信号量阻塞的进程)

18.如何使用信号量来解决资源竞争和进程同步的实际问题？（独木桥、红绿灯、有限/无限缓冲区生产者和消费者问题、父亲女儿儿子水果问题、公共汽车问题、和尚喝水问题）
答：四个步骤：1.把**行为主体**抽象出来；2.找出**临界资源**（理清资源的互斥规则）；3.主体执行的步骤（**行为过程**）；4.行为主体的**同步规则**列出来；（然而这题真正目的是叫我们去把这些题都弄懂怎么做并不是真的要答步骤）

19.什么是管程？其结构是什么？与信号量相比它的优点是什么？(特别需要注意的三个排队队列：入口队列、条件变量队列、紧急队列)
答：管程是一个程序设计语言结构，是由**一个或多个过程**/函数（映射到主体的行为），一个**初始化序列**（赋初值）和**局部数据**（条件变量、共享缓冲区等）组成的软件模块。
具有特点：①局部数据变量只能被管程的过程访问，任何外部过程都不能访问。②一个进程通过调用管程的一个过程进入管程。③在任何时候，只能有一个进程在管程中执行，调用管程的任何其他进程都被阻塞，以等待管程变得可用（体现互斥机制）。
管程的结构为：1.入口出口和入口队列；2.条件变量和它们的条件队列以及紧急队列；3.局部数据；4.过程和初始化代码；



```
/* program producerconsumer */
monitor boundedbuffer;
char buffer [N];                                    /* space for N items */
int nextin, nextout;                                  /* buffer pointers */
int count;                                  /* number of items in buffer */
cond notfull, notempty;          /* condition variables for synchronization */

void append (char x)
{
     if (count == N)
        cwait(notfull);                    /* buffer is full; avoid overflow */
     buffer[nextin] = x;
     nextin = (nextin + 1) % N;
     count++;
     /* one more item in buffer */
     csignal(notempty);                         /* resume any waiting consumer */
}
void take (char x)
{
     if (count == 0)
        cwait(notempty);                /* buffer is empty; avoid underflow */
     x = buffer[nextout];
     nextout = (nextout + 1) % N;
     count--;                                    /* one fewer item in buffer */
     csignal(notfull);                         /* resume any waiting producer */
}
{                                                           /* monitor body */
     nextin = 0; nextout = 0; count = 0;           /* buffer initially empty */
}
```

```
void producer()
char x;
{
     while (true)
     {
     produce(x);
     append(x);
     }
}
void consumer()
{
     char x;
     while (true)
     {
       take(x);
       consume(x);
     }
}
void main()
{
     parbegin (producer, consumer);
}
```

Figure 5.16 A Solution to the Bounded-Buffer Producer/Consumer Problem Using a Monitor

20.什么是消息传递？如何通过 mailbox（信箱）实现进程的同步与互斥？

答：消息传递是指使用一对消息原语 send(destination,message)和 receive(source,message)实现进程间的同步和交换信息/通信。(同步是为了实施互斥，通信是为了进程合作)

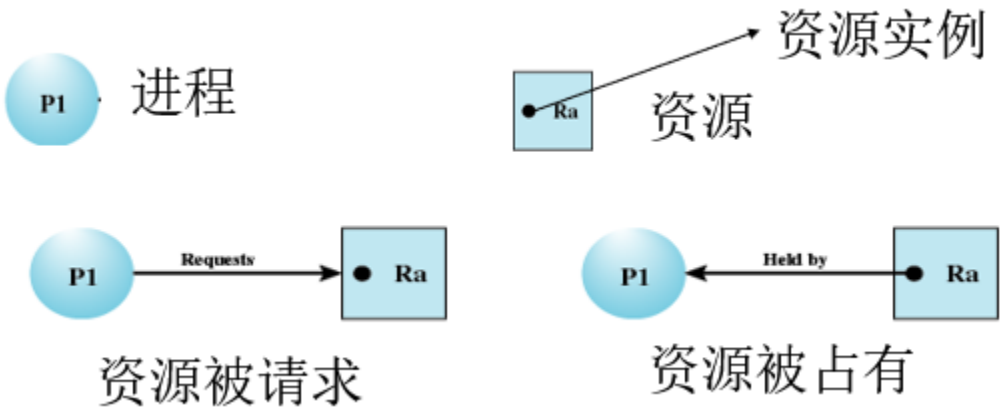
（对于阻不阻塞发送进程和接收进程，除了阻塞 send，不阻塞 receive 不可行外有三种可行组合。默认使用的是不阻塞 send，阻塞 receive。）

mailbox 指的是通过**间接寻址**的方式实现的消息传递。即消息不是直接从发送者发送到接收者，而是**发送到共享的保存临时消息的队列**(可能不止一个)，这些队列就叫 mailbox(信箱)。两个进程之间的消息通信**暗含了同步**:只有当一个进程发送消息之后,接收者才能接收消息。进程间**互斥的控制**：一条消息仅被传递给一个进程。

21.什么是资源分配图？从图上可以获取哪些信息？

答：阐述系统资源和进程的状态的有向图。节点表示资源（方形）和进程（圆形）；资源节点中的一个圆点表示资源的一个实例；箭头从进程节点指向**资源节点**表示进程申请资源；箭头从**资源实例（圆点）**指向进程表示该进程实例被指向的进程占有。

从图上可以获得进程名、资源名（类型）、进程数量、资源种类（方节点个数）和数量（圆点个数）、进程对资源的请求情况和进程对资源的占有情况。



22.死锁产生的 4 个条件是什么？死锁的三种解决方法？

答：死锁的 4 个条件为：

1.互斥；2.占有且等待；3.不可抢占；4.循环等待（官方定义：存在一个封闭的进程链，使得每个进程至少占下一个进程所需要的一个资源）。

（其中 123 为必要非充分条件；4 加入形成充分必要条件。）

死锁的三种解决方法：

1、死锁预防：至少破坏死锁的四个条件之一。互斥作为资源特性一般不去打破；占有且等待：通过要求进程一次性请求所有需要的资源，并将进程阻塞直到满足所有资源的要求（即资源打包，一次性申请），将占有且等待变成等待不占有/占有不等待；不可抢占：主动释放自己已占有的或者要求另一个进程释放资源（抢占）；循环等待：定义资源类型的线性顺序（资源线性化），并要求进程只能申请已分配的资源类型之后的资源类型。

