第一章

1．设计现代OS的主要目标是什么？

答：（1）有效性 （2）方便性 （3）可扩充性 （4）开放性

2．OS的作用可表现在哪几个方面？

答：（1）OS作为用户与计算机硬件系统之间的接口

（2）OS作为计算机系统资源的管理者

（3）OS实现了对计算机资源的抽象

3．为什么说OS实现了对计算机资源的抽象？

答：OS首先在裸机上覆盖一层I/O设备管理软件，实现了对计算机硬件操作的第一层次抽象；在第一层软件上再覆盖文件管理软件，实现了对硬件资源操作的第二层次抽象。OS 通过在计算机硬件上安装多层系统软件，增强了系统功能，隐藏了对硬件操作的细节，由它们共同实现了对计算机资源的抽象。

4．试说明推动多道批处理系统形成和发展的主要动力是什么？

答：主要动力来源于四个方面的社会需求与技术发展：

（1）不断提高计算机资源的利用率；

（2）方便用户；

（3）器件的不断更新换代；

（4）计算机体系结构的不断发展。

5．何谓脱机I/O和联机I/O?

答：脱机I/O 是指事先将装有用户程序和数据的纸带或卡片装入纸带输入机或卡片机，在外围机的控制下，把纸带或卡片上的数据或程序输入到磁带上。该方式下的输入输出由外围机控制完成，是在脱离主机的情况下进行的。

而联机I/O方式是指程序和数据的输入输出都是在主机的直接控制下进行的。

6．试说明推动分时系统形成和发展的主要动力是什么？

答：推动分时系统形成和发展的主要动力是更好地满足用户的需要。主要表现在：CPU 的分时使用缩短了作业的平均周转时间；人机交互能力使用户能直接控制自己的作业；主机的共享使多用户能同时使用同一台计算机，独立地处理自己的作业。

7．实现分时系统的关键问题是什么？应如何解决？

答：关键问题是当用户在自己的终端上键入命令时，系统应能及时接收并及时处理该命令，在用户能接受的时延内将结果返回给用户。

解决方法：针对及时接收问题，可以在系统中设置多路卡，使主机能同时接收用户从各个终端上输入的数据；为每个终端配置缓冲区，暂存用户键入的命令或数据。针对及时处理问题，应使所有的用户作业都直接进入内存，并且为每个作业分配一个时间片，允许作业只在自己的时间片内运行，这样在不长的时间内，能使每个作业都运行一次。

8．为什么要引入实时OS？

答：实时操作系统是指系统能及时响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时任务协调一致地运行。引入实时OS 是为了满足应用的需求，更好地满足实时控制领域和实时信息处理领域的需要。

9．什么是硬实时任务和软实时任务？试举例说明。

答：硬实时任务是指系统必须满足任务对截止时间的要求，否则可能出现难以预测的结果。举例来说，运载火箭的控制等。

软实时任务是指它的截止时间并不严格，偶尔错过了任务的截止时间，对系统产生的影响不大。举例：网页内容的更新、火车售票系统。

10．在8位微机和16位微机中，占据了统治地位的是什么操作系统？
答：单用户单任务操作系统，其中最具代表性的是CP/M和MS-DOS.
11．试列出Windows OS 中五个主要版本，并说明它们分别较之前一个版本有何改进。
答：
（1）Microsoft Windows 1.0是微软公司在个人电脑上开发图形界面的首次尝试。
（2）Windows 95是混合的16位/32位系统，第一个支持32位。带来了更强大、更稳定、更实用的桌面图形用户界面，结束了桌面操作系统间的竞争。
（3）Windows 98是微软公司的混合16位/32位Windows 操作系统，改良了硬件标准的支持，革新了内存管理，是多进程操作系统。
（4）Windows XP是基于Windows 2000的产品，拥有新用户图形界面月神Luna。简化了用户安全特性，整合了防火墙。
（5）Windows Vista 包含了上百种新功能；特别是新版图形用户界面和Windows Aero全新界面风格、加强的搜寻功能（Windows Indexing Service）、新媒体创作工具以及重新设计的网络、音频、输出（打印）和显示子系统。。
12．试从交互性、及时性以及可靠性方面，将分时系统与实时系统进行比较。
答：（1）及时性：实时信息处理系统对实时性的要求与分时系统类似，都是以人所能接受的等待时间来确定；而实时控制系统的及时性，是以控制对象所要求的开始截止时间或完成截止时间来确定的，一般为秒级到毫秒级，甚至有的要低于100微妙。
（2）交互性：实时信息处理系统具有交互性，但人与系统的交互仅限于访问系统中某些特定的专用服务程序。不像分时系统那样能向终端用户提供数据和资源共享等服务。
（3）可靠性：分时系统也要求系统可靠，但相比之下，实时系统则要求系统具有高度的可靠性。因为任何差错都可能带来巨大的经济损失，甚至是灾难性后果，所以在实时系统中，往往都采取了多级容错措施保障系统的安全性及数据的安全性。
13．OS有哪几大特征？其最基本的特征是什么？
答：并发性、共享性、虚拟性和异步性四个基本特征；最基本的特征是并发性。
14．处理机管理有哪些主要功能？它们的主要任务是什么？
答：处理机管理的主要功能是：进程管理、进程同步、进程通信和处理机调度；
进程管理：为作业创建进程，撤销已结束进程，控制进程在运行过程中的状态转换。
进程同步：为多个进程（含线程）的运行_____进行协调。
通信：用来实现在相互合作的进程之间的信息交换。
处理机调度：
（1）作业调度。从后备队里按照一定的算法，选出若干个作业，为他们分配运行所需的资源（首选是分配内存）。
（2）进程调度：从进程的就绪队列中，按照一定算法选出一个进程，把处理机分配给它，并设置运行现场，使进程投入执行。
15．内存管理有哪些主要功能?他们的主要任务是什么？

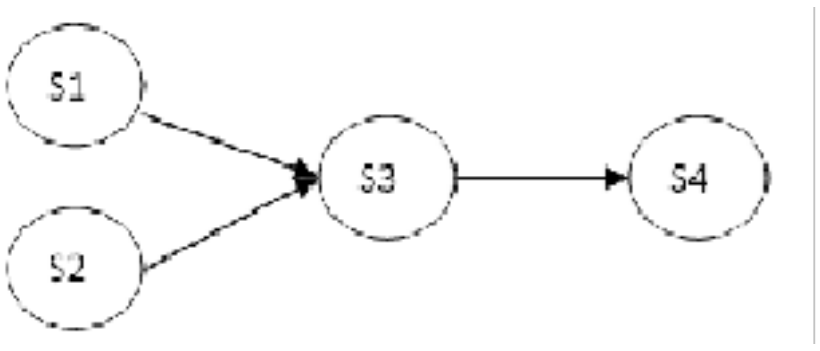
答：内存管理的主要功能有：内存分配、内存保护、地址映射和内存扩充。
内存分配：为每道程序分配内存。
内存保护：确保每道用户程序都只在自己的内存空间运行，彼此互不干扰。
地址映射：将地址空间的逻辑地址转换为内存空间与对应的物理地址。
内存扩充：用于实现请求调用功能，置换功能等。
16．设备管理有哪些主要功能？其主要任务是什么？

答：主要功能有：缓冲管理、设备分配和设备处理以及虚拟设备等。
主要任务：完成用户提出的I/O 请求，为用户分配I/O 设备；提高CPU 和I/O 设备的利用率；提高I/O速度；以及方便用户使用I/O设备.
17．文件管理有哪些主要功能？其主要任务是什么？
答：文件管理主要功能：文件存储空间的管理、目录管理、文件的读/写管理和保护。
文件管理的主要任务：管理用户文件和系统文件，方便用户使用，保证文件安全性。
18．是什么原因使操作系统具有异步性特征？
答：操作系统的异步性体现在三个方面：一是进程的异步性，进程以人们不可预知的速度向前推进，二是程序的不可再现性，即程序执行的结果有时是不确定的，三是程序执行时间的不可预知性，即每个程序何时执行，执行顺序以及完成时间是不确定的。
19．模块接口法存在哪些问题？可通过什么样的途径来解决？
答：（1）模块接口法存在的问题：①在OS设计时，各模块间的接口规定很难满足在模块完成后对接口的实际需求。②在OS 设计阶段，设计者必须做出一系列的决定，每一个决定必须建立在上一个决定的基础上。但模块化结构设计的各模块设计齐头并进，无法寻找可靠的顺序，造成各种决定的无序性，使程序设计人员很难做到设计中的每一步决定都建立在可靠的基础上，因此模块接口法被称为“无序模块法”。
（2）解决途径：将模块接口法的决定顺序无序变有序，引入有序分层法。
20．在微内核OS中，为什么要采用客户/服务器模式？
答：C/S 模式具有独特的优点：⑴数据的分布处理和存储。⑵便于集中管理。⑶灵活性和可扩充性。⑷易于改编应用软件。
21．试描述什么是微内核OS。
答：1）足够小的内核 2）基于客户/服务器模式
3）应用机制与策略分离原理 4）采用面向对象技术。
22．在基于微内核结构的OS中，应用了哪些新技术？
答：在基于微内核结构的OS 中，采用面向对象的程序设汁技术。
23．何谓微内核技术？在微内核中通常提供了哪些功能？
答：把操作系统中更多的成分和功能放到更高的层次（即用户模式）中去运行，而留下一个尽量小的内核，用它来完成操作系统最基本的核心功能，称这种技术为微内核技术。在微内核中通常提供了进程（线程）管理、低级存储器管理、中断和陷入处理等功能。
24．微内核操作系统具有哪些优点？它为何能有这些优点？
答：1）提高了系统的可扩展性
2）增强了系统的可靠性
3）可移植性
4）提供了对分布式系统的支持
5）融入了面向对象技术

第二章

1. 什么是前趋图？为什么要引入前趋图？
答：前趋图(Precedence Graph)是一个有向无循环图，记为DAG(Directed Acyclic Graph)，用于描述进程之间执行的前后关系。
2. 画出下面四条语句的前趋图:
S1=a：=x+y; S2=b：=z+1; S3=c：=a - b; S4=w：=c+1;

答：其前趋图为：



3. 什么程序并发执行会产生间断性特征？

答：程序在并发执行时，由于它们共享系统资源，为完成同一项任务需要相互合作，致使这些并发执行的进程之间，形成了相互制约关系，从而使得进程在执行期间出现间断性。

4．程序并发执行时为什么会失去封闭性和可再现性？

答：程序并发执行时，多个程序共享系统中的各种资源，因而这些资源的状态由多个程序改变，致使程序运行失去了封闭性，也会导致其失去可再现性。

5．在操作系统中为什么要引入进程概念？它会产生什么样的影响?

答：为了使程序在多道程序环境下能并发执行，并对并发执行的程序加以控制和描述，在操作系统中引入了进程概念。

影响: 使程序的并发执行得以实行。

6．试从动态性，并发性和独立性上比较进程和程序?

答：(1)动态性是进程最基本的特性，表现为由创建而产生，由调度而执行，因得不到资源而暂停执行，由撤销而消亡。进程有一定的生命期，而程序只是一组有序的指令集合，是静态实体。

(2)并发性是进程的重要特征，同时也是OS 的重要特征。引入进程的目的正是为了使其程序能和其它进程的程序并发执行，而程序是不能并发执行的。

(3)独立性是指进程实体是一个能独立运行的基本单位，也是系统中独立获得资源和独立调度的基本单位。对于未建立任何进程的程序，不能作为独立单位参加运行。

7．试说明PCB 的作用，为什么说PCB 是进程存在的惟一标志？

答：PCB 是进程实体的一部分，是操作系统中最重要的记录型数据结构。作用是使一个在多道程序环境下不能独立运行的程序，成为一个能独立运行的基本单位，成为能与其它进程并发执行的进程。OS是根据PCB对并发执行的进程进行控制和管理的。

8．试说明进程在三个基本状态之间转换的典型原因。

答：　（1）就绪状态→执行状态：进程分配到CPU资源

（2）执行状态→就绪状态：时间片用完

（3）执行状态→阻塞状态：I/O请求

（4）阻塞状态→就绪状态：I/O完成

9．为什么要引入挂起状态？该状态有哪些性质？

答：引入挂起状态处于五种不同的需要: 终端用户需要，父进程需要，操作系统需要，对换

需要和负荷调节需要。处于挂起状态的进程不能接收处理机调度。

10．在进行进程切换时，所要保存的处理机状态信息有哪些?

