# 操作系统复习要点

1、 概述部分

操作系统概念、特征、设计目标

2、 进程管理部分

进程概念、组成、进程状态迁移图及迁移原因，进程间的关系、临机区概念，实现互斥的方法、P/V 操作，引入线程的目的、线程与进程间的关系、死锁特征、资源分配图判定死锁的方法，常用调度算法。

3、 内存管理部分

作业装入内存的方式，分区内存管理机制中的分区分配方法、特点、快表、分页管理机制原理、实现请求调页的内存管理机制的关键技术

4、 文件管理部分

文件系统设计目标、管理磁盘空闲空间的方法、目录结构、FCB 等

5、 外设管理部分

I/0 软件组成，设备驱动程序概念、四种 I/O 方式比较及其工作流程，设备管理目标。

复习题目

**概述部分**

1、 什么是操作系统？操作系统设计目标是什么？由哪些部分组成？各个部分主要解决什么问题？

操作系统是资源分配器（用来管理和分配资源）、控制程序（控制用户程序和 IO 设备的执行）、内核（一直运行在计算机上的程序）

**设计目标**：

一个运行在电脑用户和电脑硬件之间的程序。

方便使用，并且高效使用硬件。

**组成部分**：

进程管理：进程需要一定资源

主存管理：记录内存正在（被谁）使用，决定进程装入内存，分配释放内存等。

辅存管理：空闲空间分配，存储空间分配，硬盘调度

I/O 管理

文件管理：创建删除文件、目录（提供原语），将文件映射到辅存，备份文件

保护系统：控制程序、进程或用户访问由计算机系统定义的资源

联网

命令解释系统

2、 操作系统内核技术的发展？什么是微内核？并发和并行的区别？

**发展过程**：

| | |
|---|---|
| Batch Systems | （批处理系统） |
| Time-Sharing Systems | （分时系统） |
| Personal-Computer Systems | （PC 系统） |
| Parallel Systems | （并行系统） |
| Distributed Systems | （分布系统） |
| Real -Time Systems | （实时系统） |

