

# 第5章 文件系统

文件系统是操作系统中**最为可见**的部分。

文件系统：一组文件、一个目录结构

- 5.1 文件和文件系统**
- 5.2 文件目录结构**
- 5.3 文件的逻辑结构和存取方法**
- 5.4 文件的物理结构和存储介质**
- 5.5 文件记录的组块与分解**
- 5.6 文件存储器存储空间的管理**
- 5.7 文件的共享与保护**
- 5.8 文件的操作命令**
- 5.9 文件系统的组织结构**
- 5.10 存储器映射文件**

# 5.1 文件和文件系统

## 一、文件系统的引入

- 计算机的重要作用就在于它能够以极快的速度处理大量的信息。而要进行数据的处理，必须同时解决信息的组织与存取的问题。
- 信息的组织又分为逻辑组织与物理组织，前者成为今天计算机学科中的一门基础学科-----数据结构，后者与存取方法紧密结合，并由OS的信息管理逐步发展到数据库系统。

- 现代OS提供了文件系统----存取和管理信息的机构，由管理文件所需的数据结构和相应的管理软件，以及访问文件的操作所组成。
- 配置了文件系统后，用户就可以通过文件名字，使用直观的文件操作命令，按照信息的逻辑关系去存取他所需要的信息，从而使用户摆脱了存取介质的特性和I/O指令的细节。从这个意义上讲，文件系统提供了用户与外存的接口。

# 1. 文件

- 文件是存储在外部存储器上的具有符号名的相关信息的集合。
- 文件可以表示范围很广的对象，一个源程序、目标程序、一批数据、各种语言的编译程序、各种编辑程序、银行的各种帐目、公司的各种记录等。

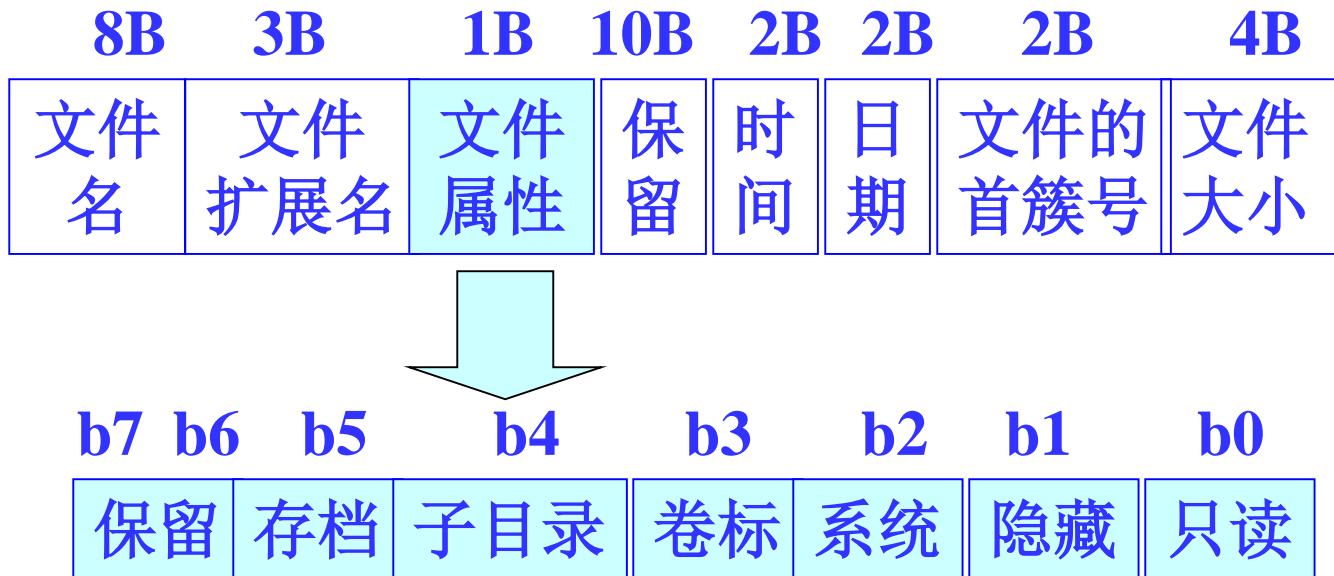
# 文件具有三个基本特征

- (1) 文件的内容为一组信息；
- (2) 文件具有保存性，它可以被放在磁盘、磁带、光盘等存储介质上，且可以被长期保存和多次使用。
- (3) 文件可按名存取，每个文件都有唯一的标识符。可以通过这个标识符来存取文件中的信息，而无须了解文件在存储介质上的具体位置。

# 文件控制块FCB

- 文件：文件控制块、文件体。
- **文件控制块**：包含了文件的说明信息和管理控制信息。如，文件名、文件标识、类型、存储位置、大小、保护方式、创建或修改日期等。
- 操作系统通过文件控制块管理文件。它将文件控制块保存在**目录文件**中，每个文件占用一个目录项。

# DOS文件目录项 (32B)



**b3**为1表示该目录项是盘卷标目录项，此时文件名是卷标名。 **b4**为1表示登记的是一个子目录文件。  
**b5**为1表示文件是新创建或最近被修改过，需要保存。

## 2. 文件分类

### ■ 按用途分类：

- ① 系统文件：由系统软件构成的文件。大多数系统文件只允许用户调用，不允许读写。
- ② 库文件：系统提供给用户使用的各种标准过程、函数和应用程序等。用户可调用。
- ③ 应用程序文件
- ④ 用户文件：用户委托文件系统保存的文件。

- 按文件的保护方式分：只读文件，读/写文件，无保护文件。
- 按信息的流向分类：
  - 输入文件。读卡机或键盘上的文件，只能读入。（输入设备上的文件）
  - 输出文件。如打印机上的文件，只能写出。
  - 入/出型文件。磁盘文件、磁带文件。

## ■ UNIX系统中的文件分类

- 普通文件：通常的文件。
- 目录文件：由文件目录构成的一类用来维护文件系统结构的文件。对其处理同普通文件。
- 特别文件：输入设备和输出设备（字符型特别文件），输入/输出型设备（字符块特别文件），管道文件。

# UNIX的特别文件

从使用上看，与普通文件相同，都要查找目录、验证使用权限、进行读或写等。只是系统把对特别文件的操作转为对不同设备的操作。

# 文件分类的目的

- 对不同文件进行管理, 提高系统效率;
- 提供用户界面友好性

文件的命名规则随着系统的不同而异，但几乎所有的操作系统允许用1~8个字符作为合法的文件名。

现代操作系统都允许文件名最长255个字符。

### 3. 文件系统

- **文件系统：**OS中管理文件的软件机构。  
包括管理文件所需的数据结构、相应的管理软件和被管理的文件。
- 用户只需给出一个文件名，文件系统就能自动地找到文件所在位置，实现对文件的各种操作。