答：进行进程切换时，所要保存的处理机状态信息有：

（1）进程当前暂存信息

（2）下一指令地址信息

（3）进程状态信息

（4）过程和系统调用参数及调用地址信息。

11．试说明引起进程创建的主要事件。
答：引起进程创建的主要事件有：用户登录、作业调度、提供服务、应用请求。
12．试说明引起进程被撤销的主要事件。
答：引起进程被撤销的主要事件有：正常结束、异常结束（越界错误、保护错、非法指令、特权指令错、运行超时、等待超时、算术运算错、I/O 故障）、外界干预（操作员或操作系统干预、父进程请求、父进程终止）。
13．在创建一个进程时所要完成的主要工作是什么？
答：
（1）OS 发现请求创建新进程事件后，调用进程创建原语Creat()；
（2）申请空白PCB；
（3）为新进程分配资源；
（4）初始化进程控制块；
（5）将新进程插入就绪队列.
14．在撤销一个进程时所要完成的主要工作是什么？
答：
（1）根据被终止进程标识符，从PCB 集中检索出进程PCB，读出该进程状态。
（2）若被终止进程处于执行状态，立即终止该进程的执行，置调度标志真，指示该进程被终止后重新调度。
（3）若该进程还有子进程，应将所有子孙进程终止，以防它们成为不可控进程。
（4）将被终止进程拥有的全部资源，归还给父进程，或归还给系统。
（5）将被终止进程PCB 从所在队列或列表中移出，等待其它程序搜集信息。
15．试说明引起进程阻塞或被唤醒的主要事件是什么？
答：a. 请求系统服务；b. 启动某种操作；c. 新数据尚未到达；d. 无新工作可做.
16．进程在运行时存在哪两种形式的制约？并举例说明之。
答：
（1）间接相互制约关系。举例：有两进程A 和B，如果A 提出打印请求，系统已把唯一的一台打印机分配给了进程B，则进程A 只能阻塞；一旦B 释放打印机，A 才由阻塞改为就绪。
（2）直接相互制约关系。举例：有输入进程A 通过单缓冲向进程B 提供数据。当缓冲空时，计算进程因不能获得所需数据而阻塞，当进程A 把数据输入缓冲区后，便唤醒进程B；反之，当缓冲区已满时，进程A 因没有缓冲区放数据而阻塞，进程B 将缓冲区数据取走后便唤醒A。
17．为什么进程在进入临界区之前应先执行“进入区”代码？而在退出前又要执行“退出区”代码？
答：为了实现多个进程对临界资源的互斥访问，必须在临界区前面增加一段用于检查欲访问的临界资源是否正被访问的代码，如果未被访问，该进程便可进入临界区对资源进行访问，并设置正被访问标志，如果正被访问，则本进程不能进入临界区，实现这一功能的代码为"

进入区"代码；
在退出临界区后，必须执行"退出区"代码，用于恢复未被访问标志，使其它进程能再访问此临界资源。
18. 同步机构应遵循哪些基本准则？为什么？
答：同步机构应遵循的基本准则是：空闲让进、忙则等待、有限等待、让权等待

原因：为实现进程互斥进入自己的临界区。

19. 试从物理概念上说明记录型信号量wait 和signal。

答：wait(S)：当S.value>0 时，表示目前系统中这类资源还有可用的。执行一次wait 操作，意味着进程请求一个单位的该类资源，使系统中可供分配的该类资源减少一个，因此描述为S.value:=S.value-1；当S.value<0时，表示该类资源已分配完毕，进程应调用block 原语自我阻塞，放弃处理机，并插入到信号量链表S.L中。

signal(S)：执行一次signal操作，意味着释放一个单位的可用资源，使系统中可供分配的该类资源数增加一个，故执行S.value:=S.value+1 操作。若加1 后S.value≤0，则表示在该信号量链表中，仍有等待该资源的进程被阻塞，因此应调用wakeup 原语，将S.L 链表中的第一个等待进程唤醒。

20. 你认为整型信号量机制是否完全遵循了同步机构的四条准则？

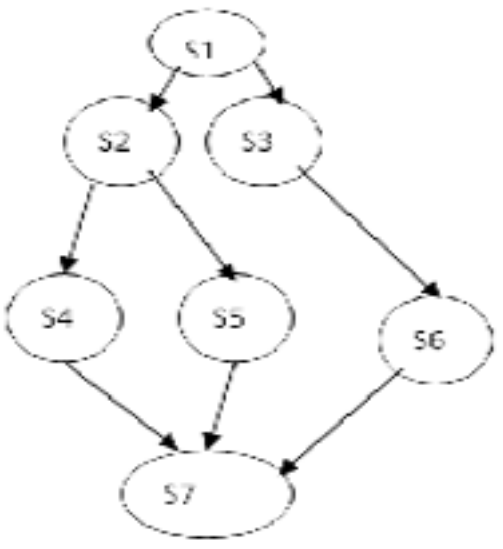
答：整型信号量机制不完全遵循同步机制的四条准则，它不满足“让权等待”准则。

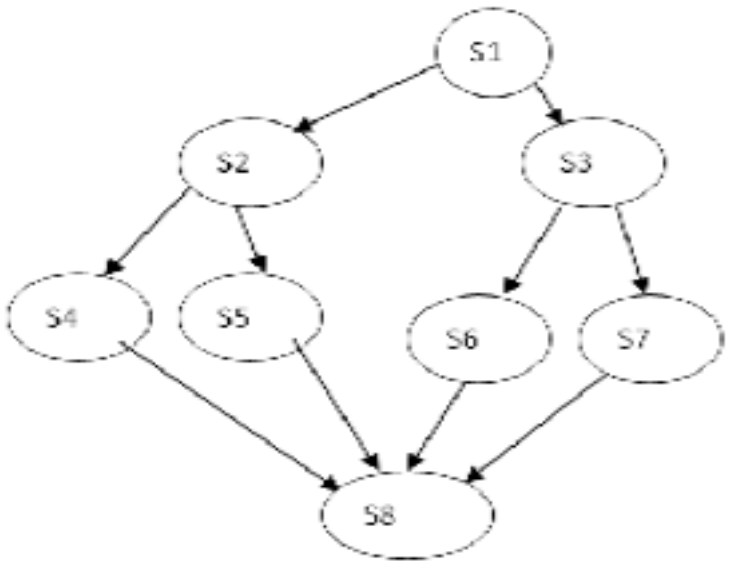
21. 如何利用信号量机制来实现多个进程对临界资源的互斥访问？并举例说明之。

答：为使多个进程互斥访问某临界资源，只需为该资源设置一互斥信号量mutex，并设其初值为1，然后将各进程访问该资源的临界区CS置于wait(mutex)和signal(mutex)操作之间即可。这样，每个欲访问该临界资源的进程在进入临界区之前，都要先对mutex 执行wait 操作，若该资源此刻未被访问，本次wait 操作必然成功，进程便可进入自己的临界区，这时若再有其他进程也欲进入自己的临界区，此时由于对mutex 执行wait操作定会失败，因而该进程阻塞，从而保证了该临界资源能被互斥访问。当访问临界资源的进程退出临界区后，应对mutex执行signal 操作，释放该临界资源。利用信号量实现进程互斥的进程描述如下：

```
Var mutex: semaphore:=1;
   begin
   parbegin
     process 1: begin
            repeat
              wait(mutex);
              critical section
              signal(mutex);
              remainder seetion
              until false;
              end
     process 2: begin
            repeat
              wait(mutex);
              critical section
              signal(mutex);
              remainder section
              until false;
              end
   parend
```

22. 试写出相应的程序来描述图2-17所示的前驱图。





答：（a）Var a, b, c, d, e, f, g, h; semaphore:= 0, 0, 0, 0, 0, 0, 0, 0;

```
begin
  parbegin
     begin S1; signal(a); signal(b); end;
     begin wait(a); S2; signal(c); signal(d); end;
     begin wait(b); S3; signal(e); end;
     begin wait(c); S4; signal(f); end;
     begin wait(d); S5; signal(g); end;
     begin wait(e); S6; signal(h); end;
     begin wait(f); wait(g); wait(h); S7; end;
  parend
end
```

（b）Var a, b, c, d, e, f, g, h,i,j; semaphore:= 0, 0, 0, 0, 0, 0, 0,0,0, 0;

```
begin
  parbegin
     begin S1; signal(a); signal(b); end;
     begin wait(a); S2; signal(c); signal(d); end;
     begin wait(b); S3; signal(e); signal(f); end;
     begin wait(c); S4; signal(g); end;
     begin wait(d); S5; signal(h); end;
     begin wait(e); S6; signal(i); end;
     begin wait(f); S7; signal(j); end;
     begin wait(g);wait(h); wait(i); wait(j); S8; end;
  parend
end
```

23. 在生产者消费者问题中，如果缺少了signal(full)或signal(empty),对执行结果有何影响？

答：

如果缺少signal(full)，那么表明从第一个生产者进程开始就没有改变信号量full 值，即使缓冲池产品已满，但full 值还是0，这样消费者进程执行wait(full)时认为缓冲池是空而取不到产品，消费者进程一直处于等待状态。

如果缺少signal(empty)，在生产者进程向n个缓冲区投满产品后消费者进程才开始从中取产品，这时empty=0，full=n，那么每当消费者进程取走一个产品empty 值并不改变，

直到缓冲池取空了，empty 值也是0，即使目前缓冲池有n 个空缓冲区，生产者进程要想再往缓冲池中投放产品也会因为申请不到空缓冲区被阻塞。

24．在生产消费者问题中，如果将两个wait 操作即wait(full)和wait(mutex)互换位置，或者将signal(mutex)与signal（full）互换位置，结果如何？

答：将wait(full)和wait(mutex)互换位置后，可能引起死锁。考虑系统中缓冲区全满时，若一生产者进程先执行了wait(mutex)操作并获得成功，则当再执行wait(empty)操作时，它将因失败而进入阻塞状态，它期待消费者进程执行signal(empty)来唤醒自己，在此之前，它不可能执行signal(mutex)操作，从而使试图通过执行wait(mutex)操作而进入自己的临界区的其他生产者和所有消费者进程全部进入阻塞状态，这样容易引起系统死锁。

若signal(mutex)和signal(full)互换位置后只是影响进程对临界资源的释放次序，而不会引起系统死锁，因此可以互换位置。

25．我们在为某一临界资源设置一把锁W，当W=1时表示关锁，当W=0时表示锁已打开。试写出开锁和关锁的原语，并利用他们实现互斥。

答：整型信号量：lock(W): while W=1 do no-op
W:=1;
unlock(W): W:=0;

记录型信号量：lock(W): W:=W+1;
if(W>1) then block(W, L)
unlock(W): W:=W-1;
if(W>0) then wakeup(W, L)

例子：

```
Var W:semaphore:=0;
      begin
          repeat
            lock(W);
            critical section
            unlock(W);
            remainder section
          until false;
       end
```

26．试修改下面生产者－消费者问题解法中的错误:

答： producer:

```
begin
  repeat
  …
  producer an item in nextp;
  wait(mutex);
  wait(full); /* 应为wait(empty),而且还应该在wait(mutex)的前面 */
  buffer(in):=nextp;
  /* 缓冲池数组游标应前移: in:=(in+1) mod n; */
  signal(mutex);
```

```
    /* signal(full); */
    until false;
  end
consumer:
  begin
    repeat
    wait(mutex);
    wait(empty); /* 应为wait(full),而且还应该在wait(mutex)的前面 */
    nextc:=buffer(out);
    out:=out+1; /* 考虑循环，应改为: out:=(out+1) mod n; */
    signal(mutex);/* signal(empty); */
    consumer item in nextc;
    until false;
        end
```

27. 试利用记录型信号量写出一个不会出现死锁的哲学家进餐问题的算法.

答：Var chopstick:array[0,…,4] of semaphore;

所有信号量均被初始化为1，第i 位哲学家的活动可描述为：

```
Repeat
    Wait(chopstick[i]);
    Wait(. chopstick[(i+1) mod 5]);
    …
    Ea.t ;
    …
    Signal(chopstick[i]);
    Signal(chopstick[(i+1) mod 5])
    Ea.t ;
    …
    Think;
  Until false;
```

28. 在测量控制系统中的数据采集任务，把所采集的数据送一单缓冲区；计算任务从该单缓冲中取出数据进行计算.试写出利用信号量机制实现两者共享单缓冲的同步算法。

答：

```
a. Var mutex, empty, full: semaphore:=1, 1, 0;
   gather:
      begin
          repeat
          ……
          gather data in nextp;
          wait(empty);
          wait(mutex);
          buffer:=nextp;
          signal(mutex);
```