**微内核**：

精心设计的、能实现现代 OS 核心功能的小型内核。

不是完整操作系统，只是为构建通用 OS 提供基础，提供基本功能：

进程管理、存储器管理、进程间通讯、低级 I/O 功能

微内核结构以微内核为 OS 核心，以客户/服务器为基础，采用面向对象程序设计的特征，是当今最有前途的 OS 结构。

**进程管理部分：**

1、 为什么要引入进程？为什么要引入线程？从调度性、并发性、拥有的资源以及系统开销等方面，区别和比较进程和线程？

引入进程:提高并发性

引入线程：进一步提高并发性，使得一个进程可以完成多个类似任务，资源共享

**进程两个基本特征**：资源分配的独立单位，调度的基本单位

**引入线程的思想**：将进程资源分配和调度分开，引入线程

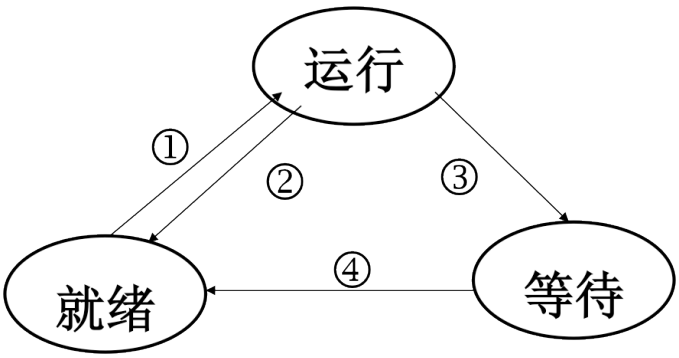
**比较**：

启动一个新的进程必须分配独立的空间，这是一种很“昂贵的”多任务方式

运行于同一个进程中的线程彼此使用相同的地址空间，共享数据，空间代价小

线程间切换的时间也远远小于进程，启动线程时间小于进程

2、 进程状态迁移图，引起状态迁移的原因和事件？




**就绪→执行**:处于就绪状态的进程，当进程调度程序为之分配了处理机后，该进程便由就绪状态转变成执行状态。

**执行→就绪**:处于执行状态的进程在其执行过程中，因分配给它的一个时间片已用完而不得不让出处理机，于是进程从执行状态转变成就绪状态。

**执行→等待**:正在执行的进程因等待某种事件发生而无法继续执行时，便从执行状态变成阻塞状态，如等待 I/O 操作。

**等待→就绪**:处于阻塞状态的进程，若其等待的事件已经发生，于是进程由阻塞状态转变为就绪状态，如 I/O 完成，被中断处理程序唤醒，抢占式内核有高优先级进程就绪。

为什么在转换图中没有就绪到阻塞和阻塞到运行的转换方向？

就绪进程没有占有处理机，也即没有经过运行，其状态就不会改变。

阻塞状态进程唤醒后先要进入就绪队列，才会被调度程序选中，进入了执行状态。

3、 进程组成？PCB 的含义？

**组成**：PCB、数据、代码、栈

**PCB 含义**：进程控制块。每个进程在操作系统中都以进程控制块来表示。

内容：进程状态、程序计数器、CPU 寄存器、CPU 调度信息、内存管理信息、记账信息（CPU 时间、使用时间、时间界限等）、I/O 状态信息

**PCB 表**：系统把所有 PCB 组织在一起并放在内存固定区域就形成了 PCB 表

PCB 表大小决定了系统最多村同时存在的进程数，称为系统并发度

4、进程之间的关系？什么是临界区？如何实现临界区的互斥访问？

竞争、协作。进程同步、进程互斥

临界资源：系统中一次只允许一个进程使用的资源，也成为互斥资源

**临界区（互斥区）**：进程中涉及临界资源的程序段

**访问临界区**：软件方法（设置什么标志位、如何检查标志位），硬件方法

互斥、有空让进、有限等待

5、P/V 操作的含义？信号量的含义？如何定义信号量的初值？如何利用 P/V 操作实现多个进程之间的同步和互斥？如利用其实现单缓冲区的读写问题？如何实现生产者消费者等问题？

**信号量**：信号量表示资源的实体，是一个与队列有关的整型变量。只能通过初始化和 P/V 原语来访问，访问信号量的进程不受进程调度的打断。

**信号量物理含义**：

S>0：表示有 S 个资源可用

S=0：表示无资源可用

S<0：绝对值表示等待队列中的进程个数

**信号量的初值**：

公用信号量：初值为 1，用来实现进程间的互斥

私用信号量：初值为 0 或者大于零的一个正整数，用来实现进程间的同步

**P 原语 P(S)**：执行 P 操作意味着申请分配一个单位的资源

S=S-1

如果 S>=0,则调用 P(S)的进程继续执行

如果 S<0，则调用 P(S)的进程被阻塞，并将其插入到等待信号量 S 的队列当中

**V 原语 V(S)**：执行 V 操作意味着释放一个单位的资源

S=S+1

如果 S>0 则调用 V(S)的进程继续执行

如果 S<=0，则从等待 S 的阻塞队列中唤醒头一个进程到就绪队列中，然后调用 V(S)的进程继续执行

**实现进程互斥**：

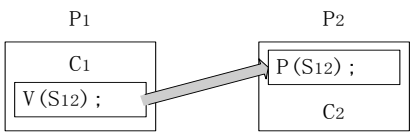
为临界资源设置一个互斥信号量 mutex

在每个进程中，将临界区代码置于 P(mutex)和 V(mutex)原语之间

```
P(mutex);
  critical section
V(mutex);
  remainder section
```

**实现进程同步**：

前驱关系，为每个前驱关系设置一个互斥信号量 S12，初值为 0



**读者写者问题**：

第一类：读者优先，mutex=1，readcount=0

读者：

```
while (true) {
    P(mutex);
        readcount ++;
        if (readcount==1)
            P (w);
    V(mutex);
       读
P(mutex);
        readcount --;
        if (readcount==0)
            V(w);
     V(mutex);
     };
```

写者：

```
while (true) {

    P(w);
      写
V(w);

     };
```

第二类：写者优先

读者：

```
while(true) {
  P(r);
    P(x);
      readcount++;
      if(readcount ==
  1) P(w);
    V(x);
  P(r);
  读
  P(x);
    readcount--;
    if (readcount == 0)
  V(w);
  V(x);
}
```

写者：

```
while(true) {
 P(y);
  writecount++;
  if (writecount == 1) P(r);
 V(y);
 P(w);
 写
 V(w);
 P(y);
   writecount--;
  if (writecount == 0) V(r);
 V(y);
}
```

**生产者消费者问题**：full=0，empty=N，mutex=1

Producer

```
P(empty);
P(mutex);
  one unit --> buffer;
V(mutex);
V(full);
```

Consumer

```
P(full);
P(mutex);
  one unit <-- buffer;
V(mutex);
V(empty);
```

练习 1. 设有 n 个进程共享一个互斥段，则

如果每次只允许一个进程进入互斥段。

如果最多允许 m 个进程（m<n）同时进入互斥段。

问：

所采用的互斥信号量初值是否相同？

信号量值的变化范围如何？

练习 2. 桌上有一只盘子，每次只能放入一个水果。爸爸专　向盘中放苹果，妈妈专向盘中放桔子，一个女儿专等吃盘中的苹果，一个儿子专等吃盘中的桔子。

使用 P/V 操作写出他们能同步进程过程。

练习 3. 一条小河上有一座独木桥（如图），规定每次只允许一个人过桥。现河东和河西都有相等的人数在等待过桥，为了使两边的人都有同样的过桥机会，规定某边的一个人过桥后要让另一边的一个人过桥，即两边的人交替过桥。如果把每个过桥者看做一个进程，为保证安全，可用 PV 操作来管理。