2、死锁避免：（允许三个必要条件）①如果进程的请求会导致死锁，则不启动该进程。（假设新进程+所有当前进程都物理并行，它们最大资源需求(之和）都能被满足才启动新进程）②如果一个进程增加的资源请求会导致死锁，则不允许此分配（银行家算法）。

3、死锁检测：周期性检测死锁，如果没有发现死锁，就把当前状态标记为检查点。如果发现死锁就活锁，可能的方法：1.取消所有死锁进程（被检测过程没有被标记的进程就是死锁进程）；2.回滚所有死锁进程到检查点；3.逐个取消死锁进程；4.逐个抢占死锁进程占用的资源（不一定一次性抢占死锁进程的所有资源）。

23.什么是进程的竞争？竞争会不会引起死锁？如何解决死锁？（即 22 的三个方法）

答：当并发进程竞争使用同一资源时，它们之间会发生冲突；可能会引起死锁；

（本问题对应考题应该是给出进程和资源数量，判断是否会出现死锁。参考 06 章 PPT 练习二和练习四）

（具体案例中判断死锁是否存在：存在，举个例子；不存在，通过某个充要条件无法满足做原理性说明）

已知某系统中的所有资源是相同的（只有一种资源），系统中的进程严格按照一次一个的方式申请或释放资源。在此系统中，没有进程所需要的资源数量超过系统的资源总拥有数量，试对下面列出的情况说明是否会发生死锁。

情况序号	系统中进程数	资源总量
A	1	2
B	2	1
C	2	2
D	2	3

- 某个系统有两个进程和三个资源。每个进程最多需要两个资源。这种情况下有没有可能发生死锁？为什么？
- 假设有三个进程，每个进程需要四个资源，则系统至少需要有多少资源才会保证不会死锁？
- 假设现在有P个进程，每个进程最多需要m个资源，并且有r个资源可用。什么样的条件可以保证死锁不会发生？

24.什么是银行家算法？引入银行家算法为了解决什么问题？其具体的计算过程和步骤是什么？（5 步：合法性判断、可用性判断、预分配、状态检测、执行分配）

答：银行家算法是死锁避免的资源分配拒绝策略。引入是为了解决死锁问题。具体的计算过程和步骤有 5 步：

① 合法性判断：将已分配资源数加上(本次)(Allocation)请求的资源数不能大于声明的对资源的最大需求数(Claim)。

② 可用性检测：（本次）请求的资源数不能大于（当前）可用资源数(Available)。

③ 预分配：满足①②后模拟资源分配。

④ 状态检测（安全检测）：检测分配后得可用资源数(Aailable)能否至少满足一个进程当前需要的最大资源数(Claim-Allocation)

⑤ 执行分配：包括如果是安全状态就分配资源，如果是非安全状态就将预分配回收。

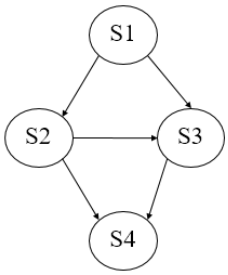
25.银行家算法当中，什么是安全状态？什么是不安全状态？不安全状态和死锁的关系是什么？

答：安全状态是至少有一个资源分配序列不会导致死锁（即所有进程都能运行直到结束）。非安全状态就是在当下会存在无法获得到所有需要的资源的进程的状态。

死锁一定是非安全状态。**非安全状态有可能引发死锁，但是不一定是死锁**，因为可能会有进程主动释放自己已占有的资源重新排队，从而转变成安全状态。（活锁的方法之一）

26.什么是进程的趋势图？如何根据进程趋势图实现进程同步？（看 PPT（看了也没用），当时没讲，相对简单）

5.7 进程趋势图



答：表示进程的先后关系。S1 执行完 S2 和 S3 才能执行。S2 执行完 S3 和 S4 才能执行……本图可以简化为 S1→S2→S3→S4，但是具体做题不能做这样的简化。