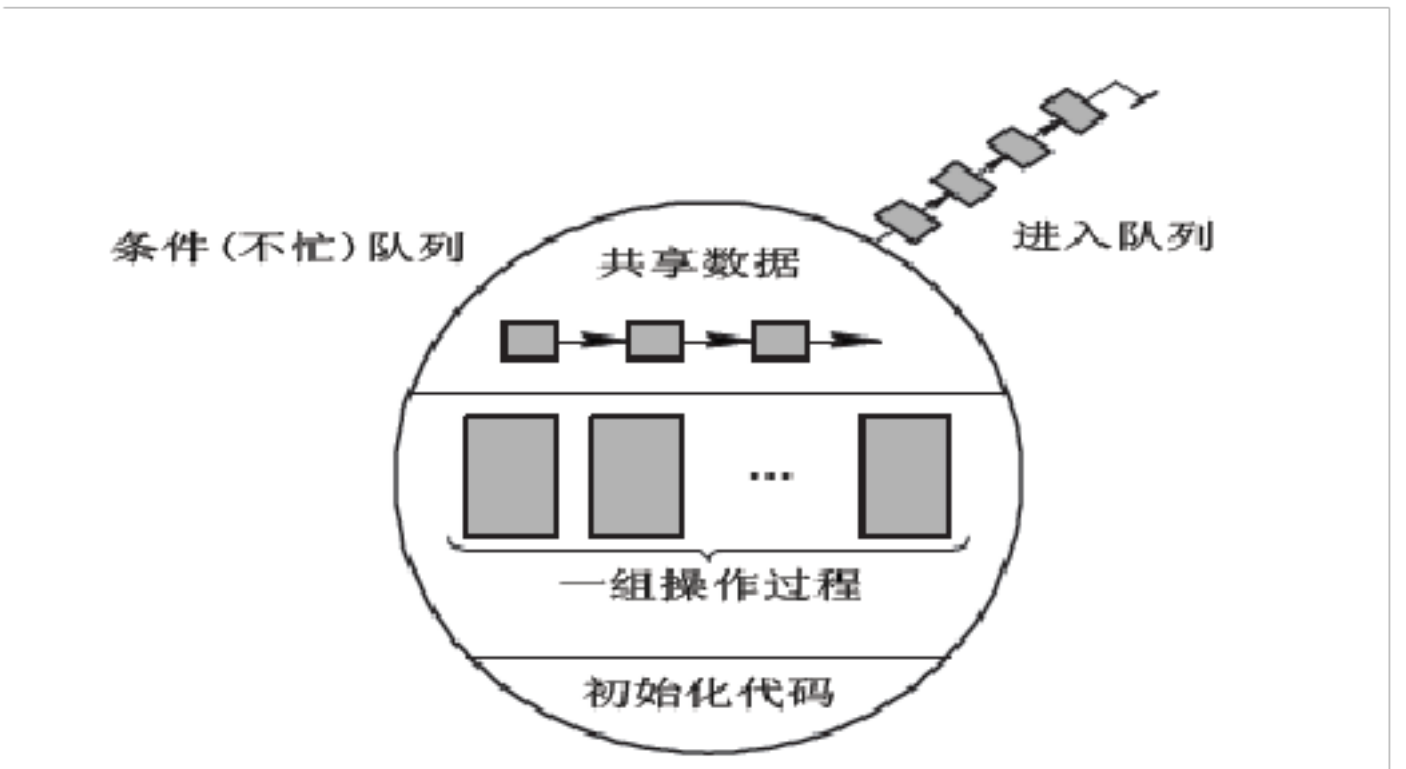
```
        signal(full);
        until false;
    end
compute:
  begin
        repeat
        ……
        wait(full);
        wait(mutex);
        nextc:=buffer;
        signal(mutex);
        signal(empty);
        compute data in nextc;
        until false;
    end
```

b. Var empty, full: semaphore:=1, 0;

```
    gather:
       begin
           repeat
           ……
           gather data in nextp;
           wait(empty);
           buffer:=nextp;
           signal(full);
           until false;
        end
    compute:
       begin
           repeat
            ……
            wait(full);
           nextc:=buffer;
           signal(empty);
           compute data in nextc;
           until false;
       end
```

29. 画图说明管程由哪几部分组成，为什么要引入条件变量？

答：管程由四部分组成：①管程的名称；②局部于管程内部的共享数据结构说明；③对该数据结构进行操作的一组过程；④对局部于管程内部的共享数据设置初始值的语句；



当一个进程调用了管程，在管程中时被阻塞或挂起，直到阻塞或挂起的原因解除，而在此期间，如果该进程不释放管程，则其它进程无法进入管程，被迫长时间地等待。为了解决这个问题，引入了条件变量condition。

30．如何利用管程来解决生产者与消费者问题？

答：首先建立一个管程，命名为ProclucerConsumer，包括两个过程：

（1）Put（item）过程。生产者利用该过程将自己生产的产品放到缓冲池，用整型变量count 表示在缓冲池中已有的产品数目，当count≥n 时，表示缓冲池已满，生产者须等待。

（2）get（item）过程。消费者利用该过程从缓冲池中取出一个产品，当count≤0 时，表示缓冲池中已无可取的产品，消费者应等待。

PC 管程可描述如下：

```
type producer-consumer =monitor
Var in,out,count:integer;
buffer:array[0,…,n-1]of item;
notfull，notempty:condition;
procedure entry dot(item)
    begin
      if count>=n then not full.wait;
      buffer(in):=nextp;
      in:=(in+1)mod n;
      count:=count+1;
      if notempty.queue then notempty.signal;
    end
procedure entry get(item)
    begin
      if count<=0 then not full.wait;
      nextc:=buffer(out);
      out:=(out+1)mod n;
      count:=count-1;
      if notfull.quene then notfull.signal;
```

```
      end
begin in:=out:=0;
count:=0
end
```

在利用管程解决生产者一消费者问题时，其中的生产者和消费者可描述为：

```
producer: begin
   pepeat
     produce an inem in nestp
     PC.put(item);
     until false;
   end
consumer: begin
   repeat
     PC.get(item);
     consume the item in enxtc;
     until false;
   end
```

31．什么是AND信号量？试利用AND信号量写出生产者一消费者问题的解法。

答：为解决并行带来的死锁问题，在wait 操作中引入AND 条件，其基本思想是将进程在整个运行过程中所需要的所有临界资源，一次性地全部分配给进程，用完后一次性释放。解决生产者－消费者问题可描述如下:

```
var mutex,empty,full: semaphore:=1,n,0;
    buffer: array[0,...,n-1] of item;
    in,out: integer:=0,0;
    begin
      parbegin
      producer: begin
      repeat
      …
      produce an item in nextp;
      …
      wait(empty);
      wait(s1,s2,s3,...,sn); //s1,s2,...,sn为执行生产者进程除empty 外其余的条件
      wait(mutex);
      buffer(in):=nextp;
      in:=(in+1) mod n;
      signal(mutex);
      signal(full);
      signal(s1,s2,s3,...,sn);
      until false;
    end
consumer: begin
          repeat
          wait(full);
```

```
            wait(k1,k2,k3,...,kn); //k1,k2,...,kn 为执行消费者进程除full 外其余的
条件
            wait(mutex);
            nextc:=buffer(out);
            out:=(out+1) mod n;
            signal(mutex);
            signal(empty);
            signal(k1,k2,k3,...,kn);
            consume the item in nextc;
            until false;
      end
   parend
end
```

32．什么是信号量集？试利用信号量集写出读者一写者问题的解法。

答：对AND信号量加以扩充，形成的信号量集合的读写机制。

解法：

```
Var RN integer;
        L,mx: semaphore:=RN,1;
         begin
            parbegin
               reader:begin
               repeat
               Swait(L,1,1);
               Swait(mx,1,1);
               …
               perform read operation;
               …
               Ssignal(L,1);
             until false
         end
              writer:begin
              repeat
                    Swait(mx,1,1;L,RN,0);
                    perform write operation;
                    Ssignal(mx,1);
                    until false
            end
     parend
end
```

33．试比较进程间的低级与高级通信工具。

答：用户用低级通信工具实现进程通信很不方便，效率低，通信对用户不透明，所有操作都必须由程序员来实现，而高级通信工具弥补了这些缺陷，用户直接利用操作系统提供的一组通信命令，高效地传送大量的数据。

34．当前有哪几种高级通信机制？

答：共享存储器系统、消息传递系统以及管道通信系统。

35．消息队列通信机制有哪几方面的功能？
答：（1）构成消息（2）发送消息（3）接收梢息（4）互斥与同步。
36．为什么要在OS 中引入线程？
答：在操作系统中引入线程，则是为了减少程序在并发执行时所付出的时空开销，使OS具有更好的并发性，提高CPU的利用率。进程是分配资源的基本单位,而线程则是系统调度的基本单位。
37．试说明线程具有哪些属性？
答：（1)轻型实体（2）独立调度和分派的基本单位（3）可并发执行（4）共享进程资源。
38．试从调度性，并发性，拥有资源及系统开销方面对进程和线程进行比较。
答：
（1）调度性。线程在OS 中作为调度和分派的基本单位，进程只作为资源拥有的基本单位。
（2）并发性。进程可以并发执行，一个进程的多个线程也可并发执行。
（3）拥有资源。进程始终是拥有资源的基本单位，线程只拥有运行时必不可少的资源，本身基本不拥有系统资源，但可以访问隶属进程的资源。
（4）系统开销。操作系统在创建、撤消和切换进程时付出的开销显著大于线程。
39．为了在多线程OS 中实现进程之间的同步与通信，通常提供了哪几种同步机制？
答：同步功能可以控制程序流并访问共享数据，从而并发执行多个线程。共有四种同步模型：互斥锁、读写锁、条件变量和信号。
40．用于实现线程同步的私用信号量和公用信号量之间有何差别？
答：
（1）私用信号量。当某线程需利用信号量实现同一进程中各线程之间的同步时，可调用创建信号量的命令来创建一个私用信号量，其数据结构存放在应用程序的地址空间中。
（2）公用信号量。公用信号量是为实现不同进程间或不同进程中各线程之间的同步而设置的。其数据结构是存放在受保护的系统存储区中，由OS为它分配空间并进行管理。
41．何谓用户级线程和内核支持线程？
答：
（1）用户级线程：仅存在于用户空间中的线程，无须内核支持。这种线程的创建、撤销、线程间的同步与通信等功能，都无需利用系统调用实现。用户级线程的切换通常发生在一个应用进程的诸多线程之间，同样无需内核支持。
（2）内核支持线程：在内核支持下运行的线程。无论是用户进程中的线程，还是系统线程中的线 程，其创建、撤销和切换等都是依靠内核，在内核空间中实现的。在内核空间里还为每个内核支持线程设置了线程控制块，内核根据该控制块感知某线程的存在并实施控制。
42．试说明用户级线程的实现方法。
答：用户级线程是在用户空间中的实现的，运行在“运行时系统”与“内核控制线程”的中间系统上。运行时系统用于管理和控制线程的函数的集合。内核控制线程或轻型进程LWP可通过系统调用获得内核提供服务，利用LWP进程作为中间系统。
43．试说明内核支持线程的实现方法。
答：系统在创建新进程时，分配一个任务数据区PTDA，其中包括若干个线程控制块TCB空间。创建一个线程分配一个TCB，有关信息写入TCB，为之分配必要的资源。当PTDA中的TCB 用完，而进程又有新线程时，只要所创建的线程数目未超过系统允许值，系统可在为之分配新的TCB；在撤销一个线程时，也应回收线程的所有资源和TCB。

第三章

第三章 处理机调度与死锁

1．高级调度与低级调度的主要任务是什么？为什么要引入中级调度？

答：高级调度的主要任务是根据某种算法，把外存上处于后备队列中的那些作业调入内存。

低级调度是保存处理机的现场信息，按某种算法先取进程，再把处理器分配给进程。

引入中级调度的主要目的是为了提高内存利用率和系统吞吐量。使那些暂时不能运行的进程不再占用内存资源，将它们调至外存等待，把进程状态改为就绪驻外存状态或挂起状态。

2．何谓作业、作业步和作业流？

答：作业包含通常的程序和数据，还配有作业说明书。系统根据该说明书对程序的运行进行控制。批处理系统中是以作业为基本单位从外存调入内存。

作业步是指每个作业运行期间都必须经过若干个相对独立相互关联的顺序加工的步骤。

作业流是指若干个作业进入系统后依次存放在外存上形成的输入作业流;在操作系统的控制下，逐个作业进程处理，于是形成了处理作业流。

3．在什么情况下需要使用作业控制块JCB？其中包含了哪些内容？

答：每当作业进入系统时，系统便为每个作业建立一个作业控制块JCB，根据作业类型将它插入到相应的后备队列中。

JCB 包含的内容通常有:1） 作业标识2)用户名称3)用户账户4)作业类型(CPU繁忙型、I/O 芳名型、批量型、终端型）5)作业状态6)调度信息（优先级、作业已运行）7)资源要求8)进入系统时间9） 开始处理时间10） 作业完成时间11） 作业退出时间12） 资源使用情况等

4．在作业调度中应如何确定接纳多少个作业和接纳哪些作业？

答:作业调度每次接纳进入内存的作业数，取决于多道程序度。应将哪些作业从外存调入内存，取决于采用的调度算法。最简单的是先来服务调度算法，较常用的是短作业优先调度算法和基于作业优先级的调度算法。

5．试说明低级调度的主要功能。

答：（1）保存处理机的现场信息（2）按某种算法选取进程（3）把处理机分配给进程。

6．在抢占调度方式中，抢占的原则是什么？

答：抢占的原则有：时间片原则、优先权原则、短作业优先权原则等。

7．在选择调度方式和调度算法时，应遵循的准则是什么？

答：

（1）面向用户的准则：周转时间短、响应时间快、截止时间的保证、优先权准则。

（2）面向系统的准则：系统吞吐量高、处理机利用率好、各类资源的平衡利用。

8．在批处理系统、分时系统和实时系统中，各采用哪几种进程（作业）调度算法？

答：批处理系统的调度算法：短作业优先、优先权、高响应比优先、多级反馈队列调度算法。

分时系统的调度算法：时间片轮转法。

实时系统的调度算法：最早截止时间优先即EDF、最低松弛度优先即LLF算法。

9．何谓静态和动态优先级？确定静态优先级的依据是什么？

答：静态优先级是指在创建进程时确定且在进程的整个运行期间保持不变的优先级。

动态优先级是指在创建进程时赋予的优先权，可以随进程推进或随其等待时间增加而改变的优先级，可以获得更好的调度性能。

确定进程优先级的依据：进程类型、进程对资源的需求和用户要求。

10．试比较FCFS和SPF两种进程调度算法。

答：相同点：两种调度算法都可以用于作业调度和进程调度。

不同点：FCFS调度算法每次都从后备队列中选择一个或多个最先进入该队列的作业，将它们调入内存、分配资源、创建进程、插入到就绪队列。该算法有利于长作业/进程，不利于短作业/进程。SPF算法每次调度都从后备队列中选择一个或若