（1）写出应定义的信号量及其初值。

（2）假定开始时让河东的一个人先过桥，然后交替过桥。现进程的程序如下。请在空白处填上适当的 PV 操作，达到上述管理要求。

解答:

独木桥是各进程的共享资源，由于每次只允许一个人过桥，且河两边的人必须交替过桥，因而相互间要互通消息。在本题中应区分“允许河东的人过桥”和“允许河西的人过桥”两个不同的消息。所以，应定义两个信号量 SI 和 SZ 分别与两个消息对应。若开始时让河东的一个人先过桥，则信号量 S1 的初值应为 1，而 S2 的初值应为 0。任何一方的人欲过桥前应调用 P 操作来测试允许过桥的消息是否到达，只有在消息到达后才可过桥，过桥后应调用 V 操作把允许另一方的一个人过桥的消息发送出去。

[题解]

（1）定义两个信号量 S1 和 S2，S1：=1，S2：=0。

（2）假定开始时让河东的一个人先过桥，则用 PV 操作管理时的程序应如下：

```
process E->W；
begin
……
P（S1）；
过桥；
V（S2）；
……
end；
process W->E；
begin
……
P（S2）；
过桥；
V（S1）；
……
end；
```

6、 高级通信方式中，理解 send()和 receive（）的工作过程。

**发送进程**：

发送进程需要发送消息时，执行 send 原语，产生自愿性中断，进入操作系统，操作系统为发送进程分配一个空缓冲区，并将所发送的消息从发送进程 copy 到缓冲区中，然后将该载有消息的缓冲区连接到接收进程的消息链链尾。发送进程返回到用户态继续执

**接受进程**：

在以后某个时刻，接收进程执行到 receive 接收原语时，也产生自愿性中断进入操作系统。操作系统将载有消息的缓冲区从消息链中取出，并把消息内容 copy 到接收进程空间，之后收回缓冲区。完成了消息的接收，接收进程返回到用户态继续进行

7、 有哪些常用调度算法？引起进程调度的事件有那些？多级反馈队列调度算法的分析？

**常用算法**：

先到先服务调度

最短作业优先调度

优先权调度

轮转法调度

多级队列调度

多级反馈队列调度


（1）设置多级就绪队列。系统中有多个就绪进程队列，每个就绪队列对应一个调度级别，各级具有不同的优先级。第 1 级队列的优先级最高，第 2 级队列优先级次之，其余级队列的优先级随级增大而降低。

（2）各级就绪队列具有不同大小的时间片。优先级最高的第 1 级队列中进程的时间片最小,随着队列的级数增大其中进程的优先级降低,但时间片却增大。

（3）一个新进程在系统就绪队列中排队的规则。当一个新进程进入内存后，首先被放到第 1 级就绪队列末尾。该队列中的进程按 FCFS 原则分配处理机，并运行相应于该队列的一个时间片。若进程在这个时间片中完成其全部工作，该进程离开就绪队列撤离系统；若进程运行完一个时间片后仍未完成，则该进程被强迫放弃处理机，放入下一级就绪队列的末尾。

（4）按队列优先级高到低进行进程调度。每次进程调度都是从第 1 级就绪队列开始调度，仅当第 1 级队列空时，调度程序才调度第 2 级队列中的进程；依此类推。第 n 级队列中的进程采用时间片轮转方法进行调度。

（5）一个进程进入较高优先级队列时可能要重新调度。

**CPU 调度**：调度程序从内存就绪可执行进程中挑选一个分配给 CPU

**调度时刻**：

当一个进程从运行状态切换到等待状态

当一个进程从运行状态切换到就绪状态

当一个进程从等待状态切换到就绪状态

当一个进程终止时

8、 引起死锁的四个特征是什么？如何针对这是个特征克服死锁？资源分配图的方法判定死锁？

**特征**：

*互斥*：至少有一个资源必须处于非共享模式，即一次只有一个进程使用。

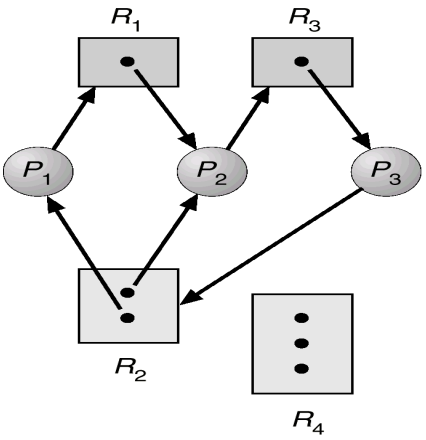
*占有并等待*：一个进程必须占有至少一个资源并等待另一资源，而该资源被其他进程占有