# 文件系统的功能

1. 管理文件存储器。记录空间使用情况，分配空间，调整或回收空间。
2. 实现按名存取。利用目录结构快速定位文件。
3. 应具有灵活多样的文件结构和存取方法，便于用户存储和加工处理信息。
4. 提供一套使用方便、简单的操作命令。
5. 保证文件信息的安全性。
6. 便于文件的共享。

引入文件和文件系统之后，用户就可以用统一的观点看待各种文件存贮介质上的信息。

用户所要知道的只是文件名和文件的特征信息，如文件名，允许的访问权限等。

文件在外存空间的分配和文件存放的物理位置等一些物理操作均由系统自动实现。

## 5.2 文件目录结构

- 引入文件系统的主要目的是使用户实现按名存取文件。
- 文件控制块(FCB)——目录表项
- 文件目录可分为：
  - 一级目录
  - 二级目录
  - 多级目录

文件名	索引节点号
fileA	11
fileB	15
fileC	12

# 1. 一级目录结构

- **一级目录结构：**在整个文件系统中只建立一张目录表，每个文件占据其中的一项。
- **创建文件：**在目录表中增加一个新目录项。
- **删除文件：**删除文件对应的目录项。
- **优点：**简单，易实现。
- **缺点：**查找目录需时间长；易重名。

## 2. 二级目录结构

- 为每个用户建立一个独立的用户文件目录。
- 主文件目录：记录各用户名及用户文件目录所在的物理地址。
- 优点：可解决文件重名问题，可获得较高的查找速度。
- 缺点：当文件较多时，存取速度仍然较慢。无法实现文件共享。

# 主目录

用户A	A目录所在地址
用户B	B目录所在地址
⋮	⋮

# 二级目录结构图

## 用户A的目录表

Fa1的目录项

Fa2的目录项

⋮

## 用户B的目录表

Fb1的目录项

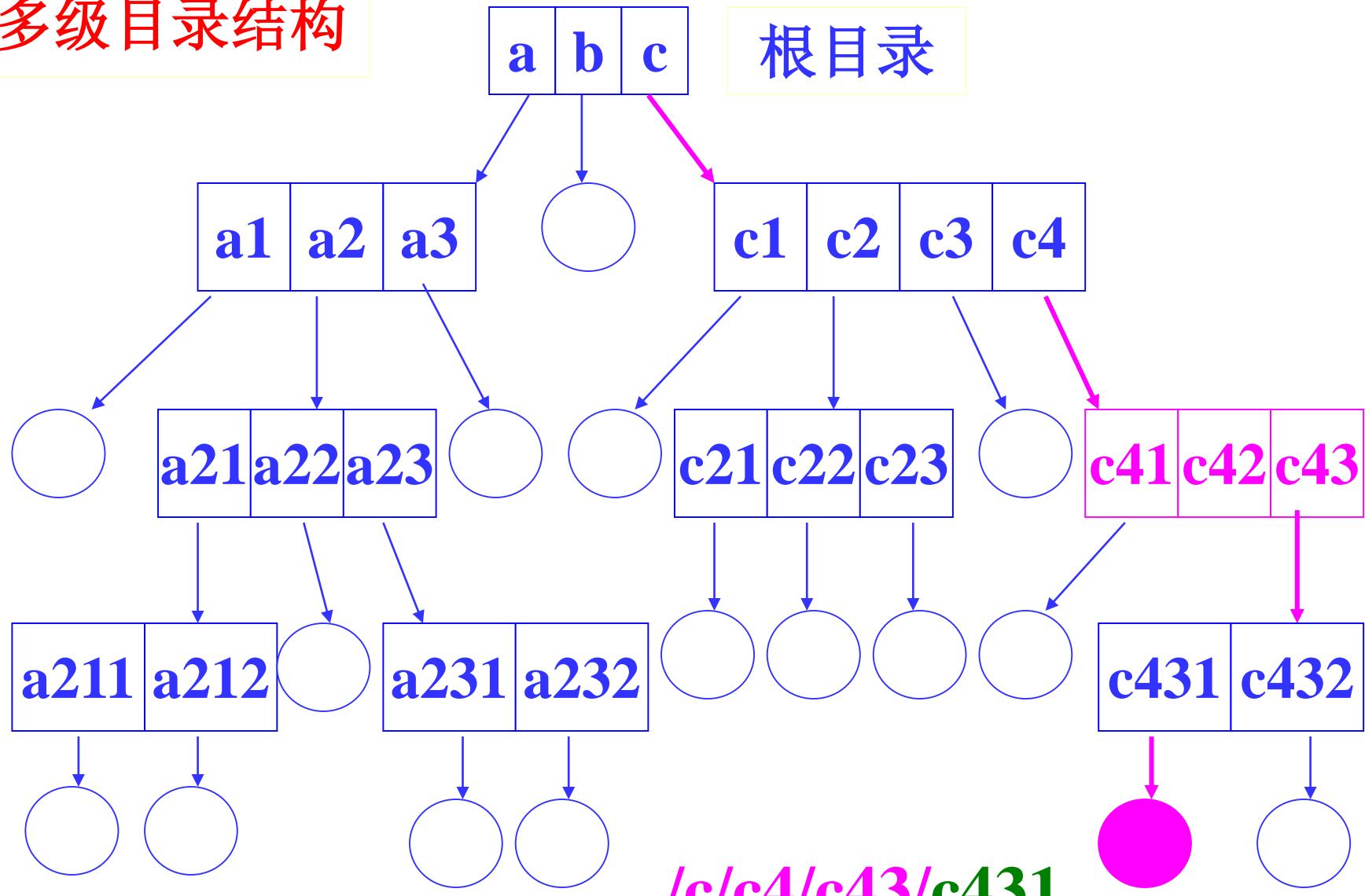
Fb2的目录项

⋮

### 3. 多级目录结构

- 多级目录结构：树形目录结构。
- 完全路径名//绝对路径名：是由根到文件通路上所有目录与该文件的符号名拼接而成的。
- 当前目录//工作目录：用户根据自己的工作需要，在一定时间内，指定某个目录为当前目录。
- 文件的相对路径：从当前目录出发的路径。

## 多级目录结构



**优点：**层次结构清晰，便于管理和保护；  
有利于文件分类；解决重名问题；提高  
文件检索速度；能够控制存取权限。

**缺点：**查找一个文件需要按路径名逐层检  
查多级目录，需多次访盘，影响文件的  
访问速度。

# 目录结构

- 纯树形目录结构禁止共享文件和目录。
- 无环图目录结构允许共享文件和子目录。共享文件只有一个副本。
- 在创建新链接时要避免环，有环的操作复杂些。
- 实现共享文件和目录的方法。
  - 符号链接：创建一个新目录项，其中存有指向另一个文件或目录的绝对路径名。
  - 硬链接：在共享目录项中简单地重复被共享文件的信息，因此两个目录项完全相同。