干个估计运行时间最短的作业，调入内存中运行。该算法有利于短作业/进程，不利于长作业/进程。

11．在时间片轮转法中，应如何确定时间片的大小？

答：时间片应略大于一次典型的交互需要的时间。一般应考虑三个因素：系统对相应时间的要求、就绪队列中进程的数目和系统的处理能力。

12．通过一个例子来说明通常的优先级调度算法不能适用于实时系统？

答：实时系统的调度算法很多，主要是基于任务的开始截止时间和任务紧急/松弛程度的任务优先级调度算法，通常的优先级调度算法不能满足实时系统的调度实时性要求而不适用。

13．为什么说多级反馈队列调度算法能较好地满足各方面用户的需求？

答：（1）终端型作业用户提交的作业大多属于较小的交互型作业，系统只要使这些作业在第一队列规定的时间片内完成，终端作业用户就会感到满足。

（2）短批处理作业用户，开始时像终端型作业一样，如果在第一队列中执行一个时间片段即可完成，便可获得与终端作业一样的响应时间。对于稍长作业，通常只需在第二和第三队列各执行一时间片即可完成，其周转时间仍然较短。

（3）长批处理作业，它将依次在第1，2，…，n个队列中运行，然后再按轮转方式运行，用户不必担心其作业长期得不到处理。所以，多级反馈队列调度算法能满足多用户需求。

14．为什么在实时系统中，要求系统（尤其是CPU）具有较强的处理能力？

答：实时系统中通常有着多个实时任务。若处理机的处理能力不够强，有可能因为处理机忙不过来而使某些实时任务得不到及时处理，导致发生难以预料的后果。

15．按照调度方式可将实时调度算法分为哪几种？

答：可分为非抢占式和抢占式两种算法。而非抢占式算法又分为非抢占式轮转和优先调度算法；抢占式调度算法又分为基于时钟中断的抢占式优先权和立即抢占式优先权调度算法。

16．什么是最早截止时间优先调度算法？举例说明。

答：根据任务的开始截止时间确定的任务优先级调度算法。截止时间越早则优先级越高。该算法要求在系统中保持一个实时任务就绪队列，该队列按各任务截止时间的先后排序。

举例：非抢占式调度方式用于非周期实时任务。图3-9 是将该算法用于非抢占调度方式之例。该例中具有四个非周期任务，它们先后到达。系统首先调度任务1执行，在任务1执行期间，任务2、3又先后到达。由于任务3的开始截止时间早于任务2，故系统在任务1后将调度任务3执行。在此期间又到达作业4，其开始截止时间仍是早于任务2的，故在任务3执行完后，系统又调度任务4执行，最后才调度任务2执行。

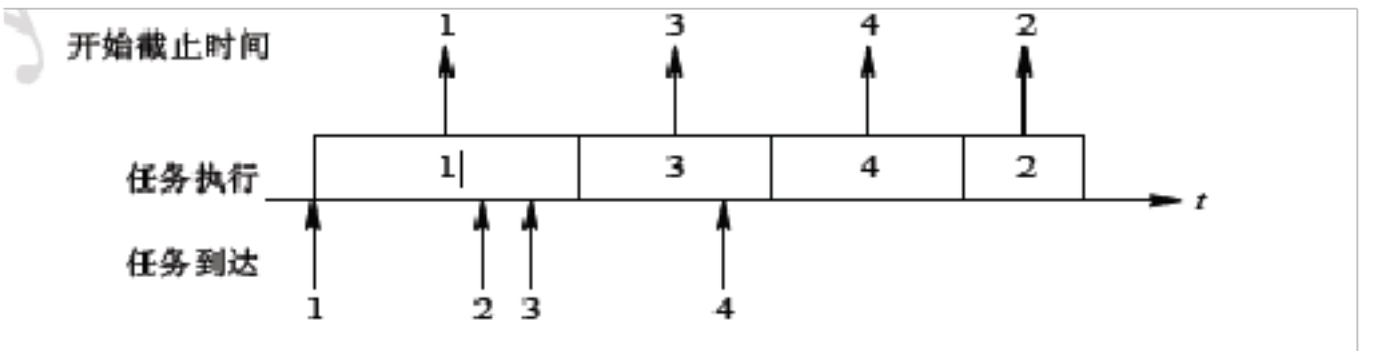


图3-9 EDF算法用于非抢占调度的调度方式

17．什么是最低松弛度优先调度算法？举例说明之。

答：该算法是根据任务紧急(或松弛)的程度，来确定任务的优先级。任务的紧急程度愈高，为该任务所赋予的优先级就愈高，以使之优先执行。例如，一个任务在200 ms 时必须完

成，而它本身所需的运行时间就有100 ms，因此，调度程序必须在100 ms 之前调度执行，该任务的紧急程度(松弛程度)为100 ms。又如，另一任务在400 ms 时必须完成，它本身需要运行 150 ms，则其松弛程度为 250 ms。

18．何谓死锁？产生死锁的原因和必要条件是什么？

答：死锁是指多个进程在运行过程中因争夺资源而造成的一种僵局，当进程处于这种僵持状态时，若无外力作用，它们都将无法再向前推进。

产生死锁的原因为竞争资源和进程间推进顺序非法。其必要条件是：互斥条件、请求和保持条件、不剥夺条件、环路等待条件。

19．在解决死锁问题的几个方法中，哪种方法最易于实现？哪种方法使资源利用率最高？

答：解决死锁的四种方法即预防、避免、检测和解除死锁中，预防死锁最容易实现；
避免死锁使资源的利用率最高。

20．请详细说明可通过哪些途径预防死锁。

答：（1）摈弃“请求和保持”条件，就是如果系统有足够资源，便一次性把进程需要的所有资源分配给它；

（2）摈弃“不剥夺”条件，就是已经拥有资源的进程，当它提出新资源请求而不能立即满足时，必须释放它已保持的所有资源，待以后需要时再重新申请；

（3）摈弃“环路等待”条件，就是将所有资源按类型排序标号，所有进程对资源的请求必须严格按序号递增的次序提出。

21．在银行家算法的例子中，如果P0发出请求向量由Request(0,2,0)改为Request(0,1,0)，问系统可否将资源分配给它？

答：（1）可以。银行家算法各种资源数量分别为10、5、7，在T_0时刻的资源分配如图所示：

资源情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	5	3	0	1	0	7	4	3	3	3	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

（2）具体分析如下：

①$Requst_0(0,1,0) <= Need_0(7,4,3)$；

② $Requst_0(0,1,0) <= Available(2,3,0)$；

系统先假定可为P_0分配资源，并修改$Available_0$，$Allocation_0$和$Need_0$向量，由此形成的资源变化情况如下图所示：

资源情况 / 进程	Max			Allocation			Need			Available		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	7	6	3	0	1	0	7	4	3	3	2	2
P_1	3	2	2	2	0	0	1	2	2			
P_2	9	0	2	3	0	2	6	0	0			
P_3	2	2	2	2	1	1	0	1	1			
P_4	4	3	3	0	0	2	4	3	1			

（3）P0请求资源：P_0发出请求向量$Requst_0(0,1,0)$，系统按银行家算法进行检查：

① Requst$_0$(0,1,0)<=Need$_0$(7,4,3);

② Requst$_0$(0,1,0)<=Available(2,3,0);

③ 系统暂时先假定可为P0分配资源，并修改_____有关数据，如下图所示

资源情况	Work			Need			Allocation			Work+ Allocation			
进程	A	B	C	A	B	C	A	B	C	A	B	C	Finish
P_0	3	2	2	7	3	3	0	2	0	3	4	2	Ture
P_1	3	4	2	1	2	2	2	0	0	5	4	2	Ture
P_2	5	4	2	6	0	0	3	0	2	8	4	4	Ture
P_3	8	4	4	0	1	1	2	1	1	10	5	5	Ture
P_4	10	5	5	4	3	1	0	0	2	10	5	7	Ture

综上所述系统可以将资源分配给它。

22. 银行家算法中出现以下资源分配，试问（1）该状态是否安全？（2）若进程P2 提出Request(1,2,2,2)后，系统能否将资源分配给它？

Process	Allocation	Need	Available
P_0	0 0 3 2	0 0 1 2	1 6 2 2
P_1	1 0 0 0	1 7 5 0	
P_2	1 3 5 4	2 3 5 6	
P_3	0 3 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

试问： （1）该状态是否安全?

（2）若进程P_2提出请求Request（1,2,2,2）后，系统能否将资源分配给它？

答： （1）安全，因为存在安全序列{P_0,P_3,P_4,P_1,P_2}

（2）系统能分配资源，分析如下。

① Request(1,2,2,2) <= Need$_2$(2,3,5,6);

② Request(1,2,2,2) <= Available$_2$(1,3,5,4);

③ 系统先假定可为P_2分配资源，并修改Available$_2$，Allocation$_2$和Need$_2$向量，

由此形成的资源变化情况如下图所示：

Process	Allocation	Need	Available
P_0	0 0 3 2	0 0 1 2	0 4 0 0
P_1	1 0 0 0	1 7 5 0	
P_2	2 5 7 6	2 3 5 6	
P_3	0 3 3 2	0 6 5 2	
P_4	0 0 1 4	0 6 5 6	

④ 再利用安全性算法检查此时系统是否安全。如下图

Process	Work	Allocation	Need	Work+Allocation	Finish
P_2	0 4 0 0	2 5 7 6	2 3 5 6	2 9 7 6	true
P_0	2 9 7 6	0 0 3 2	0 0 1 2	2 9 10 8	true
P_1	2 9 10 8	1 0 0 0	1 7 5 0	3 9 10 8	true
P_3	3 9 10 8	0 3 3 2	0 6 5 2	3 12 13 10	true
P_4	3 12 13 10	0 0 1 4	0 6 5 6	3 12 14 14	true

由此进行的安全性检查得知，可以找到一个安全序列{P_2, P_0, P_1, P_3, P_4}。

第四章

1．为什么要配置层次式存储器？

答：设置多个存储器可以使存储器两端的硬件能并行工作；采用多级存储系统，特别是Cache 技术，是减轻存储器带宽对系统性能影响的最佳结构方案；在微处理机内部设置各种缓冲存储器，减轻对存储器存取的压力。增加CPU中寄存器数量大大缓解对存储器压力。

2．可采用哪几种方式将程序装入内存？它们分别适用于何种场合？

答：　（1）绝对装入方式，只适用于单道程序环境。

（2）可重定位装入方式，适用于多道程序环境。

（3）动态运行时装入方式，用于多道程序环境；不允许程序运行时在内存中移位置。

3．何谓静态链接？何谓装入时动态链接和运行时的动态链接？P120

答：静态链接是指在程序运行前，先将各目标模块及它们所需的库函数，链接成一个完整的装配模块，以后不再拆开的链接方式。

装入时动态链接是指将用户源程序编译后得到的一组目标模块，在装入内存时采用边装入边链接的链接方式。

运行时动态链接是指对某些目标模块的链接，是在程序执行中需要该目标模块时，才对它进行的链接。

4．在进行程序链接时，应完成哪些工作？

答：由链接程序Linker将编译后形成的一组目标模块，以及它们需要的库函数链接在一起，形成一个完整的装入模块Load Module。主要工作是修改程序内的相对地址和修改目标程序中的外部调用标号。

5．在动态分区分配方式中，应如何将各空闲分区链接成空闲分区链？

答：在每个分区的起始部分，设置一些控制分区分配的信息，以及用于链接各分区所用的前向指针；在分区尾部设置一个后向指针，通过前后向链接指针，将所有空闲分区链成一个双向链。当分区分配出去后，把状态位由“0”改为“1”。

6．为什么要引入动态重定位？如何实现？

答：在程序执行过程中，每当访问指令或数据时，将要访问的程序或数据的逻辑地址转换成物理地址，引入了动态重定位；

具体实现方法是在系统中增加一个重定位寄存器，用来装入程序在内存中的起始地址，

程序执行时，真正访问的内存地址是相对地址与重定位寄存器中的地址相加之和，从而实现动态重定位。

7．在采用首次适应算法回收内存时，可能出现哪几种情况？应怎样处理这些情况？

答：在采用首次适应算法回收内存时可能出现4种情况：

（1）回收区前邻空闲区。将回收区与前邻空闲区合并，将前邻空闲区大小修改为两者之和。

（2）回收区后邻空闲区。将两区合并，改后邻空闲区始址为回收区始址，大小为两者之和。

（3）回收区前后均邻空闲区。将三个分区合并，修改前邻空闲区大小为三者之和。
（4）回收区前后均不邻空闲区。为回收区设置空闲区表项，填入回收区始址和大小并插入空闲区队列。

8．令$buddy_k(x)$表示大小为2^k、地址为x 的块的伙伴系统地址，试写出$buddy_k(x)$的通用表达式。

答:当 $x \ MOD \ 2^{k+1} =0$ 时，$buddy_k(x)=x+2^k$；当$x \ MOD \ 2^{k+1}=2^k$ 时，$buddy_k(x)=x-2^k$