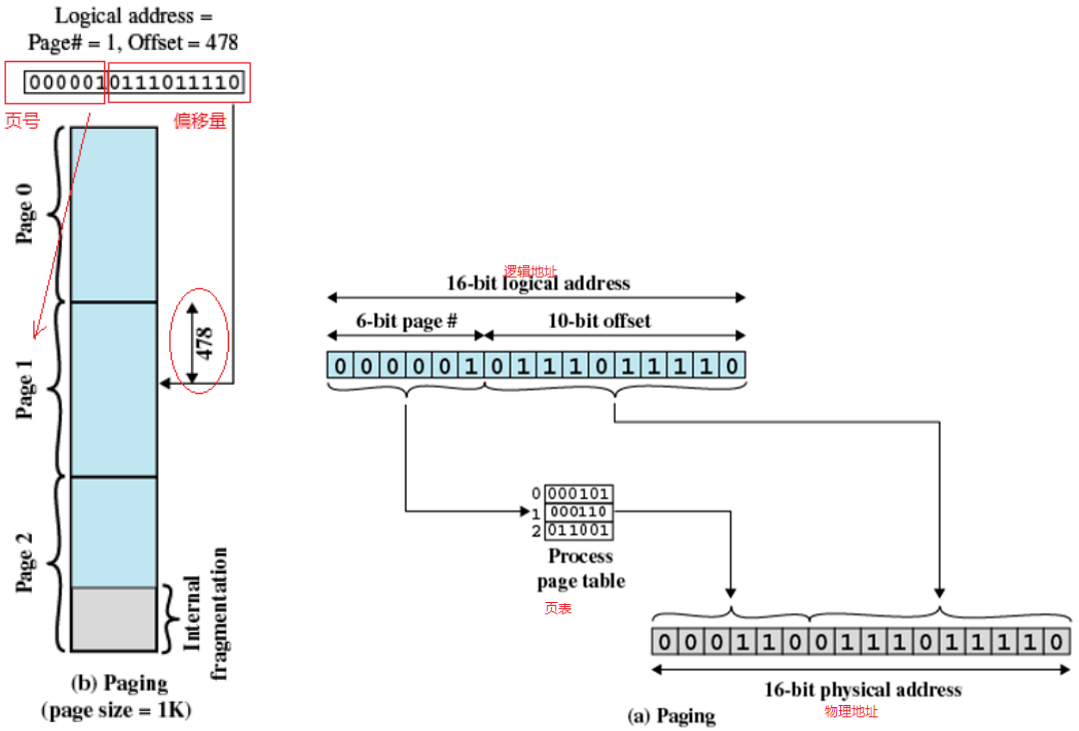
用信号量 s1 等于 0 表示进程 S1 还没结束，进程 1 最后 semSignal(s1)表示 S1 结束，那么进程 S2 和进程 S3 的开头都要 semWait(s1)。这里会出现一个问题，进程 S1 是两个进程（S2 和 S3）的前驱，所以 S1 需要两个 semSignal，才能满足进程 S2 和进程 S3 都能接收到信号。
（需要出度数量的 semSignal）
（可能会给你一个进程趋势图让你用信号量实现。）

27.什么是磁盘内存的动态分区？可以使用哪些算法进行空闲区块的分配？什么是伙伴系统？如何确定伙伴的地址?

（老师自己的联想：并发性的 6 种解决方案（中断、两个专有机器指令、消息、信号量、管程）。内存管理也有 6 个方法：动态、静态、伙伴系统、页式系统、段式系统、段页式系统）
答：动态分区是管理用户内存空间的一种方案，把用户内存空间分区，分区长度和数目是可变的。当进程被装入内存时，给它分配一块和它所需容量完全相等的内存空间。分配算法（即放置算法）有：①最佳适配：选择与要求的大小最接近的块；②首次适配：从开始扫描内存，选择大小足够的第一个可用块；③下次适配：从上一次放置的位置开始扫描内存，选择下一个大小足够的可用块。
伙伴系统：开始时，可用于分配的整个空间视为一个大小为 2^U 的块。如果请求的大小 s 满足 $2^{(U-1)}<s\leqslant 2^U$，则分配整个空间。否则，该块被分成两个大小相等的伙伴，大小均为 $2^{(U-1)}$。如果有 $2^{(U-2)}<s\leqslant 2^{(U-1)}$，则给该请求分配两个伙伴中的任何一个；否则，其中一个伙伴又被分成两半。这个过程一直继续，直到产生大于或等于 s 的最小块，并分配给该请求。当一对伙伴都变成未分配时，它们将合并成一个大块。
伙伴系统的地址：设块大小为 2^n，则其伙伴系统的第 n+1 位与其不同（0 和 1 相对），其他位相同；（伙伴系统貌似是最佳适配）
例：011011110000 当块大小为 4，其伙伴系统为 011011110100，当块大小为 16，其伙伴系统为 011011100000

28.什么是页表？页表的作用是什么？（页表做地址映射，从逻辑地址到物理地址）
答：在页式管理系统中，操作系统为每个进程维护的一个记录了进程的每一页对应的帧（页框）的位置的表。页表的作用就是存储进程的每一页在内存中的帧（页框）的位置。可以用于给定逻辑地址时查询页表获得页框号从而可以将逻辑地址转化为物理地址。

29.给定页表或段表，如何进行逻辑地址到物理地址的映射和转换？有哪些需要考虑的因素？具体过程是什么？
答：先了解分页系统：将进程分成大小相等的小块，每块称为一页，将内存分成同大小的小块，每块称帧（页框）。将进程每一页装进内存的空闲帧中。（链表和物理存储的关系）进程形式上是连续的，用逻辑地址表示进程中每个字节空间的位置，逻辑地址 = 页号+（拼接）偏移量。进程存进的内存位置未必是连续的，在内存中的位置为物理地址，物理地址=帧号（页框号）+（拼接）偏移量。由于偏移量指的是距小块开始位置的距离大小，所以物理地址的偏移量=逻辑地址的偏移量。而页表中可以查到每一页的帧号（页框号），所以可以通过逻辑地址的**页号去查页表，获得**存在内存的**页框号**。拼接完就映射成物理地址了。



段式同理。不过段表中存的是真正的**内存起始位置的真实物理地址**，所以最后不是拼接是真的加法。

30.什么是二级页表？（就是页表的页表）**什么是反向页表？**（用内存作为计算起始的方式）
答：二级页表就是页表的页表，根页表中存的是二级页表的起始位置，如下图：