## 5.3 文件的逻辑结构和存取方法

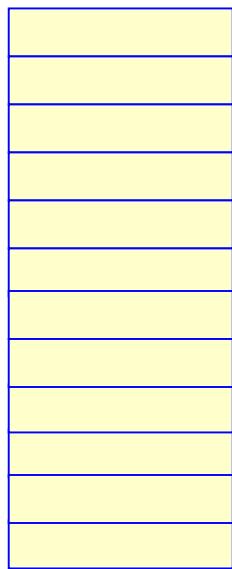
- 文件结构是指文件的组织形式。
- **文件的逻辑结构：**从用户观点出发所看到的文件组织形式。与存储设备特性无关。
- **文件的物理结构：**文件在外存上的存放组织形式。与存储设备的特性有很大关系。
- **文件系统的重要作用之一：**就是在用户逻辑文件和物理文件之间建立映射，实现二者之间的相互转换。

# 文件的逻辑结构

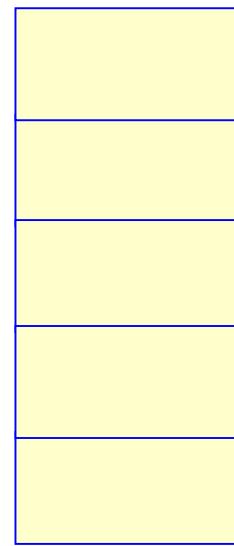
- 无结构的字节流式文件。由无结构的先后到达的相关字节组成，其文件长度就是所包含的字节个数。
- 有结构的记录式文件。分为定长记录式文件和变长记录式文件。

# 文件的逻辑结构

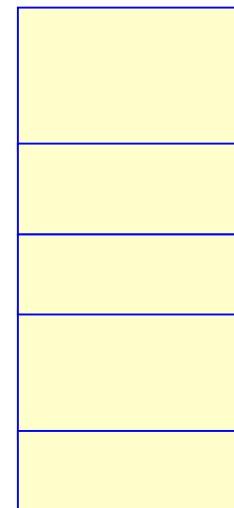
字节  
↓



(a) 字节流文件



(b) 定长记录  
式文件



(c) 变长记录  
式文件

- 通常，操作系统不了解也不关心文件的内容，它所“看到的”都是字节流。文件信息的实际意义是由用户级程序强加和关心的。
- UNIX、MS-DOS和Windows等都是以无结构的字节流形式来组织文件和处理文件的。

# 文件的存取方法

- (1) 顺序存取：按照文件信息的逻辑顺序依次存取。是在前一次存取的基础上进行的。
- 在存取过程中总有两个位置指针指向其中要读写的位置。适用于顺序访问设备（磁带）和随机访问设备（磁盘）。

按照文件信息的逻辑顺序依次存取。

- 在记录式文件中，顺序存取反映为按记录的排列顺序来存取。例如：为了存取记录 $R_i$ ，必须先经过记录 $R_0$ 、 $R_1$ 、...、 $R_{i-1}$ 。
- 在流式文件中，顺序存取反映为当前读写指针的变化，在存取完一段信息之后，读写指针自动加上这段信息的长度，以便指出下次存取时的位置。

## (2) 直接存取（随机存取）

- 基于文件的磁盘模型，磁盘允许对任意文件块进行随机读和写。
- 对记录式文件而言。根据记录的编号来直接存取文件中的任意一个记录。
- 对字节流文件而言。根据系统调用命令把读/写指针调整到欲读/写位置上，然后读/写指定字节数的信息。

### (3) 按键采取

每个记录有一个键，可按键进行查找。

按键存取就是按给定的字段值进行存取。它被广泛地用于数据库系统。查找方法（**搜索算法**）就是数据结构中介绍的顺序法（**线性搜索**）、二分法和散列法等。

# 按键存取中的有关搜索算法

## (1) 顺序法（线性搜索）

- 它是一种最简单、最直观的搜索方法。它从第一个键或记录开始，依次和所要搜索的键或记录相比较，直到找到所需要的记录为止。
- 该算法的搜索效率较低，在文件中记录个数较多时不宜采用。

## (2) 散列法

- 它被广泛用于现代OS的数据查找。
- 该方法的核心思想是定义一个散列函数 $h(k)$ ，使得对于给定的键 $k$ ，散列函数 $h(k)$ 将其变换为 $k$ 所对应的逻辑地址。

## (3) 二分搜索法

- 对于顺序结构排列的键或记录来说，该方法具有较高的搜索效率。

- 该方法首先把所要搜索的键与队列的首尾键比较，如果和其中之一相等，则返回所搜索到的键的逻辑位置；
- 否则，再与队列1/2处的键比较，如果所要搜索的键正好等于该键的话，则返回该键的逻辑地址；

- 否则，如果所要搜索的键小于位于队列中央的键的话，则继续搜索左边的半个队列。如果所要搜索的键大于队列中央的键的话，则继续搜索右边的半个队列。
- 这样，每次用以中央键划分的部分组成新的队列反复进行上述搜索操作，直到找到为止。

# 5.4 文件的物理结构和存储介质

## 5.4.1 文件的物理结构

- **文件的物理结构：**文件在外存上的存放组织形式。
- **物理块：**把文件存储空间划分成若干大小相等的块。物理块是分配及传输信息的基本单位。
- 一个物理块包括一个或几个连续扇区。
- **逻辑块：**把文件信息划分成与物理块大小相等的逻辑块，方便管理。