*非抢占*：资源不能被抢占，即只有在进程完成任务后资源才释放

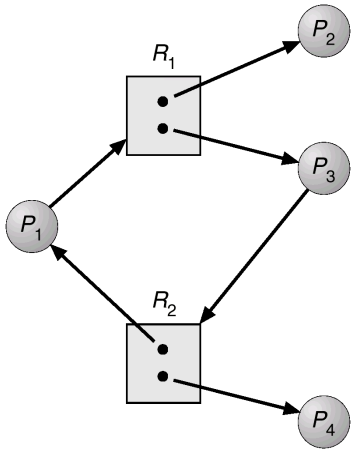
*循环等待*：破坏环来破除死锁

**资源分配图**：

P 表示进程，R 表示资源。R->P 为分配变，P->R 为申请边。




存在死锁




有环但无死锁

如果图没有环，那么系统就没有进程死锁。

如果图有环，那么可能存在死锁。

如果每个资源刚好有一个实例，那么有环就意味着会出现死锁。充要条件

如果每个资源类型有多个实例，那么有环并不意味着已经出现死锁。必要条件

## 内存管理部分

1、 程序装入内存有几种方式？什么是可重定位的装入技术？

**绝对装入技术**：

也成为固定地址再定位。程序地址在定位在执行前就已经确定。

程序地址空间和内存地址空间一一对应。

优点：装入过程简单；缺点：依赖硬件结构，不适合多道系统

**可重定位装入技术**：列出各个需要重定位的地址单元和相对地址值，装入时根据所定位的内存地址修改每个重定位地址项，添加偏移量。

*静态再定位*：装入程序在程序执行之前再定。

容易实现，无需硬件支持；程序定位后不能移动，无法有效利用内存，程序在存储空间只能连续分配

*动态再定位*：程序在装入内存时不修改逻辑地址，在访问物理内存之前再实时地讲逻辑地址转换成物理地址。

程序在执行过程中可以移动，利于内存充分利用，程序可离散存放；需要硬件支持。

2、 存储管理方案

**连续内存分配方式**:

单一连续存储管理

分区存储管理:

基本原理:将内存分为大小相等或者不等的几个分区,每个进程占用一个或几个分区,操作系统占用一个分区。

问题:内外碎片,难以共享。

固定分区存储管理:内外碎片

动态分区存储管理

**离散内存分配方式**:

分页存储管理(单位是页)

段式存储管理(单位是段)

段也是存储管理

3、 在动态分区分配中,有那些分区分配算法?各个是如何实现的?

无内碎片,有外碎片

**最先适配算法**:按照分区先后次序从前往后查找,找到符合要求的第一个分区

实质:尽可能地用低地址空闲区,在高地址保留较大空闲区备用

优点:简单;缺点:前面的空间分得太小,满足分配条件的可能性小

**循环最先适配算法**:按分区先后顺序从上次分配的分区开始查找(到头后再回到开头),找到符合要求的第一个分区

特点:空间分配均匀,但是较大的空闲区不易保留

**最佳适配算法**:在所有大于或等于要求分配长度的空闲区中挑选一个最小的分区

实现:空闲存储区管理表采用从小到大的顺序结构。

优点:较大的区可以保留

缺点:空闲区是按照大小排列的,所以释放的时候需要合并前后几个区

**最坏适配算法**:取空闲区中最大的一块,剩下的分成较小的空闲区

实现:由大到小排序

优点:分配时仅需一次查找,快

缺点:剩余分区越来越小,无法运行大程序

多次分配后许多许多小的空闲块无法满足分配需求,但是总和满足分配需求,叫做碎片。空间浪费。解决:紧缩技术。

4、 页式存储管理

把用户程序按逻辑页分成大小相等的部分,称谓页(虚页)

按页的大小将内存非为大小相等的区域,称为内存块(实页,页框,物理页面)

5、 什么是快表?其中内容是什么样子的?什么是页表?其结构是如何?

**联想寄存器(快表)**:为缩短查找时间,可将页表从内存装入关联寄存器(TLB),按内容查找,即逻辑页号->物理页号。

TLB 是关联型寄存器,条目有两部分组成:键(标签)和值

**页表**:系统为每个进程建立一张页表,页表给出逻辑页号和具体内存块号相应的关系。页表放在内存,属于进程现场信息。

6、 什么是虚拟存储器?其特征是什么?虚拟存储器容量是如何确定的?