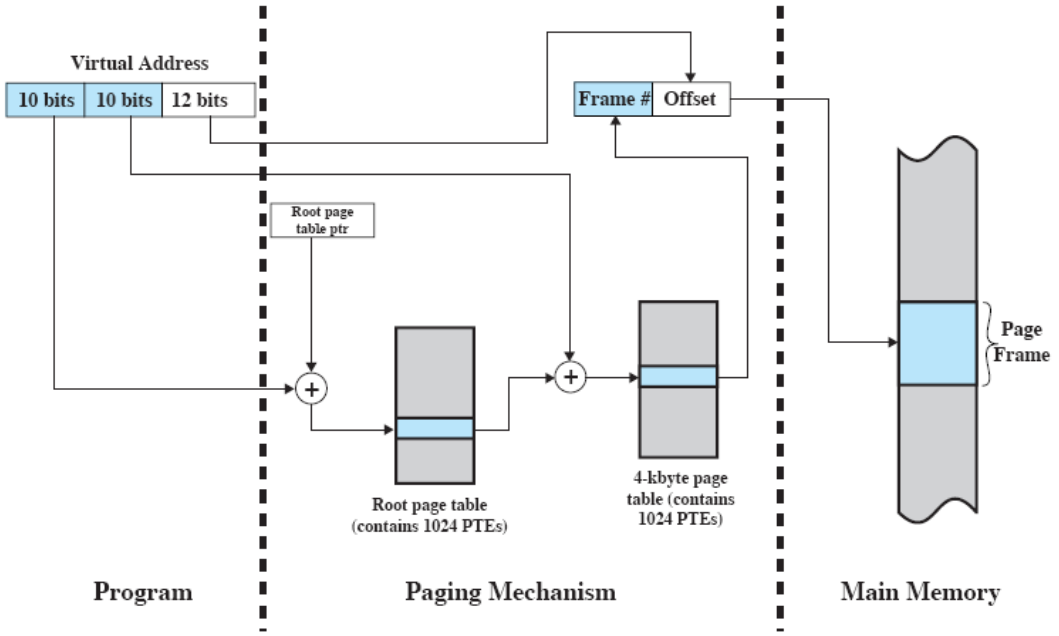


Figure 8.5 Address Translation in a Two-Level Paging System

倒排页表：虚拟地址的页被某哈希函数映射到一个哈希表中，散列表包含一个倒排表的地址，

而倒排表中含有页表项。倒排页表之所以称为倒排，是因为它使用页框号而非虚拟页号来索引页表项。

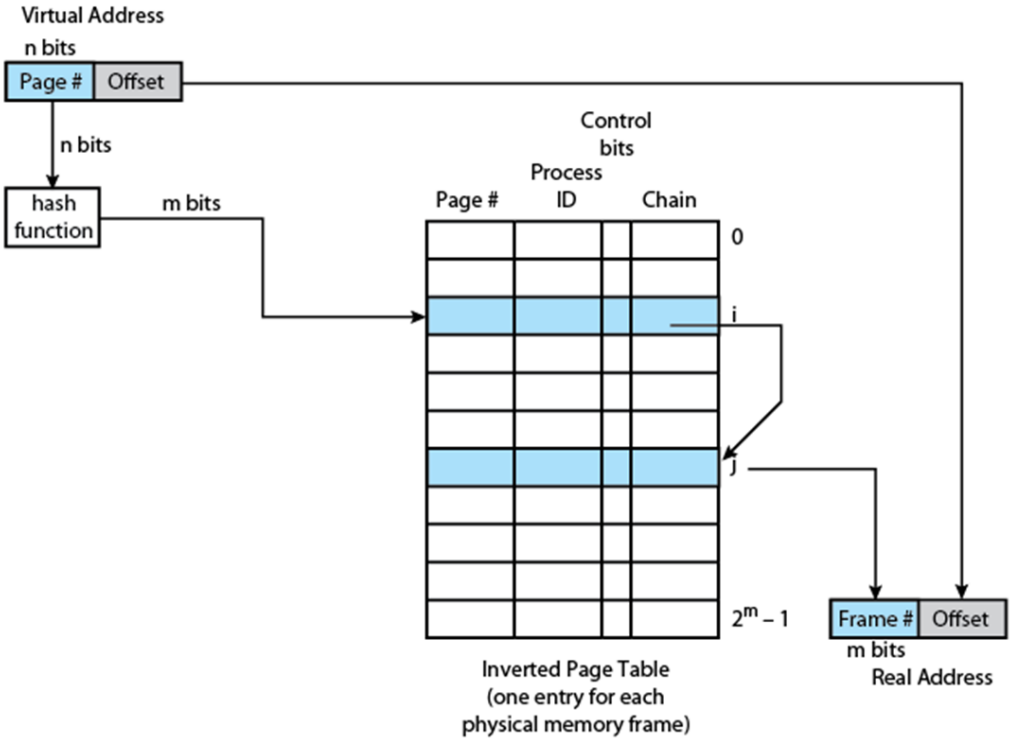


Figure 8.6 Inverted Page Table Structure

31.时间片、系统响应时间、系统效率之间有什么关系?

答：时间片：进程在被中断前可执行的最大时间段。

时间片的长度影响系统响应时间，时间片越短，系统响应时间有可能就越短（短作业会相对比较快地通过系统），但是可能系统周转时间会变长，而系统响应时间越短一般意味着系统效率越高（即在最短的时间内处理最多的事务）

32.页式管理系统中，页面大小与内存浪费的关系?

答：页式管理系统对于**每个进程内存的浪费是集中在最后一页的一部分**，理论上来说是越小越好。但是实际上，**页面越小意味着页表项越多，页表也就越大，浪费更多内存（来维护页表）的同时降低页面换进换出的效率**。所以要选择适中的大小。

网上：在确定地址结构时，若选择的页面较小，一方面可使内碎片减小，从而减少了内碎片的总空间、有利于提高内存利用。但另一方面，也会使每个进程要求较多的页面，从而导致页表过长，占用大量内存。此外，还会降低页面换进换出的效率。若选择的页面较大，虽然可减少页表长度，提高换进换出效率，但却又会使内碎片增大。因此。页面的大小应选得适中，通常页面的大小是 2 的幂，即在 512B ~ 4096B 之间。页面大小与磁盘调度的关系不大，磁盘调度与扇区有关。

33.多个进程如何共同使用同一个临界资源?