对于无结构的字符流文件，每个物理块存放相等长度的文件信息。

对于记录式文件，无论是定长还是变长记录，一个记录的长度一般与物理块的大小不等。

在文件系统中，允许一个物理块包含几个记录，也允许一个记录占用几个物理块。

为了讨论简单，下面我们都**假定逻辑记录长度与物理块大小相等。**

为了适应用户的应用要求，文件的物理结构基本上分为连续、链接和索引等。

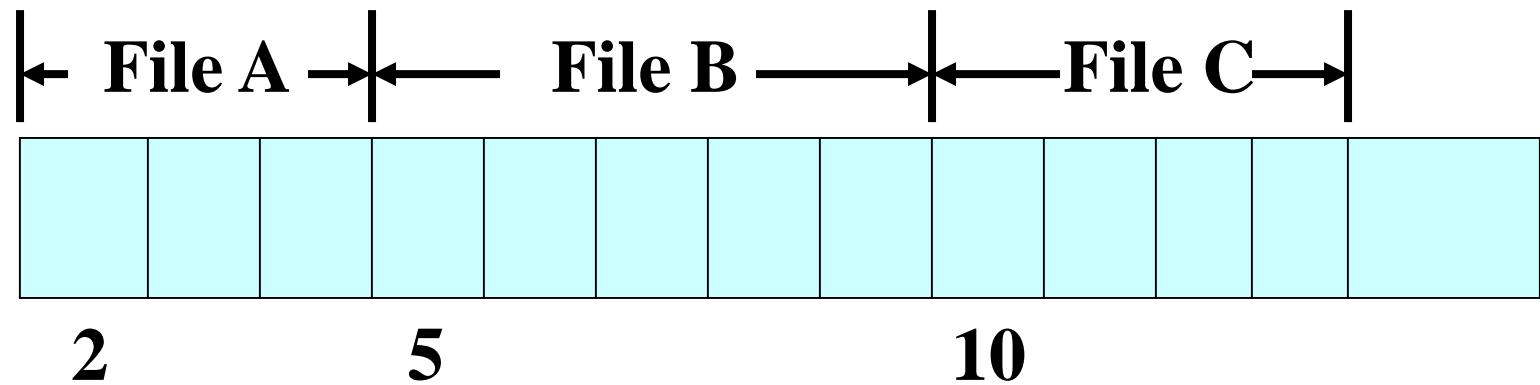
1. 连续文件（顺序文件）
2. 链接文件
3. 索引文件
4. 索引顺序文件

# 连续文件

- 文件内容连续存放。
- 优点
  - 简单。
  - 支持顺序存取和随机存取。
  - 存取速度快。只要访问一次文件的管理信息，就可方便地存取到任一记录。

# 文件目录表

文件名	开始块号	长度
File A	2	3
File B	5	5
File C	10	4



## ■ 缺点

- 不灵活。要求在文件创建时，就给出文件的最大长度。
- 容易产生碎片。由于不断地创建和删除文件，文件存储空间可能出现许多小的无法利用的空洞。

## ■ 连续结构适合存储长度不变的系统文件。

# 链接（或串联）文件

不要求文件内容连续存放。把文件所占用的物理块用链接指针链接起来。

- 优点：可以解决外存的碎片问题，提高了外存空间的利用率；允许文件动态增长。
- 缺点：只能按文件的指针链顺序存取，查找效率较低。

为了克服采用连续结构的缺点，链接(或串联)文件是把逻辑上连续的信息文件存贮在不连续的物理块中，存放信息的物理块中另设一个指针指向下一个物理块。文件最后一个物理块的指针通常为0，以指示该块是链尾。

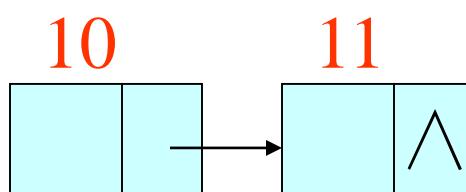
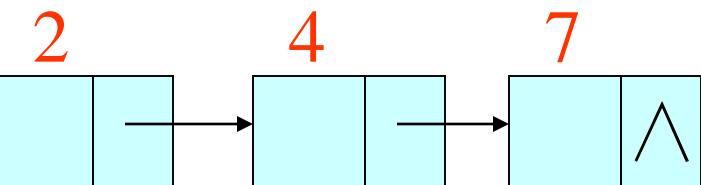
为了克服链接结构文件的缺点，可以把指针字从文件的各物理块中取出，放在一个表中，并将此表叫盘文件映射表。MS-DOS就使用这种方式分配和管理磁盘空间，并将该表叫做文件分配表。

利用文件分配表，不但能方便地实现顺序存取，而且也很容易实现随机存取。

存取任一记录时仍沿链查找，但由于该表在主存，所以不必访问磁盘就能很快定位一个记录的位置。

## 文件目录表

文件名	起始块号	长度
A	2	3
B	5	5
C	10	2



物理块号

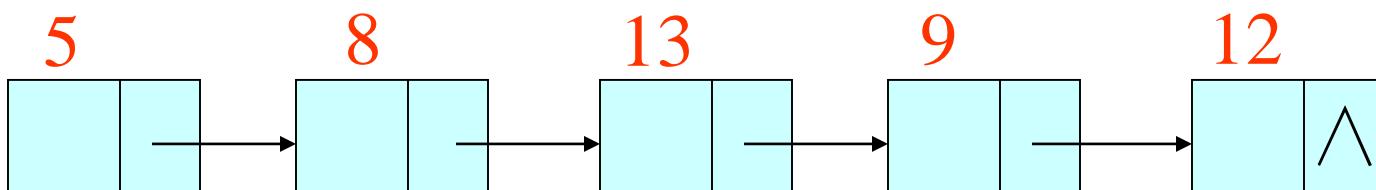
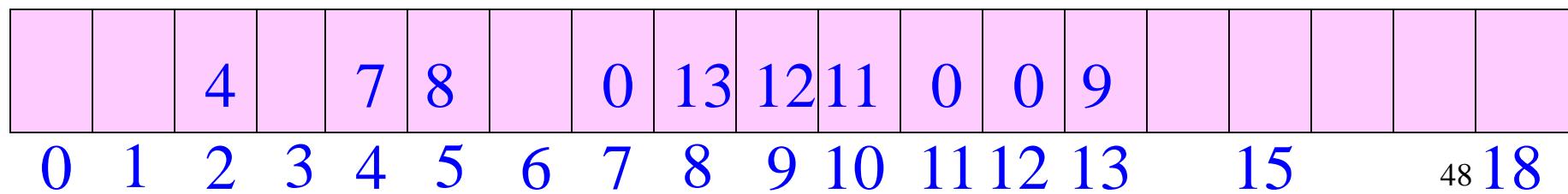


图5.3 链接结构的文件组织

把指针字取出放在索引表中



# 克服链接文件不能随机存取的缺点

索引表长度就是文件存储器能够划分的块数

MS-DOS就是使用文件分配表（FAT）来分配和管理磁盘空间的。

- **优点：**系统运行时，文件分配表在主存，可以顺序访问，也可以随机访问外存的文件。
- **缺点：**运行时整个表必须在主存，主存消耗大。

## 这种方法的主要缺点

在系统工作期间，整个表必须在主存。占用内存太多。

以10G磁盘为例，若每块按512个字节计，则这个表将占用20 M项。每项至少占用3个字节。因此，这个表要占用60M字节的主存。

为了节省主存，盘文件映射（分配）表也必须作为一个文件放在磁盘上。此时要求存放一个文件的各个盘块不要过于分散。否则，可能要读取多个存放盘文件映射表的物理块，才能找到它们之间的对应关系，从而减少了它的优越性。