虚拟存储器是建立在主存-辅存物理结构基础之上,由附加硬件装置及操作系统存储管理软件组成的一种存储体系。

**特征**:

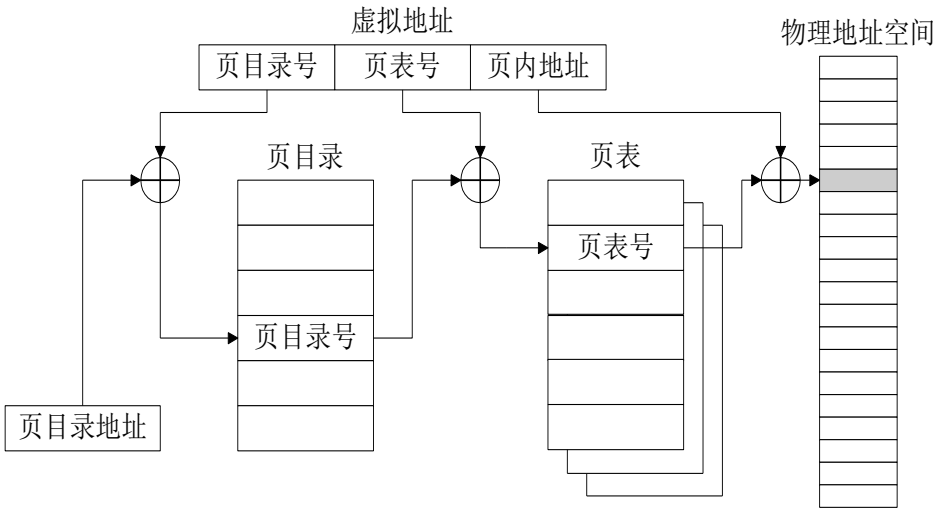
*不连续性*：物理内存分配的不连续性、虚拟地址空间使用的不连续性

*部分交换*

*大空间*

**容量确定**：虚拟存储器的最大容量是由计算机的地址结构确定的，其实际容量是由内存和硬盘交换区容量之和确定的

7、 请求分页技术中，图示 windows 下的两级分页机制？




8、 请求分页机制中，页面置换算法有那些，具体实施页面置换过程？

**先进先出算法（FIFO）**：选择建立最早的页面被置换

性能较差，抖动现象，BELADY 现象

**最佳算法**：选择“未来不再使用的”或“在离当前最远位置上出现的”页面置换

理想算法

**最近最少使用页面淘汰算法（LRU）**：使用离过去最近作为不远将来的近似，置换最长时间没有使用的页。

性能接近最佳算法

**最不常用算法（LFU）**：选择到当前时间为止被访问次数最少的页面被置换

每页设置访问计数器，每当页面被访问时该页面的访问计数器加一；

发生缺页中断时，淘汰计数值最小的页面，并将所有计数清零。

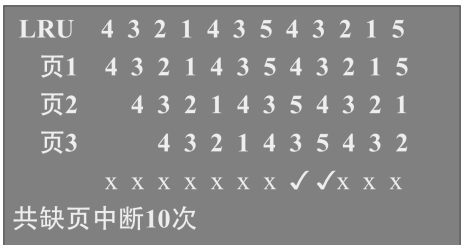
**简单轮转算法（CLOCK）**：

每页标志位 USE，若页被访问则置为 1

置换是采用一个指针，从当前指针位置开始按地址先后检查各页，寻找 USE=0 的页面置换，并将指针经过的页改为 USE=0，最后指针停留在被置换得下一页。

计算缺页次数：某程序在内存中分配三个页面，初始为空， 页面走向为4,3,2,1,4,3,5,4,3,2,1,5






9、 在交换技术中，进程置换策略是什么？

**调入策略**:

缺页中断：只调入发生缺页时所需的页。

优点：容易实现；缺点：对外 I/O 次数多，开销大

预调页：发生缺页时一次性调入该页和相邻的几个页

优点：提高 I/O 效率；缺点：缺点：若调入的页在以后少被访问，效率低

**调出策略**:

请求调出：页面被置换时才调出

缺点：调入之前先调出，慢

预调出：成批调出

缺点：可能不必要开销

**常驻集工作集策略**：分给进程多少物理页面，如何动态调整进程物理页面数

**负载控制策略**：内存驻留多少并发进程较好，尽可能提高并发水平

## 文件管理部分

1、 什么是文件？什么是文件系统？文件系统设计目标是什么？

**文件**：信息以一种单元，即文件形式存储在磁盘或其他外部设备上。文件是一组带标识的、在逻辑上有完整意义的信息项序列。文件是通过操作系统来管理的，文件内容由文件建立者和使用者解释。

**文件系统**：是操作系统中统一管理信息资源的一种软件，管理文件的存储、检索、更新，提供可靠的贡献和保护手段，并且方便用户使用。具体来说：统一管理文件的存储空间，管理空间的分配和回收

用户观点：文件如何呈现和交互

操作系统观点：具体的实现方法，包括存储管理、目录实现等。

**文件系统目标**:

实现文件的按名存取。名空间->存储空间的映射

实现文件信息共享，提供文件保护和保密措施

向用户提供方便使用的接口