答：临界区、互斥。

使用临界区的方法，进程在临界区使用临界资源，且临界区中一次只能有一个进程。

34.什么是缺页中断？如何处理缺页中断？

答：缺页中断：所谓缺页中断，即是指当页表查找的页不在内存当中时，就发生了一次缺页中断。

缺页中断的处理：缺页中断分为有无 TLB 的情况。

有 TLB：首先 CPU 检查 TLB，查看所需页表项是否在 TLB 中，如果在，那么就会直接通过页表项取出所需的页。如果不在，就去检查页表项。如果该页在内存中，就更新 TLB（即把这个页的页表项放进 TLB），然后去找物理地址；如果发生了缺页中断，操作系统就通知 CPU 去磁盘中查找该页，CPU 激活 I/O，该页从磁盘中放入内存（如果满了就置换），然后更新页表，然后返回到出错指令（这里的返回到出错指令是指，返回到要查地址的那个指令，也就是一开始重新取页表项），然后进入 TLB 继续之前的步骤。

35.操作系统如何实现文件名与文件实际物理存储之间的映射关系？

答：通过文件目录。

与任何文件管理系统和文件合集相关联的是文件目录，目录包含关于文件的信息，这些信息包括属性、位置、和所有权。大部分这类信息，特别是与存储相关的信息，都是由操作系统管理的。文件目录本身也是一个文件，并且可以被各种文件管理例程访问。

目录在用户和应用所知道的文件名与文件自身之间提供了一种映射。

36.linux/unix 系统如何创建进程？（使用 fork()函数）

答：通过在父进程中调用 fork()函数，该函数对父进程返回子进程的 PID，对子进程返回 0.

37.什么是最先适配算法、最佳适配算法、最近适配算法？

答：这三种算法都是在内存中选择等于或大于该进程的空闲块。

最佳适配：选择与要求的大小最接近的块。

首次适配（最先适配）：从开始扫描内存，选择大小足够的第一个可用块。

下次适配（最近适配）：从上次放置的位置开始扫描内存，选择下一个足够的可用块。

38.什么是作业进程的平均周转时间？在不同的调度算法下如何计算平均周转时间？（注意归一化的周转时间即最高响应比）

答：周转时间就是驻留时间或这一项在系统中花费的总时间（等待时间+服务时间）。

平均周转时间：所有作业周转时间/作业个数

在不同的调度算法下计算平均周转时间其实本质上就是计算不同调度算法执行全 部进程所需要的总的周转时间之和。（只有 SRT 是可抢占的）

FCFS（先来先服务）：最先来的最先服务。

RR（roll-roll,轮转）：其实就是先来先服务加上了一个时间片，当一个进程的时间片结束而它没有全部完成的时候，它就排到队列最后去。然后继续开始循环。（注意，在这个方法中，当一个进程时间片结束而它没有全部完成的时候，若此时有一个新的进程同时进入，应该是新进程在前，时间片结束的进程在后）。

最短进程优先（SPN,short process next）：即最短执行时间的进程优先执行。（这是个不可抢占的策略，当一个进程执行结束才会选择下一个要执行的最短进程）

最短剩余时间（SRT,short rest time）：这是一个可抢占的策略，即当有更短时间的进程进入时会抢占进程，同时当一个进程结束的时候也是按照最短剩余时间来选择。

最高响应比优先（HRRN,hight response ratio next）：（响应比 R=（w+s）/s，w 为等待处理器时间，s 为预计服务时间）这也是个不可抢占策略，即选择响应比最高的进程。

39.什么是文件的逻辑结构？什么是文件的物理结构？
答：逻辑结构：P371 文件组织就是文件中记录的逻辑结构，由用户访问记录的方式确定。
物理结构：即文件在辅存的分配方式。

40.什么是平均寻道长度，在不同的调度算法下如何计算平均寻道长度？（计算，一般情况下都是列表。一定不要一下子出结果，要把过程告诉他）（这么强调看起来必考）
答：平均寻道长度即横跨磁道道数除以寻找的磁道数。
FIFO（先进先出）：即顺序处理队列中的项目。是一个比较公平的策略。
SSTF（SSTF,short server time first,最短服务时间优先）：即找到离当前位置移动最少的请求。
SCAN：就是从开始磁道按着一个方向扫描，到了最后一个请求的位置回过头来又向着一个方向扫描（此时和之前的方向相反）。
C-SCAN：和 SCAN 十分类似，但是不一样的地方在于他到了最后一个磁道之后不是按照与原来相反的方向进行扫描，而是跳到磁盘相反的方向从第一个请求开始继续沿着之前的方向扫描。（就是如果是由小到大扫就一直是由小到大，SCAN 到了尽头回头是由大到小）

41.什么是虚拟存储器？其理论基础是什么？
答：虚拟存储器：是一种为了给用户提供更大的随机存储空间的技术（不是实际的存储器）。也就是进程的一部分在内存中而一部分实际上存在的是磁盘中，但是程序认为它本身拥有那么大的内存。
其理论基础是局部性原理（包括空间局部性和时间局部性）以及重定位技术（重定位技术里重要的是基址寄存器和界限寄存器）。

42.什么是程序的局部性原理？（包括时间和空间）
答：时间局部性原理：即如果一个信息项正在被访问，那么在近期它很可能还会被再次访问。
空间局部性原理：最近的将来使用到的信息很可能与正在使用的信息在空间地址上是临近的。

43.页式管理系统、段式管理系统、段页式管理系统的特点各是什么？
答：页式管理系统：所有的页都是等大小的，内部碎片存在于进程的最后一个页的一部分。没有外部碎片。对程序员是透明的。
段式管理系统：段的大小是不相等的，可以认为是没有内部碎片。其优点有：简化对不断增长的数据结构的处理、允许程序独立地改变或重新编译、有助于进程间的共享、有助于保护。对程序员是不透明的。
段页式管理系统：即进程分段，段内分页。就是说一个内存空间有一个段表，每个段对应一个页表。即总共一个段表、多个页表。内存还是会有分页框大小。