9．分区存储管理中常用那些分配策略？比较它们的优缺点。
答：分区存储管理中的常用分配策略：首次适应算法、循环首次适应算法、最佳适应算法、最坏适应算法。

首次适应算法优缺点：保留了高址部分的大空闲区，有利于后来的大型作业分配；低址部分不断被划分，留下许多难以利用的小空闲区，每次查找都从低址开始增加了系统开销。

循环首次适应算法优缺点：内存空闲分区分布均匀，减少了查找系统开销；缺乏大空闲分区，导致不能装入大型作业。

最佳适应算法优缺点：每次分配给文件的都是最适合该文件大小的分区，内存中留下许多难以利用的小空闲区。

最坏适应算法优缺点：剩下空闲区不太小，产生碎片几率小，对中小型文件分配分区操作有利；存储器中缺乏大空闲区，对大型文件分区分配不利。

10．在系统中引入对换后可带来哪些好处？
答：交换技术将暂不需要的作业移到外存，让出内存空间以调入其它作业，交换到外存的作业也可以被再次调入。目的是解决内存紧张问题，带来的好处是进一步提高了内存利用率和系统吞吐量。
11．为实现对换，系统应具备哪几方面的功能？
答：系统应具备三方面功能：对换空间管理，进程换出，进程换入。
12．在以进程为单位进行对换时，每次是否都将整个进程换出？为什么？
答：不是。系统首先选择处于阻塞状态且优先级最低的进程作为换出进程，然后启动磁盘，将该进程的程序和数据传送到磁盘的兑换区。若传送过程未出错，便可回收该进程占用的内存空间，并对该进程的进程控制块做相应修改，所以并不需要将整个进程换出。
13．为实现分页存储管理，需要哪些硬件的支持？
答：动态重定位技术、虚拟存储技术、多道程序设计技术。
14．较详细的说明引入分段存储管理是为了满足用户哪几方面的需要。
答：
1）方便编程。用户通常把自己的作业按照逻辑关系划分为若干段，每段都从0 编址，并有自己名字和长度。因此，希望要访问的逻辑地址是由段名和段内偏移量决定。
2）信息共享。在实现对程序和数据的共享时，是以信息逻辑单位为基础。分页系统中的页是存放信息的物理单位，无完整意义，不便于共享；段是信息的逻辑单位。为了实现段的共享，希望存储管理能与用户程序分段的组织方式相适应。
3）信息保护。对信息的逻辑单位进行保护，分段能更有效方便地实现信息保护功能。
4）动态增长。在实际应用中，有些段特别是数据段，在使用过程中会不断增长，事先又无

法确切知道增长多少。分段存储管理方式能较好解决这个问题。
5）动态链接。运行时先将主程序对应的目标程序装入内存并启动运行，运行过程中又需要调用某段时，才将该段调入内存链接。所以动态链接也要求以段作为管理单位。
15．在具有快表的段页式存储管理方式中，如何实现地址变换？
答：在CPU给出有效地址后，由地址变换机构自动将页号P送入高速缓冲寄存器，并将此页号与高速缓存中的所有页号比较，若找到匹配页号，表示要访问的页表项在快表中。可直接从快表读出该页对应物理块号，送到物理地址寄存器中。如快表中没有对应页表项，则再访问内存页表，找到后，把从页表项中读出物理块号送地址寄存器；同时修改快表，将此页表项存入快表。但若寄存器已满，则OS必须找到合适的页表项换出。
16．为什么说为什么说分段系统比分页系统更易于实现信息的共享和保护？
答：分页系统的每个页面是分散存储的，为了实现信息共享和保护，页面之间需要一一对应，为此需要建立大量的页表项；而分段系统的每个段都从0 编址，并采用一段连续的地址空间，在实现共享和保护时，只需为要共享和保护的程序设置一个段表项，将其中的基址与内存地址一一对应就能够实现。

17．分段和分页存储管理有何区别？
答：
（1）是信息的物理单位，分页是为了实现离散分配方式，以消减内存的外部零头，提高内存利用率。段则是信息的逻辑单位，它含有一组相对完整的信息。
（2）页的大小固定且由系统决定，由系统把逻辑地址划分为页号和页内地址两部分，是由机械硬件实现的,因而在系统中只能有一种大小的的页面;而段的长度却不固定,决定于用户所编写的程序,通常由编译程序在对原程序进行编译时,根据信息的性质来划分。
（3）分页的作业地址空间是一维的,而分段作业地址空间则是二维的。
18．试全面比较连续分配和离散分配方式.
答：
（1）连续分配是指为一个用户程序分配一个连续的地址空间，包括单一和分区两种分配方式。单一方式将内存分为系统区和用户区，最简单，只用于单用户单任务操作系统；分区方式分固定和动态分区。
（2）离散分配方式分为分页、分段和段页式存储管理。分页式存储管理旨在提高内存利用率，分段式存储管理旨在满足用户(程序员)的需要，段页式存储管理则将两者结合起来，具有分段系统便于实现、可共享、易于保护和动态链接等优点，又能像分页系统很好解决外部碎片及为各段可离散分配内存等问题，是比较有效的存储管理方式;
19．虚拟存储器有哪些特征？其中最本质的特征是什么？
答：虚拟存储器有多次性、对换性、虚拟性三大特征。最本质的特征是虚拟性。
20．实现虚拟存储器需要哪些硬件支持？
答：（1）请求分页（段）的页（段）表机制（2）缺页（段）中断机构（3）地址变换机构
21．实现虚拟存储器需要哪几个关键技术？
答：
（1）在分页请求系统中是在分页的基础上，增加了请求调页功能和页面置换功能所形成的页式虚拟存储系统。允许只装入少数页面的程序（及数据），便启动运行。
（2）在请求分段系统中是在分段系统的基础上，增加了请求调段及分段置换功能后形成的段式虚拟存储系统。允许只装入少数段（而非所有段）的用户程序和数据，即可启动运行。
22．在请求分页系统中，页表应包括哪些数据项？每项的作用是什么？
答：页表应包括：页号、物理块号、状态位P、访问字段A、修改位M和外存地址。

其中状态位P 指示该页是否调入内存，供程序访问时参考；访问字段A 用于记录本页在一段时间内被访问的次数，或最近已有多长时间未被访问，提供给置换算法选择换出页面时参考；修改位M 表示该页在调入内存后是否被修改过；外存地址用于指出该页在外存上的地址，通常是物理块号，供调入该页时使用。

23．在请求分页系统中，应从何处将所需页面调入内存？

答：请求分页系统中的缺页从何处调入内存分三种情况：

（1）系统拥有足够对换区空间时，可以全部从对换区调入所需页面，提高调页速度。在进程运行前将与该进程有关的文件从文件区拷贝到对换区。

（2）系统缺少足够对换区空间时，不被修改的文件直接从文件区调入；当换出这些页面时，未被修改的不必换出，再调入时，仍从文件区直接调入。对于可能修改的，在换出时便调到对换区，以后需要时再从对换区调入。

（3）UNIX 方式。未运行页面从文件区调入。曾经运行过但被换出页面，下次从对换区调入。UNIX 系统允许页面共享，某进程请求的页面有可能已调入内存，直接使用不再调入。24．在请求分页系统中，常采用哪几种页面置换算法？

答：采用的页面置换算法有：最佳置换算法和先进先出置换算法，最近最久未使用（LRU）置换算法，Clock置换算法，最少使用置换算法，页面缓冲算法等。

25．在请求分页系统中，通常采用哪种页面分配方式？为什么？

答：固定分配方式是基于进程的类型(交互型)或根据程序员、系统管理员的建议，为每个进程分配固定页数的内存空间，整个运行期间不再改变；采用可变分配方式有全局置换和局部置换两种，前者易于实现，后者效率高。

26．在一个请求分页系统中，采用LRU 页面置换算法时，假如一个作业的页面走向为 4，3，2，1，4，3，5，4，3，2，1，5，当分配给该作业的物理块数M分别为3和4时，试计算访问过程中所发生的缺页次数和缺页率 ？比较所得结果 ？

答：当分配给该作业的物理块数M为3时，缺页7次，缺页率：7/12=0.583；

当分配给该作业的物理块数M为4时，缺页4次，缺页率：4/12=0.333.

27．实现LRU算法所需的硬件支持是什么?

答：需要寄存器和栈等硬件支持。寄存器用于记录某进程在内存中各页的使用情况，栈用于保存当前使用的各个页面的页面号。

28．试说明改进型 Clock 置换算法的基本原理 .

答：因为修改过的页面在换出时付出的开销比未被修改过的页面大，在改进型Clock 算法中，既考虑页面的使用情况，还要增加置换代价的因素；在选择页面作为淘汰页面时，把同时满足未使用过和未被修改作为首选淘汰页面。

29．说明请求分段系统中的缺页中断处理过程。

答：请求分段系统中的缺页中断处理过程描述如下：

（1）根据当前执行指令中的逻辑地址查页表，判断该页是否在主存储器中

（2）该页标志为“0”形成缺页中断，中断装置通过交换PSW让操作系统的中断处理程序占用处理器。

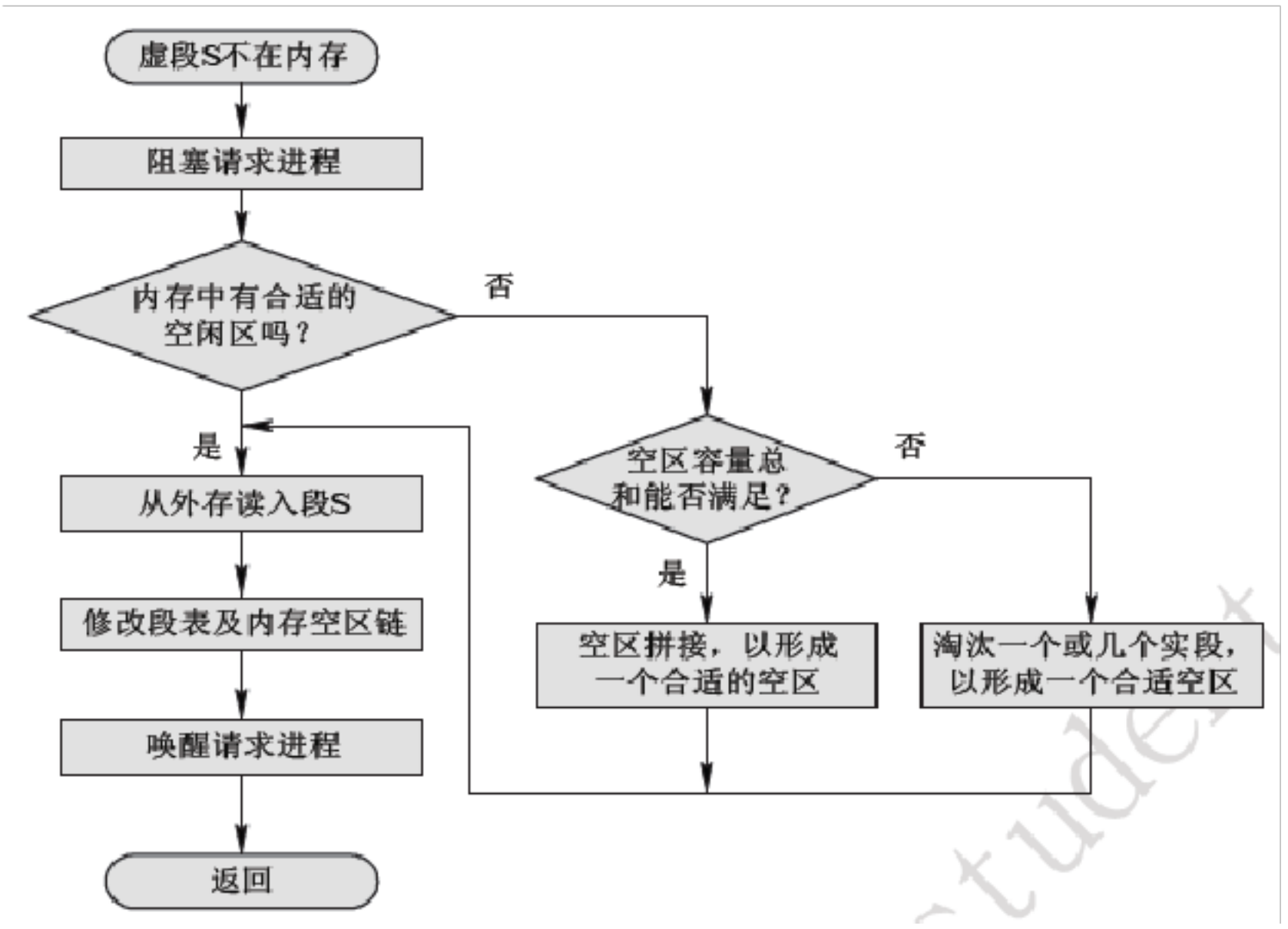
（3）操作系统处理缺页中断处理的办法是查主存分配表找一个空闲的主存块，查页表找出该页在磁盘上位置，启动磁盘读出该页信息。

（4）把从磁盘上读出的信息装入找到的主存块中。

（5）当页面住处被装入主存后，应修改页表中对应的表目，填上该页所占用的主存块把标志置为“1”，表示该页已在主存储器中

（6）由于产生缺页中断时的那条指令并没执行完，所以在把页面装入之后应重新执行被中

断指令。
请求分段系统中的缺页中断处理过程如下图所示：



30．如何实现分段共享 ？
答：在每个进程的段表中，用相应的表项指向共享段在内存中起始地址；配置相应的数据结构作为共享段表，在段表项中设置共享进程计数Count ，每调用一次该共享段，Count值增 1，每当进程释放一个共享段时，Count 减1，若减为0，则系统回收该共享段的物理内存，取消在共享段表中该段对应的表项；共享段应给不同的进程以不同的存取权限；不同的进程可以使用不同的段号去共享该段。