系统维护及向用户提供相关信息

文件系统的执行效率

提供与 I/O 的统一接口

2、 什么是文件的逻辑结构、物理结构？文件物理结构有哪些？分别如何实现？有什么特点？

**逻辑结构**：从用户角度研究文件的组织形式

无结构文件（流式文件）：基本单位字符，文件是有逻辑意义无结构的字符集，具有较好的灵活性

有结构文件（记录文件）：文件有若干记录组成，每条记录有其内部结构，每条记录有一个键，可按键查找。

**物理结构**：从系统角度看文件，从文件在物理介质上的存放方式来研究文件

*连续（顺序）结构*：文件信息存放在若干连续物理块当中

简单，支持顺序和随机存储，顺序存储速度快，寻到次数少，寻道时间少；

不能动态增长，预留空间浪费，重新分配移动，外碎片，难插入删除，存储压缩技术

*链接结构*

提高空间利用率，无外碎片，利于插入删除和文件动态扩充；

存取慢，不利随即存储，指针出错不可靠，多寻道次数时间，指针占空间

FAT 是链接结构，静态链表

*索引结构*：文件信息存储在若干不连续的物理块中，系统为每个文件建立索引表，将块的块号放在索引表中。索引表就是磁盘块地址数组，第 i 个条目指向文件第 i 块。

既能顺序存取又能随机存取，动态增长，插入删除，充分利用外存；

较多寻到次数寻道时间，索引表带来额外系统开销

链接模式、多级模式、综合模式（UFS）

| | 逻辑结构 | 物理结构 |
|---|---|---|
| 概念与定义 | 文件自身的组织形式与结构<br>用户所能看到的数据的组织形式 | 文件在物理存储器上的组织形式与结构<br>用户看不见的、根据不同分配方式形成的不同的物理文件 |
| 形式 | 记录式文件<br>流式文件 | 顺序文件<br>链接文件<br>索引文件 |

3、 UNIX 系统采用的综合索引方式是如何实现的？有何优点？

a) 每个文件索引表为 13 个索引项，每项 2 个字节。最前面 10 项直接登记存放文件信息的物理块号（直接寻址）；

b) 如果文件大于 10 块，则利用第 11 项指向一个物理块，该块中最多可放 256 个文件物理块的块号（一次间接寻址）。对于更大的文件还可利用第 12 和第 13 项作为二次和三次间接寻址；

c) UNIX 采用了三级索引结构后，文件最大可达 16 兆个物理块。

优点（自己胡诌的）：支持大文件

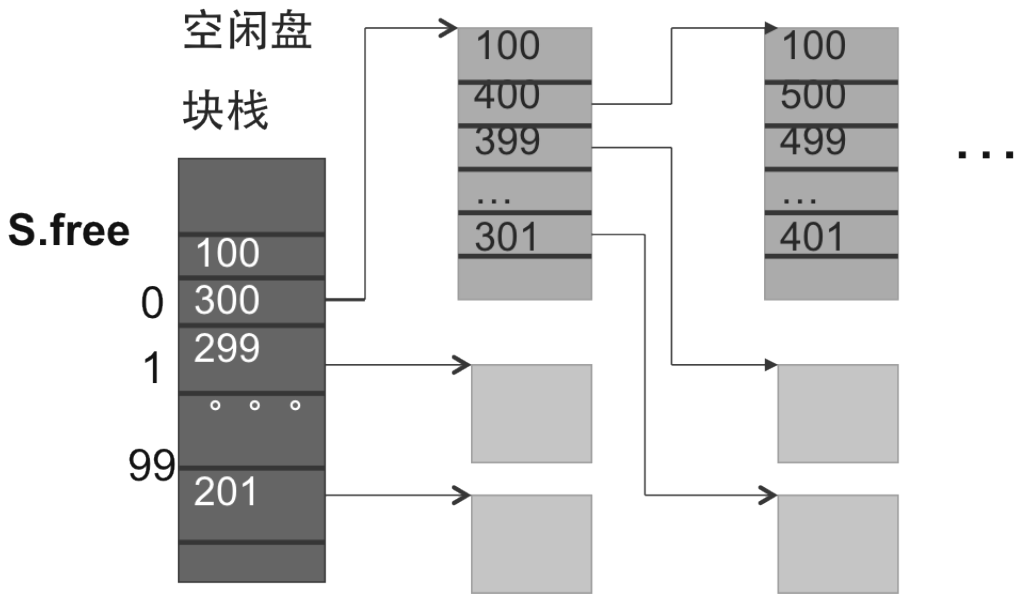
4、 磁盘空闲空间的管理方法？图示成组链接法？并说明其优点。

**空闲块表**：将空闲块记录在一个表中

**空闲块链表**：将空闲块链接成一个链

**位图**：用一串二进制位表示磁盘空间分配，分配的为 1，空闲的为 0.描述能力强，适合各种物理结构

**成组链接法(管理大空间)**：成组链接法是空闲表和空闲链表的结合。将磁盘空闲块分成若干组，如将每 100 个盘块作为一组，用索引表表示：该组空闲块总数和各空闲块块号存入下一组的第一个空闲块中（从后往前分组），各组通过链接指针连在一起形成链表，最后不满 100 块的那组空闲块总数和各空闲块块号记入磁盘区专用管理块（超级块）的空闲盘块号栈的 s_nfree 和 s_free[100]中。