44.页表中页的大小、地址位数以及空间大小之间的关系是什么（包括逻辑关系和物理关系。物理关系就是空间到底怎么存，逻辑关系就是它们相互之间有什么样的映射）
答：页越大其地址位数就长，所占的空间根据进程来决定，但是页表项越大这是肯定的。比如 1kb 大小的页就要有 10 位的地址。
无论物理地址还是虚拟地址，低 **log(页大小)**位都是偏移量，都是一致的。空间大小会是倍数的页大小。空间大小等于 2^地址位数；（虚拟空间对应虚拟地址，内存空间对应物理地址）

45.什么是 TLB？有了 TLB 之后具体的访问过程编程什么样子？如何计算平均查找时间？（如果已经知道了平均查找时间，如何计算命中率？）

答：TLB：转换检测缓冲区，是一个高速缓存，其功能类似于 cache，不同的是其里面缓存的是最近用过的页表项。

有了 TLB 的访问过程即是在指令运行的开始会先进行 TLB 检测，如果命中则可以不用去找内存直接得到物理地址；如果未命中，则分情况：其一是不发生缺页中断，那么找到页表项之后就先更新 TLB 再找物理地址；发生缺页中断，就去硬盘中放入相应的页，更新页表，然后重新开始执行指令（即从找 TLB 开始），但是更新页表之后目前的情况是不更新 TLB。

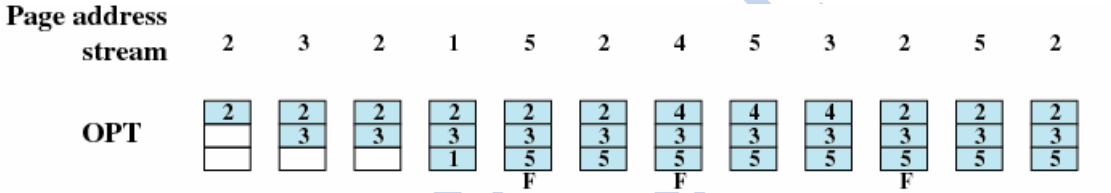
计算平均查找时间：首先应该清楚一个概念即 TLB 的命中率。所谓的 TLB 的命中率，其分母不应该是所有所寻找的页（因为有的页不在内存中，假设 TLB 的容量无限大，那么如果是这么算的话 TLB 的命中率也不可能达到百分百，但是实际上当 TLB 无限大的时候，理论上就应该有百分百的命中率）。因此 TLB 的命中率应该是 TLB 实际命中次数/页表项中存在位为 1 的页表项个数。

平均查找时间即是分为 TLB 是否命中的情况，而当 TLB 未命中时又应该分为是否发生缺页中断。

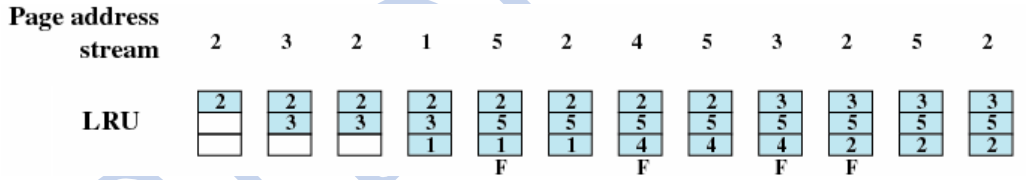
算命中率就是算平均查找时间的逆过程。

46.不同的内存访问次序对缺页率的影响？

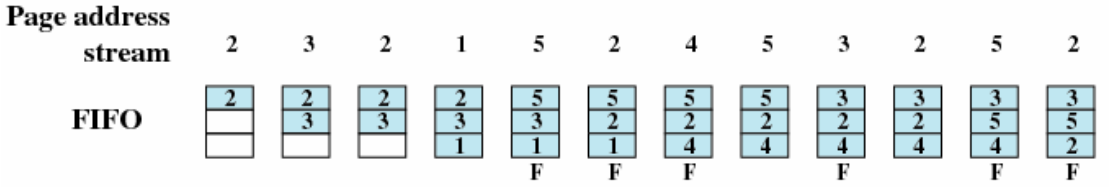
答：OPT：选择替换下次访问距离当前时间最长的页



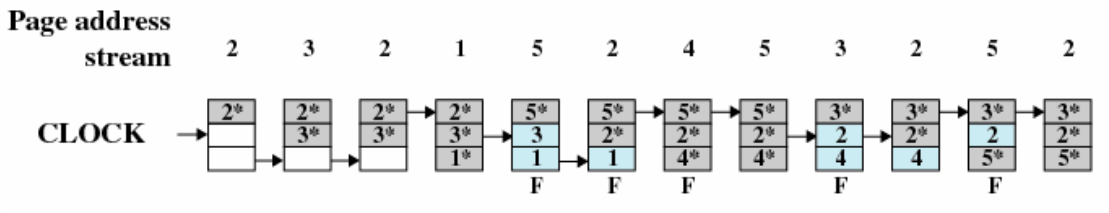
LRU：替换主存中上次使用距离当前最远的页



FIFO：先被换进的最先被换出



Clock：当某页首次装入主存中时，该帧的使用位设置为 1；当该页随后被访问到时，他的使用位也被置为 1。当需要替换一页时，操作系统扫描缓冲区，以查找使用位被置为 0 的一帧，每当遇到一个使用位为 1 的帧时，操作系统就将该位重新置为 0。当一页被替换时，该指针被设置为指向缓冲区中的下一帧。



47.什么是交换技术？好处是什么？代价是什么？

答：交换技术用于将额外的就绪进程加载到主存，从而保持处理器处于工作状态，但就交换技术本身而言，其就是一个 IO 操作。

好处：能够保持处理器处于工作状态，提高效率

代价：本身是 IO 操作，速度慢

48.文件系统如何解决同名问题？

答：使用（多级）目录。

用树形结构目录来解决同名问题。即文件名不一定要唯一，只要文件路径是唯一的就够了。而拼写完整路径也是比较困难的，因此对于进程或是交互用户，总有一个当前路径与之相关联，通常称为工作目录。