第五章

1．试说明设备控制器的组成。
答：由设备控制器与处理机的接口，设备控制器与设备的接口与I/O逻辑组成。
2．为了实现CPU与设备控制器间的通信，设备控制器应具备哪些功能？
答：接收和识别命令；数据交换；标识和报告设备状态；地址识别；数据缓冲；差错控制。
3．什么是字节多路通道？什么是数组选择通道和数组多路通道？
答：（1）字节多路通道。按字节交叉方式工作的通道。通常含有许多非分配型子通道，数量从几十到数百个，每个子通道连接一台I/O 设备，控制其I/O 操作。子通道按时间片轮转方式共享主通道。
（2）数组选择通道。按数组方式传送数据，传输速率很高，每次只允许一个设备数据。

（3）数组多路通道。将数组选择通道传输速率高和字节多路通道的各子通道分时并行操作的优点结合而成。含有多个非分配型子通道，具有很高的数据传输率和通道利用率。
4．如何解决因通道不足而产生的瓶颈问题？
答：解决问题的有效方法是增加设备到主机间的通路而不增加通道，把一个设备连到多个控

制器上，控制器又连到多个通道上，这种多通路方式解决了“瓶颈”问题，提高了系统可靠性，个别通道或控制器的故障不会使设备和存储器之间没有通路。

5．试对VESA 及PCI两种总线进行比较。

答：VESA总线的设计思想是以低价占领市场。总线带宽32位，最高传输速率132Mb/s。广泛用于486微机。缺点是能连接的设备数仅为2~4 台，控制器中无缓冲，难于适应处理器速度的提高，不支持Pentium机。

PCI总线在CPU和外设间插入了复杂的管理层，协调数据传输和提供一致接口。管理层中配有数据缓冲，放大了线路的驱动能力，最多支持10种外设，支持高时钟频率的CPU运行，最大传输速率132Mb/s。可连接ISA、EISA 等传统总线，又支持Pentium的64位系统，是基于奔腾等新一代微处理器而发展的总线。

6．试说明推动I/O控制发展的主要因素是什么？

答:推动I/O 控制发展的主要动力在于尽量减少主机对I/O 控制的干预,把主机从繁杂的I/O控制事务中解脱出来，用更多的时间和精力去完成其数据处理任务。同时，中断机制在计算机系统中的引入、DMA 控制器的出现和通道研制的成功使I/O 控制的发展具备了技术支持和成为可能。

7．有哪几种I/O控制方式？各适用于何种场合？

答：共有四种I/O 控制方式。

(1)程序I/O 方式：早期计算机无中断机构，处理机对I/O设备的控制采用程序I/O方式或称忙等的方式。

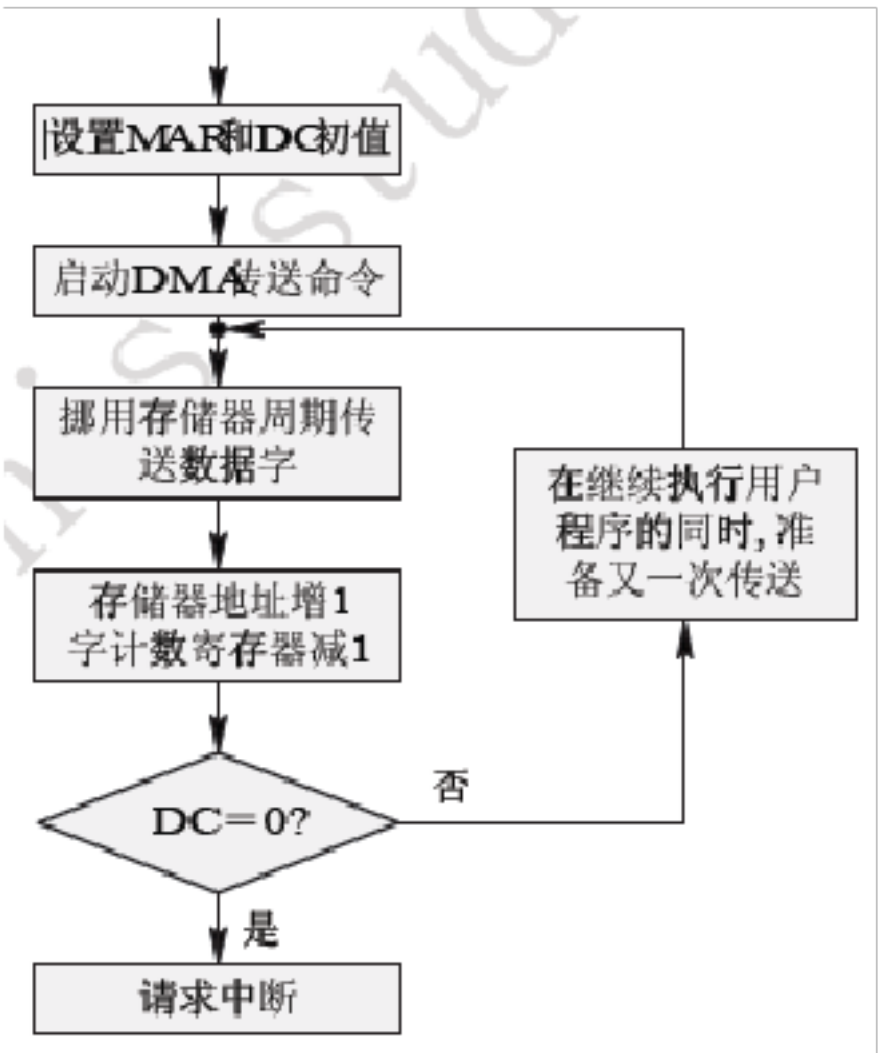
(2)中断驱动I/O 控制方式：适用于有中断机构的计算机系统中。

(3)直接存储器访问（DMA）I/O 控制方式：适用于具有DMA控制器的计算机系统中。

(4)I/O 通道控制方式：具有通道程序的计算机系统中。

8．试说明DMA 的工作流程。

答：以从磁盘读入数据为例，说明DMA的工作流程。当CPU要从磁盘读入数据块时，先向磁盘控制器发送一条读命令。该命令被送到命令寄存器CR中。同时还发送本次要读入数据的内存起始目标地址，送入内存地址寄存器MAR；本次要读数据的字节数送入数据计数器DC，将磁盘中的源地址直接送DMA控制器的I/O 控制逻辑上。然后启动DMA 控制器传送数据，以后CPU便处理其它任务。整个数据传送过程由DMA控制器控制。下图为DMA方式的工作流程图。



9．引入缓冲的主要原因是什么？

答：引入缓冲的主要原因是：

（1）缓和CPU与I/O 设备间速度不匹配的矛盾

（2）减少对CPU的中断频率，放宽对中断响应时间的限制

（3）提高CPU与I/O 设备之间的并行性

10．在单缓冲情况下，为什么系统对一块数据的处理时间为max（C,T）+M ？

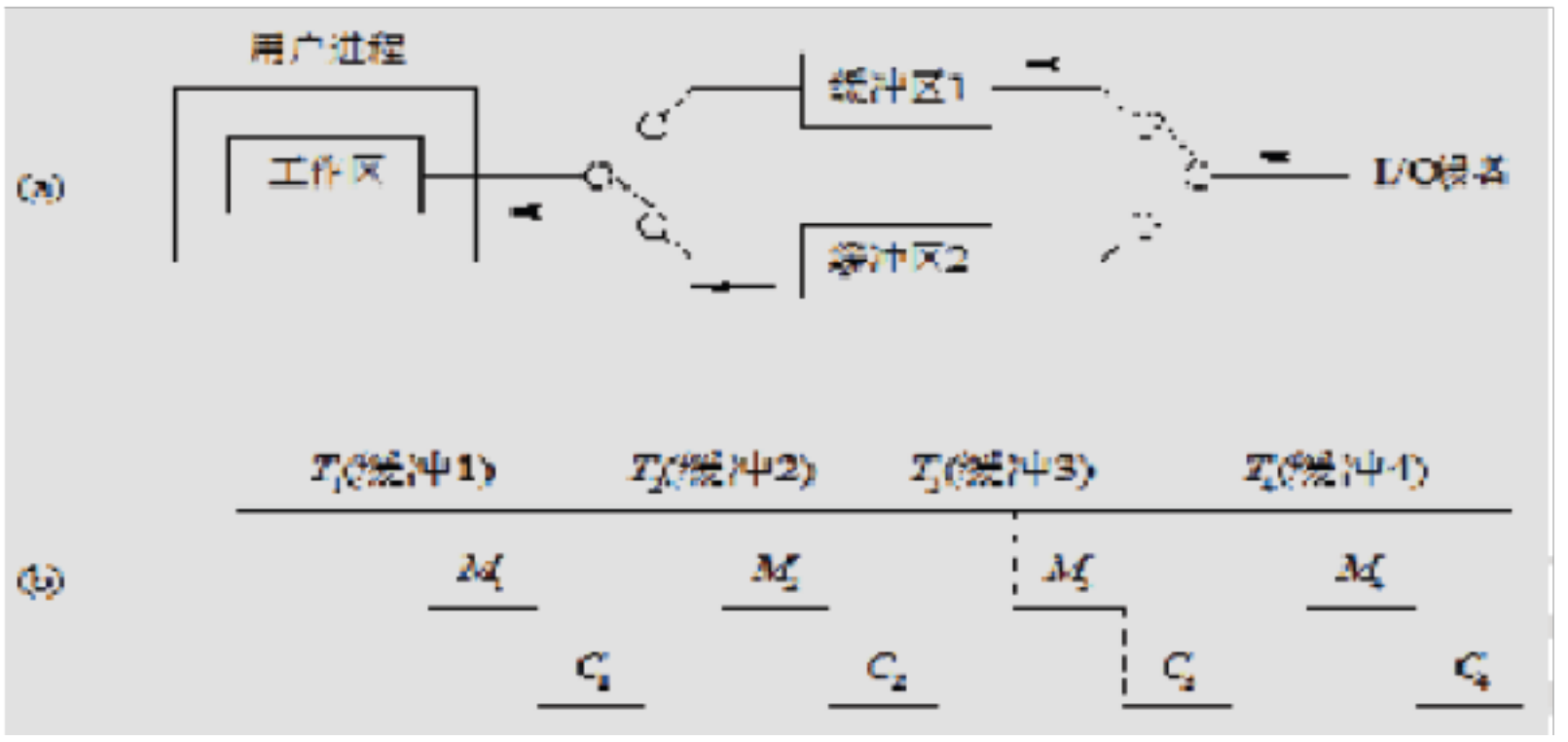
答：在块设备输入时，先从磁盘把一块数据输入到缓冲区，耗时为T；然后由操作系统将缓冲区数据送给用户区，耗时M；接下来由CPU 对块数据进行计算，耗时C。在单缓冲情况下，磁盘把数据输入到缓冲区的操作和CPU 对数据的计算过程可以并行展开，所以系统对每一整块数据的处理时间为max(C, T) + M。

11．为什么在双缓冲情况下，系统对一块数据的处理时间为max（T,C）？

答：写入者花费时间T 将数据写满一个缓冲区后再写另一个缓冲区；读出者花费时间M 将一个缓冲区数据送到用户区后再传送另一个缓冲区数据，计算者读出用户区数据进行处理。由于将数据从缓冲区传到用户区操作必须与读用户区数据处理串行进行，而且可以与从外存传送数据填满缓冲区的操作并行。因此耗时大约为max(C+M,T)。考虑M 是内存数据块的移动耗时非常短暂可以省略，因此近似地认为系统对一块数据处理时间为max(C,T)。

12．试绘图说明把多缓冲用于输出时的情况。

答：多缓冲用于输出的示意图如下：



13．试说明收容输入工作缓冲区和提取输出工作缓冲区的工作情况。

答：

① 收容输入工作缓冲区的工作情况为：在输入进程需要输入数据时，调用GetBuf(EmptyQueue)过程，从EmptyQueue队列的队首摘下一个空缓冲区，作为收容输入工作缓冲区Hin。然后把数据输入其中，装满后再调用PutBuf(InputQueue, Hin)过程，将该缓冲区挂在输入队列InputQueue的队尾。

② 提取输出工作缓冲区的工作情况为：当要输出数据时，调用GetBuf(OutputQueue)过程，从输出队列的队首取得一装满输出数据的缓冲区作为提取输出工作缓冲区Sout。在数据提取完后，再调用PutBuf(EmptyQueue, Sout)过程，将该缓冲区挂到空缓冲队列EmptyQueue的队尾。

14．何谓安全分配方式和不安全分配方式？

答：

① 安全分配方式是指每当进程发出I/O 请求后，便进入阻塞状态，直到其I/O 操作完成时才被唤醒。在采用这种分配策略时，一旦进程已获得某种设备资源后便阻塞，使它不可能再请求任何资源，而在它运行时又不保持任何资源。这种分配方式已经摒弃了造成死锁的“请求和保持”条件，分配是安全的。缺点是进程进展缓慢，CPU与I/O 设备串行工作。