# 索引文件

- 为每个文件建立一张索引表。用索引表记录文件内容的存放地址，即记录文件的逻辑块号和对应的物理块号之间的关系。
- 例子： UNIX、Linux

连续和链接结构文件存在许多问题，为了实现文件的随机存取，在文件目录表中为每个文件保留一个索引表块号，该索引块指出文件的逻辑块与物理块的映射关系。

## 文件目录表

## 文件B的索引块24

文件名	索引表块号	文件长度
A	14	3
B	24	5
C	17	2

0	5
1	8
2	13
3	9
4	12

- 在访问文件时，先将索引表调入内存，避免访问索引文件时两次访问外存。

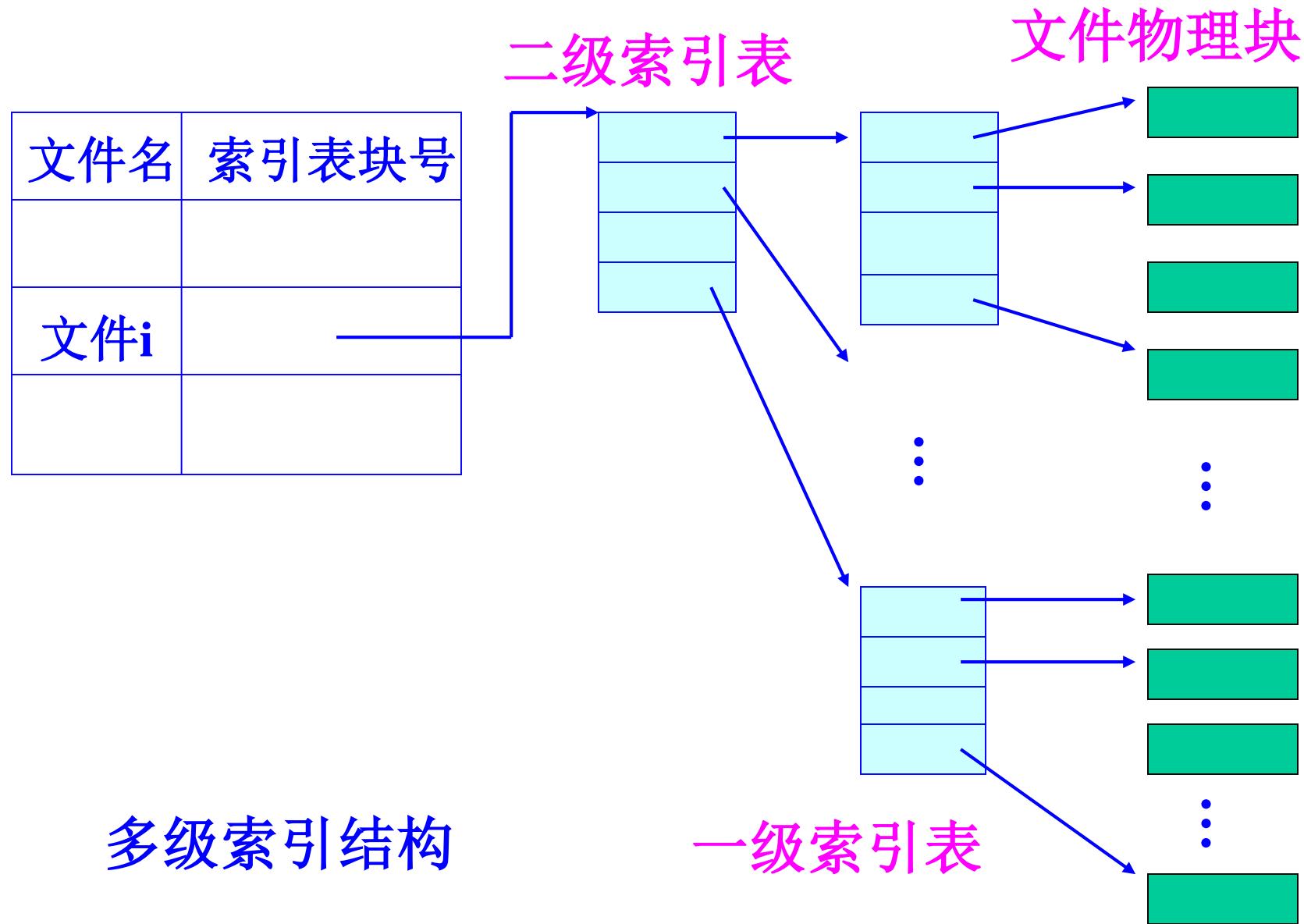
➤ 为了更有效地使用索引表，避免访问索引文件时两次访问外存（一次访问索引表确定文件信息所在的物理块号，再以物理块号获得所需要文件信息），可以在访问文件时，先将索引表调入内存中，这样文件的存取就只需一次访问外存了。

- 当文件很大时，文件的索引表也会很大。
- 如果索引表的大小超过了一个物理块时，可以将索引表本身作为一个文件，再为其建立一个“索引表”，这个“索引表”作为文件索引的索引，从而构成了二级索引。**第一级索引表的表目指向第二级索引，第二级索引表的表目指向相应信息所在的物理块号。**
- 以此类推可再逐级建立索引，进而构成多级索引。

- 当索引表的大小超过了一个物理块时，需要对索引表本身再建立索引，即建立二级索引。

**优点：**文件可动态修改；随机、顺序存取。

**缺点：**索引表的使用增加了存储空间的开销；降低了文件的存取速度。



# 索引顺序文件

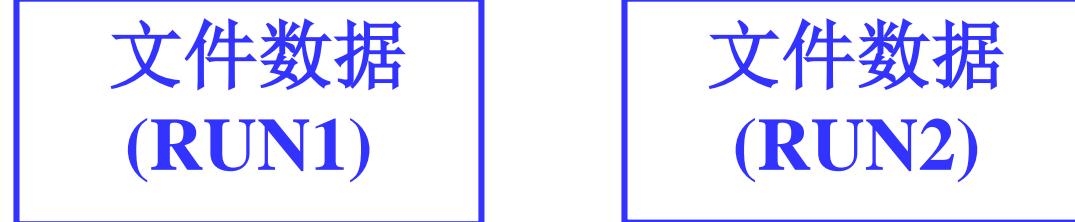
- Windows的NTFS文件系统
- MFT（主控文件表）
- 索引表

# 索引表

开始的VCN	开始的LCN	簇数
0	127	4
4	150	4



VCN: 0 1 2 3 4 5 6 7



LCN: 127 128 129 130 150 151 152 153

- ❖ DOS系统的文件采用链接结构。
- ❖ UNIX系统的文件采用多级索引结构。
- ❖ Linux的Ext2采用多级索引结构。
- ❖ Windows的NTFS采用索引顺序结构。

## 5.4.2 文件的存储介质

- **磁带**: 是一种典型的顺序存取设备, 这种设备只有在前面的物理块被存取访问过之后, 才能存取后续物理块的内容。
- **磁盘、光盘**: 是典型的直接存取设备, 允许文件系统直接存取磁盘上的任意物理块。
- 存储容量大, 存取速度高, 都是以**块**为单位进行信息存储和传输的设备。

# 存取设备、存取方法、物理结构 之间的关系

存取设备	磁盘			磁带
物理结构	顺序结构	链接结构	索引结构	顺序结构
存取方法	直接/顺序	顺序	直接/顺序	顺序
文件长度	固定	可变/固定	可变/固定	固定