5、什么是目录文件的组成？采用目标文件的目的？目录的改进方法及其改进性能比较？常用的目录结构？

文件控制块：文件控制块是文件存在的标识。

**文件目录**：把所有的 FCB 组织在一起就构成了文件目录，即文件控制块的有续集合。构成文件目录的项目叫做目录项，就是 FCB.

**目录文件**：为实现对文件目录的管理，通常将文件目录以文件的形式保存在外存，这个文件就叫做目录文件。

**采用目录文件的目的**：

高效性：提高文件查找效率

重命名：文件命名更加方便

逻辑组：文件分组更加容易

**常用目录结构**：

一级目录：简单，不允许重名，不利于共享

二级目录：查询速度快，允许重名

树形目录：常用结构。解决重名等问题，提高文件检索速度，但是每个文件都在外村，多次访盘影响速度。

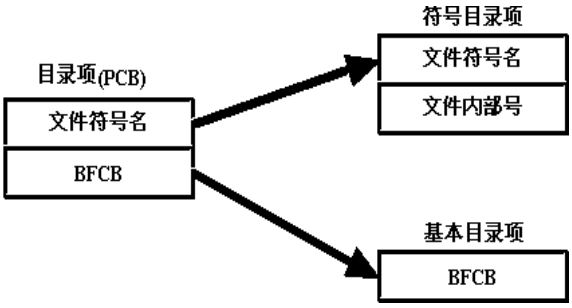
其他：哈希表、B+树

**改进**：目的是加快目录检索速度。

目录项拆解法：把 FCB 分解为两部分

符号目录项（次部）：文件名、文件号（这个文件号相当于一个指针）

基本目录项（主部）：其他信息




一个 FCB 有 48 个字节，符号目录项占 8 字节，文件名 6 字节，文件号 2 字节，基本目录项占 48-6=42 字节。假设，物理块大小 512 字节，目录文件有 128 个目录项。