②不安全分配方式是指进程发出I/O 请求后仍继续执行，需要时又可发出第二个I/O 请求、第三个I/O 请求。仅当进程请求的设备已被另一个进程占有时，进程才进入阻塞状态。优点是一个进程可同时操作多个设备，进程推进迅速。缺点是分配不安全，可能具有“请求和保持”条件，可能造成死锁。因此，在设备分配程序中需增加一个功能，用于对本次的设备分配是否会发生死锁进行安全性计算，仅当计算结果表明分配安全的情况下才进行分配。

15．为何要引入设备独立性？如何实现设备独立性？

答:现代操作系统为了提高系统的可适应性和可扩展性,都实现了设备独立性或设备无关性。基本含义是应用程序独立于具体使用的物理设备，应用程序以逻辑设备名请求使用某类设备。实现了设备独立性功能可带来两方面的好处：（1）设备分配时的灵活性；（2）易于实现I/O 重定向。

为了实现设备的独立性，应引入逻辑设备和物理设备概念。在应用程序中，使用逻辑设备名请求使用某类设备；系统执行时是使用物理设备名。鉴于驱动程序是与硬件或设备紧密相关的软件，必须在驱动程序之上设置一层设备独立性软件，执行所有设备的公有操作、完成逻辑设备名到物理设备名的转换（为此应设置一张逻辑设备表）并向用户层（或文件层）软件提供统一接口，从而实现设备的独立性。

16．在考虑到设备的独立性时，应如何分配独占设备？

答：在考虑到设备的独立性时，应按如下步骤来分配独占设备：
(1) 进程以逻辑设备名提出I/O请求。
(2) 根据逻辑设备表获得I/O请求的逻辑设备对应物理设备在系统设备表中的指针。
(3) 检索系统设备表，找到属于请求类型、空闲可用且分配安全设备的设备控制表，将对应设备分配给请求进程；未找到则等待等待唤醒和分配。
(4) 到设备控制表中找出与其相连接的控制器的控制器控制表，根据状态字段判断是否忙碌，忙则等待；否则将该控制器分配给进程。
(5) 到该控制器的控制器控制表中找出与其相连接的通道的通道控制表，判断通道是否忙碌，忙则等待；否则将该通道分配给进程。
(6) 只有在设备、控制器和通道三者都分配成功时，这次的设备分配才算成功，然后便可启动设备进行数据传送。
17．何谓设备虚拟？实现设备虚拟时所依赖的关键技术是什么？
答：设备虚拟是指把独占设备经过某种技术处理改造成虚拟设备。
可虚拟设备是指一台物理设备在采用虚拟技术后，可变成多台逻辑上的虚拟设备，则可虚拟设备是可共享的设备，将它同时分配给多个进程使用，并对这些访问该物理设备的先后次序进行控制。
18．试说明SPOOLing 系统的组成。
答：SPOOLing 系统由输入井和输出井、输入缓冲区和输出缓冲区、输入进程 SPi 和输出进程 SPo 三部分组成。
19．在实现后台打印时，SPOOLing 系统应为请求I/O 的进程提供哪些服务？
答：在实现后台打印时，SPOOLing 系统应为请求 I/O的进程提供以下服务：
（1）由输出进程在输出井中申请一空闲盘块区，并将要打印的数据送入其中；
（2）输出进程为用户进程申请空白用户打印表，填入打印要求，将该表挂到请求打印队列。
（3）一旦打印机空闲，输出进程便从请求打印队列的队首取出一张请求打印表，根据表中要求将要打印的数据从输出井传送到内存缓冲区，再由打印机进行打印。
20．试说明设备驱动程序具有哪些特点。
答：设备驱动程序具有如下特点：
（1）是请求 I/O 进程与设备控制器间的一个通信程序；
（2）驱动程序与 I/O 设备的特性紧密相关；
（3）驱动程序与 I/O 控制方式紧密相关；
（4）驱动程序与硬件紧密相关，部分程序用汇编语言书写，基本部分往往固化在ROM中。
21．试说明设备驱动程序应具有哪些功能？
答：设备驱动程序的主要功能包括：
（1）将接收到的抽象要求转为具体要求；
（2）检查用户I/O请求合法性，了解I/O 设备状态，传递有关参数，设置设备工作方式；
（3）发出I/O 命令，启动分配到的I/O设备，完成指定I/O 操作；
（4）及时响应由控制器或通道发来的中断请求，根据中断类型调用相应中断处理程序处理；
（5）对于有通道的计算机，驱动程序还应该根据用户 I/O 请求自动构成通道程序。
22.设备中断处理程序通常需完成哪些工作？
答：设备中断处理程序通常需完成如下工作：
(1) 唤醒被阻塞的驱动程序进程；
(2) 保护被中断进程的CPU环境；

(3) 分析中断原因、转入相应的设备中断处理程序；
(4) 进行中断处理；
(5) 恢复被中断进程。
23.磁盘访问时间由哪几部分组成？每部分时间应如何计算？
答：磁盘访问时间由寻道时间Ts、旋转延迟时间Tr、传输时间Tt 三部分组成。
（1）Ts 是启动磁臂时间s 与磁头移动n条磁道的时间和，即Ts = m × n + s。
（2）Tr是指定扇区移动到磁头下面所经历的时间。硬盘15000r/min时Tr为2ms;软盘300或600r/min时Tr为50~100ms。
（3）Tt 是指数据从磁盘读出或向磁盘写入经历的时间。Tt 的大小与每次读/写的字节数b和旋转速度有关：Tt = b/rN。
24.目前常用的磁盘调度算法有哪几种？每种算法优先考虑的问题是什么？
答：目前常用的磁盘调度算法有先来先服务、最短寻道时间优先及扫描等算法。
(1) 先来先服务算法优先考虑进程请求访问磁盘的先后次序；
(2) 最短寻道时间优先算法优先考虑要求访问的磁道与当前磁头所在磁道距离是否最近；
(3) 扫描算法考虑欲访问的磁道与当前磁道间的距离，更优先考虑磁头当前的移动方向。
25.为什么要引入磁盘高速缓冲？何谓磁盘高速缓冲？
答：目前磁盘的I/O速度远低于内存的访问速度，通常低上4-6个数量级。因此，磁盘I/O已成为计算机系统的瓶颈。为提高磁盘I/O的速度，便引入了磁盘高速缓冲。
磁盘高速缓冲是指利用内存中的存储空间，暂存从磁盘中读出的一系列盘块中的信息。
26.在设计磁盘高速缓冲时，如何实现数据交付？
答：数据交付是指将磁盘高速缓存中的数据传给请求进程。当进程请求访问某个盘块中的数据时，由核心先查看磁盘高速缓冲，看其中是否存在所需盘块数据的拷贝。若有便直接从中提取数据交付给请求进程，避免了访盘操作，本次访问速度提高4-6 个数量级；否则先从磁盘中将要访问的数据读入并交付给请求者进程，同时送高速缓存以便下次直接读取。

27. 何谓提前读、延迟写和虚拟盘？
答：提前读是指在读当前盘块的同时，将下一个可能要访问的盘块数据读入缓冲区，以便需要时直接从缓冲区中读取，无需启动磁盘。
延迟写是指在写盘块时，将对应缓冲区中的立即写数据暂时不立即写以备不久之后再被访问，只将它置上“延迟写”标志并挂到空闲缓冲队列的末尾。当移到空闲缓冲队首并被分配出去时，才写缓冲区中的数据。只要延迟写块仍在空闲缓冲队列中，任何要求访问都可直接从其中读出数据或将数据写入其中，而不必去访问磁盘。
虚拟盘又称RAM盘，是利用内存空间仿真磁盘。其设备驱动程序可以接受所有标准的磁盘操作，但这些操作不是在磁盘上而是在内存中，因此速度更快。
28.廉价磁盘冗余阵列是如何提高对磁盘的访问速度和可靠性的？
答：廉价磁盘冗余阵列RAID是利用一台磁盘阵列控制器，统一管理和控制一组（几台到几十台）磁盘驱动器，组成高度可靠快速大容量的磁盘系统。
操作系统将RAID中的一组物理磁盘驱动器看作一个单个的逻辑磁盘驱动器。用户数据和系统数据可分布在阵列的所有磁盘中，并采取并行传输方式，大大减少数据传输时间和提高了可靠性。

第六章

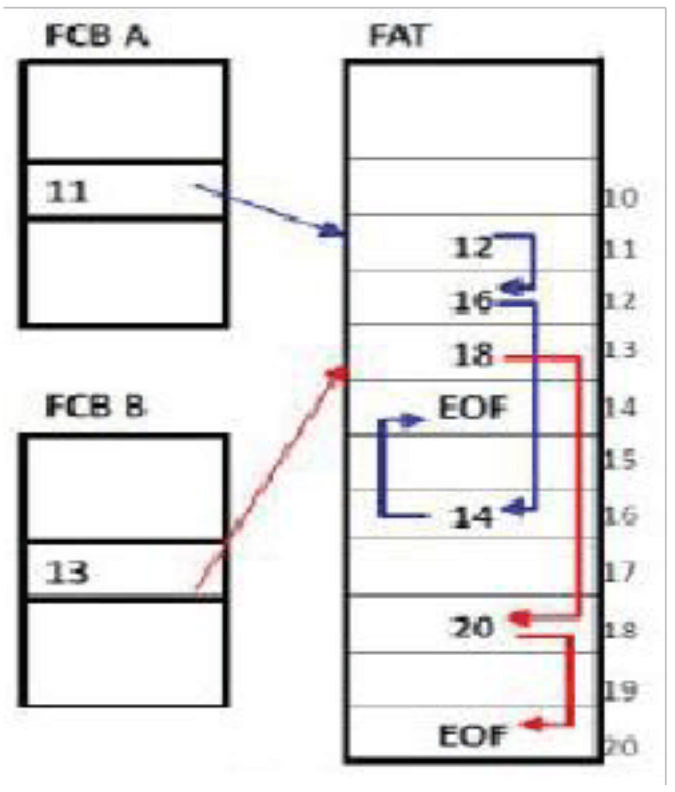
1. 何谓数据项、记录和文件？
答：①数据项分为基本数据项和组合数据项。基本数据项描述一个对象某种属性的字符集，

具有数据名、数据类型及数据值三个特性。组合数据项由若干数据项构成。
②记录是一组相关数据项的集合，用于描述一个对象某方面的属性。
③文件是具有文件名的一组相关信息的集合。
2．文件系统的模型可分为三层，试说明其每一层所包含的基本内容。
答：第一层：对象及其属性说明（文件、目录、硬盘或磁带存储空间）；
第二层：对对象操纵和管理的软件集合（I/O控制层即设备驱动程序、基本文件系统即物理I/O层、基本I/O管理程序或文件组织模块层、逻辑文件系统层）
第三层：文件系统接口（命令接口/图形化用户接口与程序接口）。
3．试说明用户可以对文件施加的主要操作有哪些？
答：用户通过文件系统提供的系统调用对文件实施操作。
（1）基本文件操作：创建、删除、读、写、截断、设置读/写位置等；
（2）文件打开和关闭操作：第一步通过检索文件目录找到指定文件属性及其在外存上位置；第二步对文件实施读写等相应操作。
（3）其他文件操作：一是文件属性操作；二是目录操作；三是文件共享与文件系统操作的系统调用实现等。

4．何谓逻辑文件？何谓物理文件？
答：逻辑文件是物理文件中存储的数据的一种视图方式，不包含具体数据，仅包含物理文件中数据的索引。物理文件又称文件存储结构，是指文件在外存上的存储组织形式。
5．如何提高对变长记录顺序文件的检索速度？
答：基本方法是为变长记录顺序文件建立一张索引表，以主文件中每条记录的长度及指向对应记录的指针（即该记录在逻辑地址空间的首址）作为相应表项的内容。由于索引表本身是一个定长记录的顺序文件，若将其按记录键排序，则实现了对主文件方便快捷的直接存取。如果文件较大，应通过建立分组多级索引以进一步提高检索效率。
6．试说明对索引文件和索引顺序文件的检索方法。
答： ① 索引文件的检索，首先根据用户（程序）提供的关键字，利用折半查找法检索索引表，找到相应表项；再利用给出的指向记录指针值，访问对应记录。
② 索引顺序文件的检索，首先利用用户（程序）提供的关键字及查找方法，检索索引表，找到该记录在记录组中的第一条记录表项，得到第一个记录在主文件中的位置；再利用顺序查找法查找主文件，找到所要求的记录。
7．试从检索速度和存储费用两方面对索引文件和索引顺序文件进行比较。
答：索引文件的主文件每条记录配置一个索引项，存储开销N，检索到具有指定关键字的记录，平均查找N/2 条记录。对于索引顺序文件，每个记录分组配置一个索引项，存储开销为N ，检索到具有指定关键字的记录，平均需要查找N /2次。
8．试说明顺序文件的结构及其优点。
答：第一种是串结构：各记录之间的顺序与关键字无关。第二种是顺序结构：指文件中的所有记录按关键字（词）排列。可以按关键词长短排序或英文字母顺序排序。
顺序文件的最佳应用场合是对诸记录进行批量存取时，存取效率最高；只有顺序文件才能存储在磁带上并有效工作。
9．在链接式文件中常用哪种链接方式？为什么？
答：链接方式分为隐式链接和显式链接两种形式。隐式链接是在文件目录的每个目录项中，都含有指向链接文件第一个盘块和最后一个盘块的指针。显式链接则把用于链接文件各物理块的指针，显式地存放在内存的一张链接表中。