49.什么是系统抖动，如何避免系统抖动？（讲到如何避免系统抖动首先一定要讲为什么会产生系统抖动，然后才讲怎么避免）

答：处理器的大部分时间都用于交换块，而不是执行指令。

由于内存块正好在将要被用到之前换出，操作系统不得不很快把它取回来，从而引起系统抖动。

可以使用基于局部性原理的推理避免系统抖动。假设在很短的时间内仅需要访问进程的一部分块是合理的，同时对在不远的将来可能会访问的块进行猜测。

50.有哪些不同的内存管理方式？（6 个）哪些管理方式会产生系统抖动？

答：固定分区、动态分区、简单分页、简单分段、虚拟内存分页、虚拟内存分段；

虚拟内存分页和虚拟内存分段可能产生系统抖动。

51.在请求式分页管理系统中，如何在不同的内存块数、不同的调度算法下计算缺页率（需要计算过程）（注意：在进程运行的开始阶段那几块空的也算进缺页率）

答：P257 图要会画，参考课后习题 8.4，根据页框数确定图的绘制

缺页中断：就是要访问的页不在主存，需要操作系统将其调入主存后再进行访问。

根据图和页数

课后习题 8.6:

8.6 一个进程在磁盘上包含 8 个虚拟页，在主存中固定分配给 4 个页帧。发生如下顺序的页访问：

1,0,2,2,1,7,0,1,2,0,3,0,4,5,1,5,2,4,5,6,7,6,7,2,4,2,7,3,3,2,3

a.如果使用 LRU 替换策略，给出相继驻留在这 4 个页帧中的页。计算主存的命中率。假设这些帧最初是空的。

b.如果使用 FIFO 策略，重复问题（a）。

c.比较使用这两种策略的命中率。解释为什么这个特殊的访问顺序，使用 FIFO 的效率接近于 LRU。

解答

a:LRU：命中率=16/33

1	0	2	2	1	7	6	7	0	1	2	0	3	0	4	5	1	5	2	4	5	6	7	6	7	2	4	2	7	3	3	2	3
1	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4	4	4	4	4	2	2	2	2	2	2	2	2
-	0	0	0	0	0	6	6	6	6	2	2	2	2	2	5	5	5	5	5	5	5	5	5	5	5	4	4	4	4	4	4	4
-	-	2	2	2	2	2	2	0	0	0	0	0	0	0	0	0	0	2	2	2	2	7	7	7	7	7	7	7	7	7	7	7
-	-	-	-	-	7	7	7	7	7	7	7	3	3	3	3	1	1	1	1	1	6	6	6	6	6	6	6	6	3	3	3	3
F	F	F			F	F		F		F		F		F	F	F		F			F	F			F	F			F			

b:FIFO：命中率=16/33

1	0	2	2	1	7	6	7	0	1	2	0	3	0	4	5	1	5	2	4	5	6	7	6	7	2	4	2	7	3	3	2	3
1	1	1	1	1	1	6	6	6	6	6	6	6	6	4	4	4	4	4	4	4	6	6	6	6	6	6	6	6	6	6	2	2
-	0	0	0	0	0	0	0	0	1	1	1	1	1	1	5	5	5	5	5	5	5	7	7	7	7	7	7	7	7	7	7	7
-	-	2	2	2	2	2	2	2	2	2	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4	4	4
-	-	-	-	-	7	7	7	7	7	7	7	3	3	3	3	3	3	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3
F	F	F			F	F			F		F	F		F	F	F		F			F	F				F			F		F	

c:这两种策略对这个特殊的页轨迹（执行顺序）是等效的。

52.系统安全状态和死锁的关系？（会不会一定产生死锁）（和 25 题重复）

答：安全状态不会产生死锁

不安全状态不是死锁状态，进入不安全状态的系统有可能进入死锁

死锁一定是不安全状态

53.文件的五种不同的逻辑组织形式，各有什么特点？哪些适合文件的记录频繁增加操作？哪些不适合？（或者还有，哪些适合经常查询的操作？哪些不适合？哪些适合经常进行修改的操作？哪些不适合？）

答：①堆：数据按他们到达的顺序被采集，每个记录由一串数据组成。当保存的数据大小和结构不同时，这种类型的文件空间使用情况很好，能较好的用于穷举搜索，且易于修改。

顺序文件：在这类文件中，每个记录都使用一种固定的格式。所有记录都具有相同的长度，并且由相同数目、长度固定的域按特定的顺序组成。

②顺序文件通常用于批处理的应用中，并且如果这类应用设计到对所有记录的处理，则顺序文件通常是最佳的。对于查询或更新记录的交互式应用，顺序文件表现出很差的性能。

③索引顺序文件：索引顺序文件保留了顺序文件的关键特征，它还增加了两个特征，用于支持随机访问的文件索引和溢出文件。索引提供了快速接近目标记录的查找能力。

④索引文件：为了基于其他的属性而不是关键域搜索一条记录时，需要一种采用多索引的结构，每种可能成为搜索条件的域都有一个索引。可以使用长度可变的记录。当往主文件增加一条新记录时，索引文件必须全部更新。所以，索引文件大多用于对于信息的及时性要求比较严格并且很少会对所有数据进行处理的应用。

⑤直接或散列文件：直接文件或散列文件开发直接访问磁盘中任何一个地址已知的块的能

力。直接文件使用基于关键字的散列，常在要求快速访问时使用，并且记录的长度是固定的，通常一次只访问一条记录。
适合文件频繁的记录和增加的是堆和直接或散列文件
适合频繁查询的是索引文件
适合经常修改的是索引文件

54.什么是缓冲技术？为什么要使用缓冲技术？缓冲技术如何减少程序的运行时间？最多可以减少到多少？（作业题）
答：在输入请求发出前就开始执行输入传递，在输出请求发出一段时间之后才开始执行输出传送
为什么采用缓冲技术？
可能出现单进程死锁；
缓冲是用来平滑 I/O 需求的峰值的一种技术，但当进程的平均需求大于 I/O 设备的服务能力时，缓冲再多也不能让 I/O 设备与这个进程一直并驾齐驱。