*改进前*：1 个物理块含 512/48=10 个 FCB,则该目录文件占 13 个物理块；

*改进后*：1 个物理块含 512/8=64 个符号目录项或 512/42=12 个基本目录项，则该目录文件符号文件占 2 块，基本文件占 11 块。

查找一个文件的平均访盘次数

*改进前*：(1+13)/2=7 次

*改进后*：(1+2)/2 +1 =2.5 次

注意：之所以减 6 而不是 8 是因为在基本目录项当中也有文件号，文件号是在将 FCB 分成两部分时冗余出来的，用来作为指针。

减少了访盘次数，提高了检索速度。

6、 RAID 的概念？关键技术是什么？

技术特点：并行交叉存取。

通过冗余改善可靠性，通过并行处理改善性能。

7、 文件操作中,open 函数实现过程及其完成的内容？

1）根据文件路径名查目录，找到 FCB 主部；

2）根据打开方式、共享说明和用户身份检查访问合法性；

3）根据文件号查系统打开文件表，看文件是否已被打开；是→共享计数加 1，否则→将外存中的 FCB 主部等信息填入系统打开文件表空表项，共享计数置为 1；

4）在用户打开文件表中取一空表项，填写打开方式等，并指向系统打开文件表对应表项。

5）返回信息 fd：文件描述符，是一个非负整数，用于以后读写文件(找到 FCB 之后就不再使用文件名了，而是用 fd，file descriptor)。

8、 影响磁盘访问的因素有那些？列举几种磁盘调度算法？

***磁盘读取时间***：

寻道：磁头移动定位到指定磁道

旋转延迟：等待指定扇区从磁头下经过

数据传输：数据在磁盘和内存间实际传输

***存取时间***：一次访盘时间=寻道时间+旋转延迟+存取时间。减少寻道、延迟时间

***磁盘调度算法***：

*先来先服务（FCFS）*：按访问请求到达先后顺序服务。简单公平；效率不高。

*最短寻道时间优先（SSTF）*：优先选择距离当前磁头最近的访问请求服务，主

要考虑寻道优先。改善磁盘平均服务时间；饥饿现象。

*扫描算法（SCAN）*：磁臂从磁盘一端移向另一端，当经过柱面时处理柱面上的请求。当到达另一端时，改编方向继续。来回扫描。

*C-SCAN*：总是从 0 号柱面开始。

*LOOK*：类似 SCAN 和 C-SCAN，但是不是到头变向，而是当遇到请求就变向。

**设备管理部分：**

1、 说明操作系统进程设备管理的初衷/目标？

管理目标

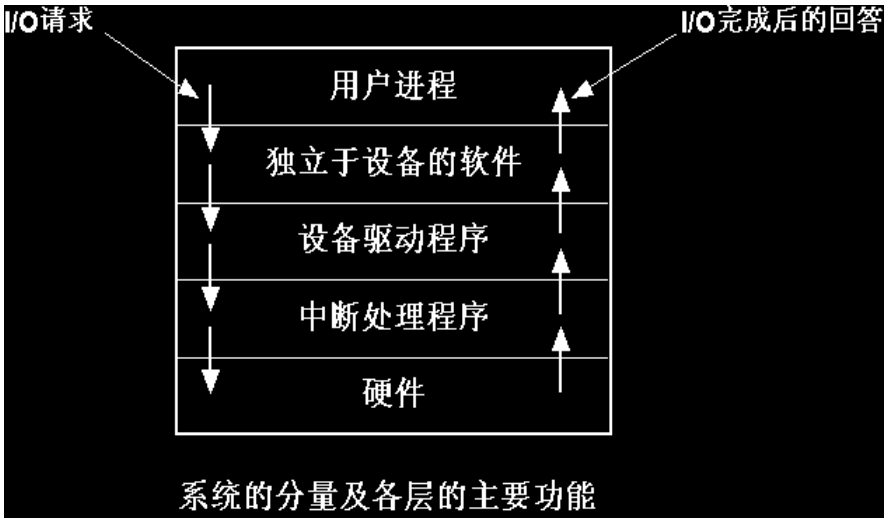
a) 完成设备与内存数据交换，完成用户 I/O 请求。
b) 向用户提供方便的使用外部设备的接口；
c) 提高 CPU 与设备、设备与设备间的并行工作能力；
d) 多个进程竞争使用设备时，按一定策略分配和管理各种设备，使系统能有条不紊的工作；
e) 保证设备数据是安全的、不被破坏的、保密的；
f) 与设备无关性（设备独立性）

2、 设备管理中，I/O 软件的组成？什么是设备驱动程序？其主要完成什么功能？介绍 I/O 管理的流程。

IO 软件：I/O 软件的基本思想是按照分层的思想构成，较低层的软件要使较高层的软件独立于硬件的特性，较高层的软件则要向用户提供一个友好的、清晰地、简单的、功能更强的接口。

中断处理程序
设备驱动程序
设备独立的 I/O 软件（核心）
用户空间的 I/O 软件

设备驱动程序：与设备密切相关的代码放在设备驱动程序当中，每个设备驱动程序处理一种设备类型（与设备相关）。每一个控制器都有一个或者多个设备寄存器，用来存放向设备发送的命令和参数。设备驱动程序负责释放这些命令，并监督他们正确执行。一般设备驱动程序接受上层与设备无关的软件的请求并执行这个请求。



（1）用户进程层执行输入输出系统调用，对 I／O 数据进行格式化，为假脱机 I/O 作准备

（2）独立于设备的软件实现设备的命名、设备的保护、成块处理、缓冲技术和设备分配

（3）设备驱动程序设置设备寄存器、检查设备的执行状态

（4）中断处理程序负责 I／O 完成时，唤醒设备驱动程序进程，进行中断处理

（5）硬件层实现物理 I／O 的操作

3、 I/O 硬件技术中，数据传输技术有那些？分别介绍？并说明中断技术和 DMA 技术的区别与联系？

轮询方式（程序控制 IO PIO）、中断方式、DMA、通道方式

**中断方式：** CPU 向外设发出命令后转去做其他工作，当数据到达数据寄存器后向 CPU 发出中断请求 CPU 服务，CPU 进行下一步数据传输。在 I/O 设备输入每个数据的过程当中无需 CPU 干预，仅当传输完一个数据之后 CPU 花费一点时间来完成中断处理。但由于 I/O 操作仍然受 CPU 控制，每传输一个字节都需要发出中断，浪费了 CPU 时间。

**DMA 方式：**数据传输的基本单位是数据块，所传数据直接送入内存不经过 CPU，仅在一个或者数个块结束的时候才中断 CPU。DMA 需要 CPU 指定好读入数据的大小和存放的位置，采用窃取周期的方式（具有较高的优先级）。

**通道方式：** 通道是一个用来控制外部设备工作的硬件机制，相当于一个功能简单的处理机。通道是独立于 CPU 的、专门负责数据的输入输出传输工作的处理器，它对外部设备实统一管理，代替 CPU 对 I/O 才做进行控制，从而使 I/O 操作可以与 CPU 并行工作。

其他一些点：

- I/O 独立编址
  - 所有端口的地址空间是完全独立的;
  - 主机使用专门的 I/O 指令对端口进行操作;
  - 优点
    - ✓ 外部设备不占用内存的地址空间;
    - ✓ 易于区分是对内存操作还是对 I/O 端口操作
  - 缺点
    - ✓ I/O 端口操作的指令类型少，操作不灵活
- 存储映像编址
  - 分配给系统中所有端口的地址空间与内存地址空间统一编址；
  - 主机把 I/O 端口看作一个存储单元，对 I/O 的读写操作等同于对存储器的操作。
  - 优点
    - ✓ 凡是可对存储器操作的指令都可对 I/O 端口操作
    - ✓ 不需要专门的 I/O 指令
    - ✓ I/O 端口可占有较大的地址空间
  - 缺点
    - ✓ 占用内存空间