磁盘：寻道时间，旋转延迟时间，读写传输时间

# 文件卷的结构 (以DOS系统为例)

磁盘低级格式化；硬盘分区；分区格式化

- ① 磁盘低级格式化：将磁盘划分成若干磁道、扇区。每个扇区512B，扇区的头标记录其柱面号、磁头号和扇区号。由生产厂家完成。

## ② 硬盘分区：用FDISK命令实现。

- 硬盘主引导扇区指的是硬盘的0面0道1扇区，是用FDISK进行硬盘分区时产生的。
- 硬盘主引导扇区，共512（200H）B。
- 它属于整个硬盘而不属于某个独立的分区。

## ③ 分区格式化：用FORMAT命令。对分区进行具体的数据组织，即制作文件系统。

- DOS引导扇区在DOS分区的第一个逻辑分区，是用FORMAT命令产生的

# 硬盘主引导扇区

1. 硬盘主引导程序：位于该扇区的0-1BDH处。
2. 硬盘分区表：位于1BEH-1FDH处，每个分区表占用16B，共有4个分区表。**16B意义如下：**0为自举标志，80H为可引导分区，00H为不可引导分区；1-3是分区的起始地址；4是分区类型（DOS分区）；5-7是分区的结束地址；8-11是分区首扇区的绝对扇区号；12-15是分区占用的总扇区数。
3. 引导扇区的有效标志：位于1FEH-1FFH处，固定值为AA55H。

- DPT硬盘分区表，共64B节，分为4项，每项16B对应一个分区。（一个磁盘的主分区和扩展分区共有4个）
- 分区规范规定：一个硬盘可以有多个主分区，但最多只能有一个扩展分区，一个扩展分区可以划分为多个逻辑分区，所以一个硬盘最多只能划为4个主分区或者3个主分区和一个扩展分区。扩展分区只作为数据盘使用。
- MS的OS最多只让我们划一个主分区和一个扩展分区，非DOS分区。一个扩展分区可以划分为多个逻辑分区，如D:、E:等。

# DOS文件卷的组成

引导或保留扇区	文件分配表1 (FAT1)	文件分配表2 (FAT2)	根目录区	文件数据区
---------	------------------	------------------	------	-------

- 引导或保留扇区：或存引导代码或保留不用。
- FAT：存有整个文件存储空间的使用情况。
- 根目录区：记录文件或子目录在根目录中的占用情况。
- 文件数据区：存放系统文件、子目录文件、各种各样的应用程序、用户文件。

## 5.5 文件记录的组块与分解

- 一个物理块可以存放若干个逻辑记录，一个逻辑记录可以存放在若干个物理块中。把一个块中存放的逻辑记录的个数叫做块因子。
- 必须使用主存缓冲区：信息交换是以块为单位进行的。用户将要写的记录先写入主存缓冲区，当缓冲区满时，再写磁盘。
- 用户使用记录时，先将包含该记录的物理块读入内存缓冲区，然后进行记录分解。

## 5.6 文件存储器存储空间的管理

实现文件存储空间的分配：

- 应记住空闲存储空间的情况。
- 应为分配存储空间而设置相应的数据结构，
- 应提供对存储空间进行分配和回收的功能。

常用的对磁盘存储空间的管理方法：

- ① 空白文件目录（是一种最简单的方法）
- ② 空闲块链表
- ③ 位映像表(bit map)或位示图

# 空白文件目录

空白文件：一个连续未用的空闲盘块区。

空白文件目录：系统为所有这些空白文件建立一张表。每个空白文件占用一个表目。

适合于文件的静态分配（连续文件的分配）。

## 空白文件目录

分配/回收/合并

第一物理块号	空白块个数	物理块号
15	4	(15,16,17,18)
23	5	(23,24,25,26,27)
:	:	:

- 当请求分配存储空间时，系统依次扫描空闲文件目录表目，直到找着一个合适的空闲文件为止。
- 当用户撤消一个文件时，系统回收该文件所占用的空间。这时也需要顺序扫描空闲文件目录，寻找一个空表目，并将释放空间的第一个物理块号及它所占的块数填入到这个表目中。

- 仅当文件存储空间中只有少量空闲区时，这种方法才有较好的效果。
- 如果存储空间中有大量小的空闲区，则其目录变得很大，因而效率大为降低。这种分配技术适用于建立连续文件。