55.什么是寻道时间，旋转延迟，存取时间，传送时间？如何计算？
答：寻道时间：磁头定位到磁道所需要的时间
旋转延迟：磁头到达扇区开始位置的时间
存取时间：信息存储到设备整个过程所需的时间（如下）
传送时间：传送所需的时间
平均存取时间为 Ta = Ts + 1/2r + b/rN　ts 表示寻道时间，r 表示速度，b 表示传输数据大小，N 表示一个磁道的字节数。

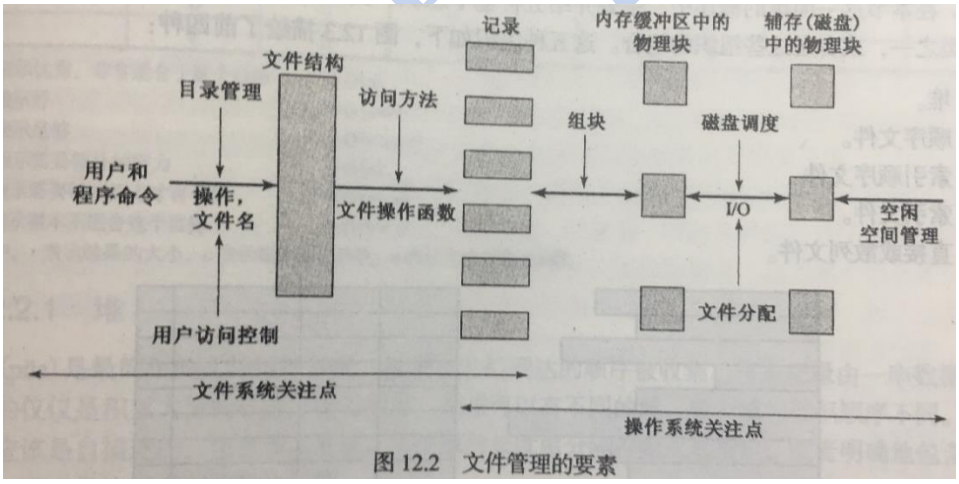
56.如何访问一个文件？其过程是什么？（第十二章 12.2 的图要理解清楚，包括读和写)
答：


图 12.2　文件管理的要素
用户和程序发起命令和指定文件名，经过访问控制判断具有权限，文件系统将命令转换成特定的文件操作命令，文件名通过目录转化成路径名，文件系统根据文件的逻辑类型判断正确的访问方法并执行响应的文件操作函数获取记录。操作系统将通过磁盘调度将辅存的（带有文件信息的）物理块输入到内存缓冲区中，然后组织成记录提供给用户。

57.磁盘调度有哪几种调度类型？有哪些磁盘调度算法？最短服务时间优先算法的调度原则是什么？
答：随机调度和策略调度。

FIFO：先进先出，顺序处理队列项目
PRI：优先级
LIFO：后进先出
SSTF：最短服务时间优先：选择使磁头臂从当前位置开始移动最少的磁盘 io 请求
SCAN：磁头臂仅沿一个方向移动，在途中完成请求，直到最后或者该方向无请求停止，反向移动；
C-SCAN：与 SCAN 唯一不同，在回来途中不服务
N-step-Scan：队列分成长度为 n 的队列，每次用 SCAN 扫描一个子队列，扫描期间新请求进入另一个队列
FSCAN：分成两个子队列，开始时所有都在一个队列，扫描时，新请求进入另一个队列

58.文件的物理存储分配有哪些方式？这些方式分别需要哪些辅助信息？（三个，FAT 表里面）(五种逻辑结构，三种物理结构，这里是物理结构)
答：①连续分配：在创建文件时，给文件分配一组连续的块。需要文件名、起始块位置、长度。(块的数量)
②链式分配：链式分配基于单个的块。链中的每一块都包含指向下一块的指针。局部性原理不再适用，适合于顺序处理的顺序文件。需要文件名、起始块位置、长度。
③索引分配：每个文件在文件分配表中有一个一级索引。分配给该文件的每个分区都在索引中有一个表项。只需要文件名和索引块位置。

59.什么是多道程序设计？在多道程序设计环境下，cpu 和 I/O 设备如何同步运行？（并发工作的能力）
答：多道程序设计是在计算机内存中同时存放几道相互独立的程序，使它们在管理程序控制之下，相互穿插的运行。 两个或两个以上程序在计算机系统中同处于开始到结束之间的状态。
当一个作业要求等待 IO 时，处理器可以切换到另一个可能并不在等待 IO 的作业，直到 IO 完成再切回该作业处理。

60.什么是记录的成组与分解?
答：组块：块是与二级存储进行 I/O 操作的基本单位，为了执行 I/O 操作，记录必须组织成块。
分解：用户使用时把读取的一块信息中分离出所需的记录，这就是记录的分解。

61.什么是假脱机系统?
答：Simultaneous Peripheral Operation On-Line，外部设备联机并行操作，是关于慢速字符设备如何与计算机主机交换信息的一种技术。实际上是一种外围设备同时联机操作技术,又称为排队转储技术
SPOOLing 技术的特点:
(1)提高了 I/O 速度.从对低速 I/O 设备进行的 I/O 操作变为对输入井或输出井的操作,如同脱机操作一样,提高了 I/O 速度,缓和了 CPU 与低速 I/O 设备速度不匹配的矛盾.
(2)设备并没有分配给任何进程.在输入井或输出井中,分配给进程的是一存储区和建立一张 I/O 请求表.
(3)实现了虚拟设备功能.多个进程同时使用一独享设备,而对每一进程而言,都认为自己独占这一设备,不过,该设备是逻辑上的设备。