10．在MS-DOS中有两个文件A 和B，A占用11、12、16和14 四个盘块；B 占用13、18和20三个盘块。试画出在文件A和B中各盘块间的链接情况及FAT的情况。

答：如下图所示。



11．NTFS文件系统对文件采用什么样的物理结构？

答：在NTFS 文件系统中，以簇作为磁盘空间分配和回收的基本单位。一个文件占若干个簇，一个簇只属于一个文件。

12．假定一个文件系统的组织方式与MS-DOS相似，在FAT中可有64K个指针，磁盘的盘块大小为512B，试问该文件系统能否指引一个512MB 的磁盘？

解：512MB/512B=1M 个盘块，而每个盘块都应有一个指针来指示，所以应该有1M 个指针，因此若有64K 指针不能指引一个512MB的磁盘。

13．为了快速访问，又易于更新，当数据为以下形式时，应选用何种文件组织方式。

⑴ 不经常更新，经常随机访问；⑵经常更新，经常按一定顺序访问；⑶经常更新，经常随机访问；

答：以上三种宜分别采用(1)顺序结构(2)索引顺序结构(3)索引结构的组织方式。

14．在UNIX 中，如果一个盘块的大小为1KB，每个盘块号占4个字节，即每块可放256个地址。请转换下列文件的字节偏移量为物理地址。

⑴9999； ⑵18000； ⑶420000

答：首先将逻辑文件的字节偏移量转换为逻辑块号和块内偏移量,就是将[字节偏移量]/[盘块大小]，商为逻辑块号，余数是块内偏移量。在FCB中，第0-9个地址为直接地址，第10个为一次间接地址，第11个地址为二次间接地址，第12个地址为三次间接地址。

再将文件的逻辑块号转换为物理块号。使用多重索引结构，在索引节点中根据逻辑块号通过直接索引或间接索引找到对应的物理块号。

（1）9999/1024=9 余783，则逻辑块号为9，直接索引第9个地址得到物理块号，块内偏移地址为783。

（2）18000/1024=17余592，则逻辑块号为10<17<10+256，通过一次间接索引在第10个地址可得到物理块号，块内偏移地址为592。

（3）420000/1024=410 余160，则逻辑块号为10+256<410，通过二次间接索引在第11个地址可得到一次间址，再由此得到二次间址，再找到物理块号，其块内偏移地址160。

15．什么是索引文件？为什么要引入多级索引？

答：索引文件是指当记录为可变长度时，通常为之建立一张索引表，并为每个记录设置一个表项构成的文件。通常将索引非顺序文件简称为索引文件。索引是为了是用户的访问速度更

快，多级索引结构可以有效的管理索引文件，可根据用户的访问情况多级处理。

16．试说明UNIX 系统中所采用的混合索引分配方式。

答：混合索引分配方式是指将多种索引分配方式结合而成的分配方式。常见的是采用直接地址和一级索引联合的分配方式,或两级索引分配方式,甚至三级索引分配方式。在UNIXSystem Ⅴ和BSD UNIX 的索引结点中，都设置了13 个地址项，即iaddr(0)～iaddr(12)，把所有地址项分成直接地址和间接地址。

17．对目录管理的主要要求是什么？

答：实现按名存取、提高检索目录的速度、文件共享、允许文件重名。

18．采用单级目录能否满足对目录管理的主要要求？为什么？

答：不能。单级目录在整个文件系统中只建立一张目录表，每个文件占一个目录项，其中含文件名、文件扩展名、文件长度、文件类型、文件物理地址、状态位等其它文件属性。

单级只能实现目录管理的基本功能，不能满足查找速度、允许重名和文件共享的要求。

19．目前广泛应用的目录结构有哪些？它有什么优点？

答：现代操作系统都采用多级目录结构。基本特点是查询速度快、层次结构清晰、文件管理和保护易于实现。

20．Hash 检索法有何优点？又有何局限性？

答:Hash检索法是系统把用户提供的文件名变换为文件目录的索引值,再利用该值查找目录，有效提高目录的检索速度，但Hash 检索法局限于非通配符文件名。

21．在Hash检索法中，如何解决“冲突”问题？

答：在Hash法查找目录时，如果目录表中相应目录项为空，表示系统中无指定文件。如果文件名与指定文件名匹配，表示找到了目标文件，也就找到了文件的物理地址。如果目录表中找到的相应文件名不匹配，则发生了冲突，需要Hash转换形成新的索引值，返回第一步重新查找。

22．试说明在树型目录结构中线性检索法的检索过程，并给出相应的流程图。

答：在树型结构的目录中，当有两个或多个用户共享一个子目录或文件时，将共享文件或字母路连接到两个或多个用户目录中，方便找到该文件。此时目录结构不再是树形结构，而是个有向非循环图DGA。

23．有一计算机系统利用图6-33 所示的位示图来管理空闲盘块。盘块的大小为1KB，现要为某文件分配量个盘块，试说明盘块的具体分配过程。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	0	1	1	1	1	0	1	1	1	1
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

答：分配量个盘块的过程如下：

⑴ 顺序扫描位示图，从中找到第一个值为0的二进制位，得到行号i=3，列号j=3。

⑵ 将找到的二进制位转换成对应盘块号。盘块号为：b=（3-1）*16+3=35；

⑶ 修改位示图，令map[3，3]=1，并将该盘块分配出去。

类似地，可使用相同的方法找到第二个值为0的二进制位，得到行号i=4，列号j=7，其对应的盘块号为55，令map[i，j]=1，并将该盘块分配出去。

24．某操作系统磁盘文件空间共500块，若用字长为32位的位示图管理磁盘空间，试问：（1）位示图需要多少字？

（2）第i字第j位对应的块号是多少？

（3）给出申请/归还一块的工作流程。

答：（1）位示图需要的字数计算：INT（500/32）=16 个字。

（2）块号b=(i-1)*32+j

（3）申请的过程：顺序扫描位示图、找到空闲块并分配、修改位示图map[i,j]=1。

归还的过程：找到回收盘块在位示图中的行和列，修改位示图map[i,j]=0。

25．对空闲磁盘空间的管理常采用哪几种分配方式？在UNIX系统中采用何种分配方式？

答：空闲表法、空闲链表法、位示图法、成组链接法。UNIX系统采用的是成组链接法

26．基于索引节点的文件共享方式有何优点？

答：优点是建立新的共享链接时，不改变文件拥有者关系，仅把索引结点共享计数器加1，系统可获悉了由多少个目录项指向该文件。缺点是拥有者不能删除自己的文件否则会出错。

27．基于符号链的文件共享方式有何优点？

答：能够通过网络链接世界上 任何地方的计算机中的文件。

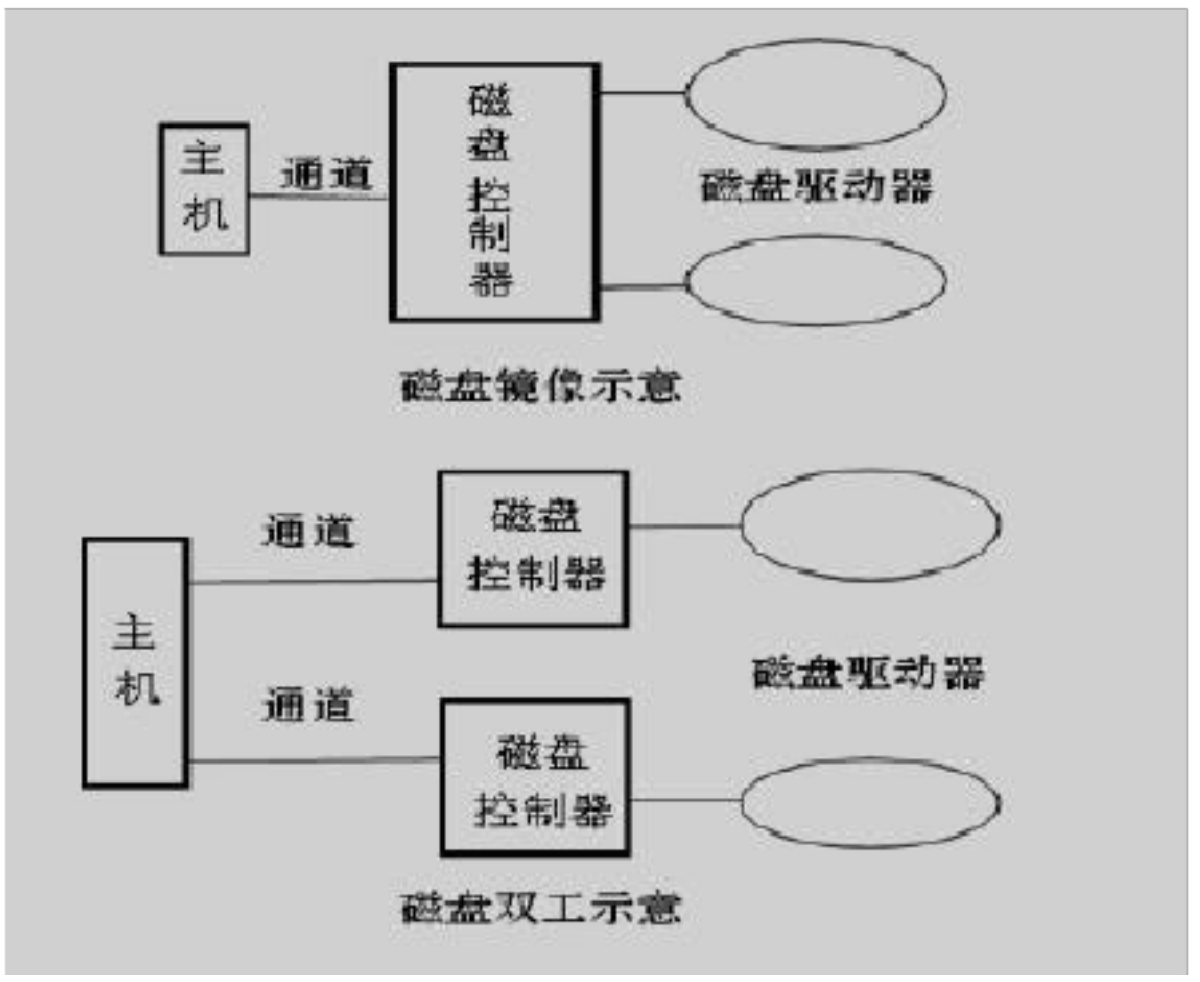
28．在一级系统容错技术中，包括哪些容错措施？什么是写后读校验？

答：一级系统容错技术包括双份目录、双份文件分配表及写后读校验等容错措施。

写后读校验是每次从内存缓冲区向磁盘写入一个数据块后，又立即从磁盘上读出该数据块，并送至另一缓冲区中，再将该缓冲区内容与内存缓冲区中在写后仍保留的数据进行比较。若两者一致，才认为写入成功，继续写下一个盘块。否则重写。若重写后仍不一致，则认为盘块缺陷，便将应写入该盘块的数据，写入到热修复重定向区中。

29．在第二级系统容错技术中，包括哪些容错措施？画图说明之。

答：第二级容错技术包括磁盘镜像和磁盘双工两种容错措施。图示如下：



30．何谓事务？如何保证事务的原子性？

答：事务是用于访问和修改各种数据项的一个程序单位。

要保证事务的原子性必须要求一个事务在对一批数据执行修改操作时，要么全部完成，

用修改后的数据代替原来数据，要么一个也不改，保持原来数据的一致性。

31．引入检查点的目的是什么？引入检查点后又如何进行恢复处理？

答：引入检查点的目的是使对事务记录表中事务记录的清理工作经常化。

恢复处理由恢复例程来实现。首先查找事务记录表，确定在最近检查点以前开始执行的最后的事务Ti。找到Ti后再返回搜索事务记录表，找到第一个检查点记录，从该检查点开始，返回搜索各个事务记录，利用redo和undo 过程对他们进行相应的处理。

32．为何引入共享锁？如何用互斥锁或共享锁来实现事务的顺序性？

答：引入共享锁是为了提高运行效率。在给对象设置了互斥锁和共享锁的情况下，如果事务Ti要对Q执行读操作，只需获得Q的共享锁。如果对象Q已被互斥锁锁住，则Ti必须等待；否则便获得共享锁对Q执行读操作。如果Ti 要对Q 执行写操作，则Ti还要获得Q的互斥锁。若失败则等待；成功则获得互斥锁并对Q执行写操作。

33．当系统中有重复文件时，如何保证他们的一致性？

答：可以采用两种方法：一是对所有的重复文件进行同样的修改，二是用新修改的文件替换所有的重复文件。

34．如何检索盘块号的一致性？检查时可能出现哪几种情况？

答：为了保证盘块号的一致性，先将计数器表中的所有表项初始化为0，用N 个空闲盘块号计数器组成的第一组计数器对从空闲盘块表中读出的盘块号计数，用N 个数据盘块号计数器组成的第二组计数器对从文件分配表中读出的已分配给文件使用的盘快号计数。如果两组计数中的对应数据互补则数据一致，反之则发生错误。

检查时可能出现的情况：

（1）两组计数器中盘块K 的计数值均为0，应在空闲盘块表中增加盘块号K；

（2）空闲盘块号计数器中盘块K 的计数值为2，应删除一个空闲盘块号K；

（3）空闲盘块号计数器中盘块号K 的计数值为0，而数据盘块号计数器中盘块号K 的计数值大于1，则错误严重，存在数据丢失等事件，必须立即报告系统加以处理。