# 空闲块链表

## (1) 空闲块链

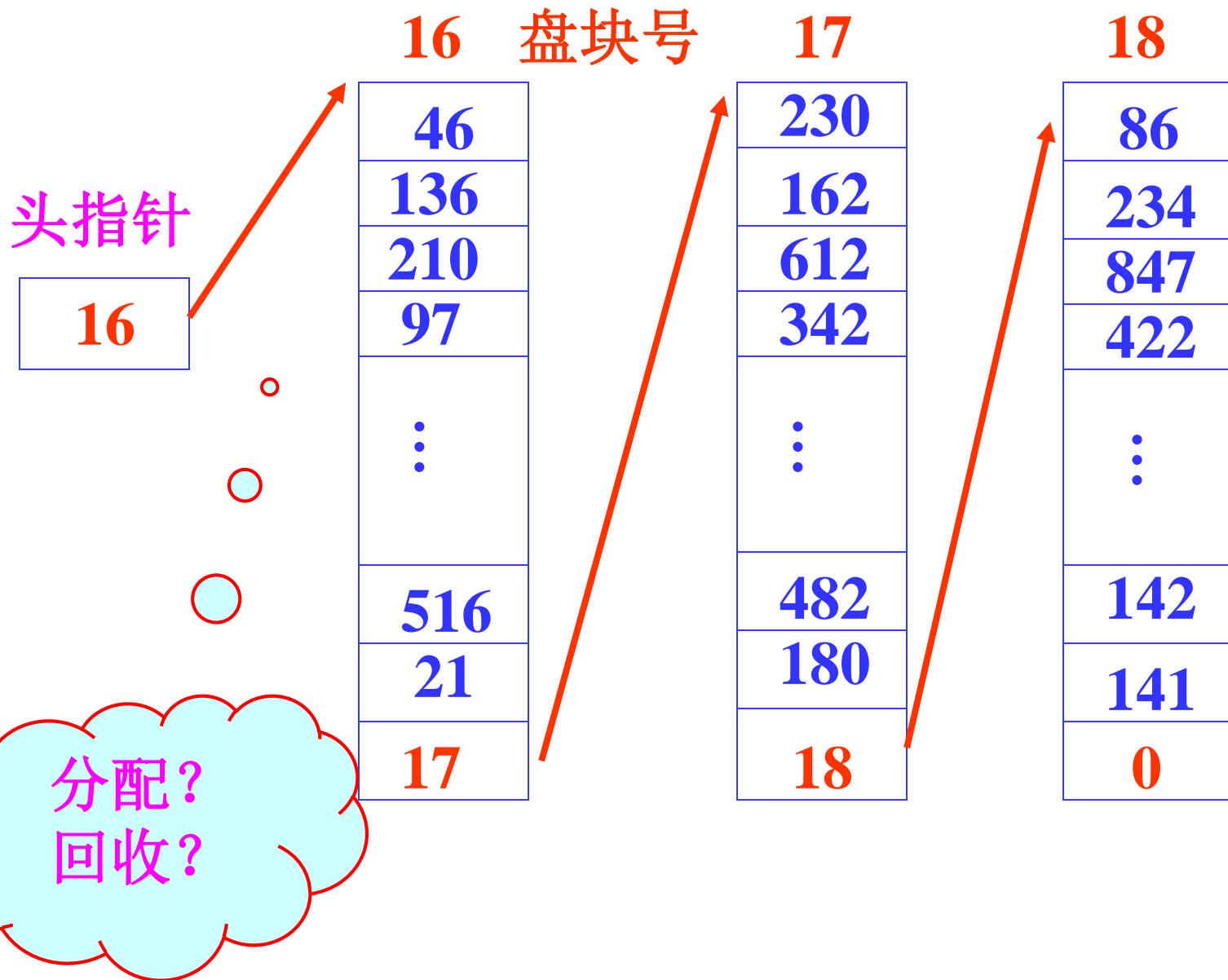
把所有空闲块连接成一个链表。在主存保留一个链头指针。

- ❖ **优点：**简单。适合文件动态分配。
- ❖ **缺点：**工作效率低。分配和回收多个盘块时要多次访问磁盘才能完成。

- 当用户建立文件时，就按需要从链首依次取下几块分配给文件。当撤消文件时，回收其存储空间，并将回收的空闲块依次链入空闲块链中。
- 这种方法的优点是实现简单；但工作效率低，因为每当在链上增加或移去空闲块时需要对空闲块链做较大的调整，因而会有较大系统开销。

## (2) 空闲块成组链表

- 利用盘空闲块来管理盘上的空闲块，每个磁盘块记录尽可能多的空闲块而成一组。各组之间也用链指针链接在一起。
- 便于空闲块的分配与回收。
- 适合连续文件、链接文件和索引文件的存储分配。



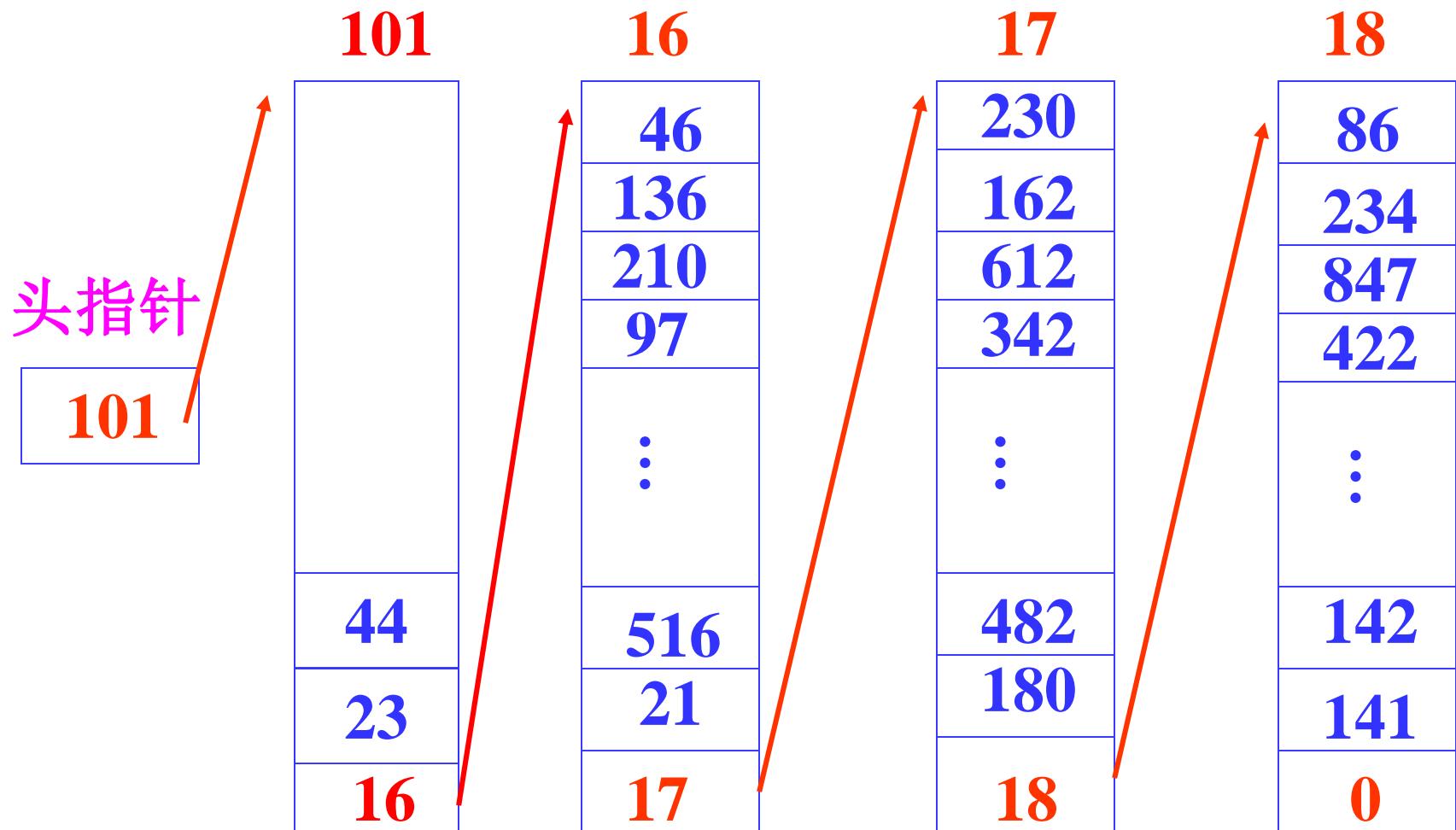
当用户要求分配文件存贮空间时，可将链头指针所指的组块读入主存，从中找到一个空闲块分配给它。当未满足分配要求时，再从链中将记录下一组空闲块的盘块读入内存，再进行分配。

当回收时，则将释放的盘块记录在主存的组盘块中即可。若该块已满，除将该块内容写回磁盘外，可用刚释放块记录其它要释放块，并将其放到链头即可。

显然这种管理方法既适合连续结构，也适合链接和索引结构。

连续回收了3个盘块: 101, 23, 44

盘块号



# 位映像表（位示图）

是适合文件静态分配和动态分配的最简单方法

位映像表（位向量）：每一个二进制位对应一个物理盘块。为1时表示块已分配，为0时空闲。

[例] 一个10G的磁盘，每个盘块为4k字节时，它要求一个2.5M位的映像表，这个表需占用  $2.5M/8/4k = 78$  个盘块。

➤ 利用位示图来进行空闲块分配时，只需查找图中的“0”位，并将其置为“1”位。反过来，利用位示图回收时，只需把相应的比特位由“1”改为“0”即可。

字	位	0	1	2	3	4	5	6	…	15
0		1	1	0	0	0	...			
1		0	0	0	1	1	...			
		0	0	0	0	1	...			
		...		...						

优点：易找到一个或几个连续的空闲块。  
其尺寸固定，通常又比较小，可以保存在  
主存，便于分配和回收空间。

## 采用位映像表时系统要做的工作

1. 字节号、位号转换成磁盘的相对块号

    磁盘的相对块号=字节号×8+位号

2. 将磁盘的相对块号转换成字节号、位号

    字节号=（相对块号/8）的商

    位号=（相对块号/8）的余数

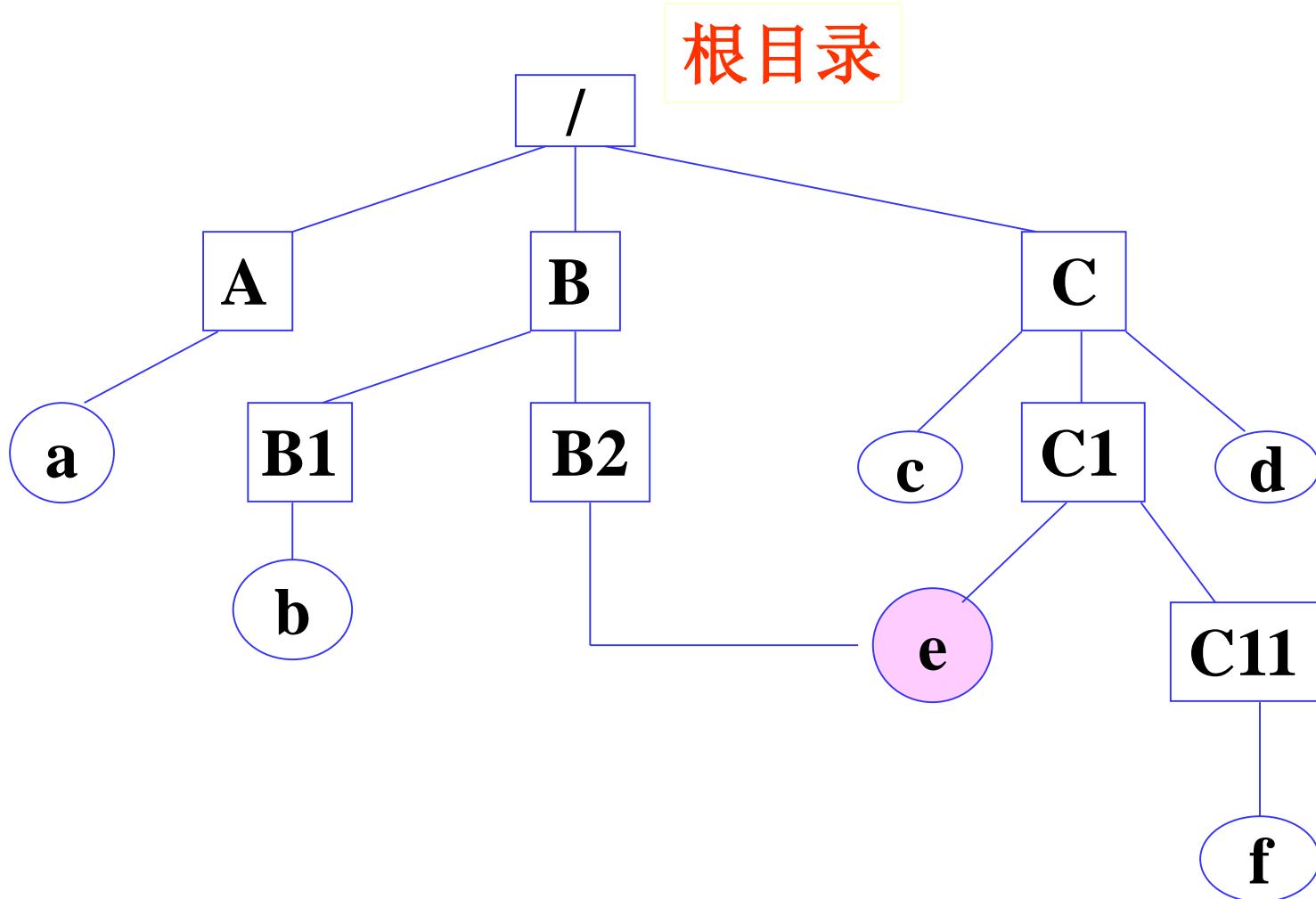
- ❖ UNIX系统采用**空闲块成组链表**
- ❖ Linux的Ext2文件系统把磁盘块分为组，即**块组**，每组用“**位图**”来管理组内的磁盘块。
- ❖ Windows的NTFS用“**位图**”记录卷上各簇的使用情况。

## 5.7 文件的共享与保护

- **文件共享：**允许多个用户共同使用同一文件。
- **文件保护：**防止数据丢失和被无权使用的用户窃取。
- **文件的存取控制：**是指一个用户对其文件所做的“谁能使用和如何使用”的规定，以便对文件的共享加以限制。

# 文件的共享

- 1) 使一个共享文件的信息同时出现在属于不同用户的不同目录中。共享文件的目录项完全相同。——硬链接
- 2) 在共享文件的目录项中，必须增加一个“用户计数”字段，以说明有几个用户共享使用。



文件e的路径名： /C/C1/e ; /B/B2/e

# 文件的保护

## 1. 文件复制

- 可以对系统保存的所有文件进行双份或多份**复制**，一旦一份被破坏，就可以使用另一份。
- 文件复制方法：周期性的全量转存、增量转存。

# 转贮方法

- (1) 从最近一次全量转贮盘或带中装入全部系统文件，使系统得以重新启动，并在其控制下完成以下操作；
- (2) 由近到远从增量转存盘上恢复文件。
- (3) 对于由增量转贮中仍未恢复的文件，再由全量转贮中进行恢复。

## 2. 增加防护设施

- ❖ 防止病毒程序破坏文件。
- ❖ 用户的鉴别----口令。在多用户系统中，保存了一个已被加密的通行字（口令）文件，每个用户在文件中占有一行。系统登录程序根据键入的用户名和口令在文件中逐行比较，若匹配，则允许进入。
- ❖ 物理鉴定。使用磁卡、指纹和签名等技术。
- ❖ 对文件进行加密。加密程序根据用户提供的代码键对文件进行转换。

为了防止文件泄密，对一些关键文件进行加密。用户加密一个文件时，须提供一个代码键，加密程序根据这一个代码键对用户文件进行变换，从而得到其相应的密码文件。在读取和执行文件时，加密程序再用相同的代码键对文件进行解密，还原成正常文件后，再使用。

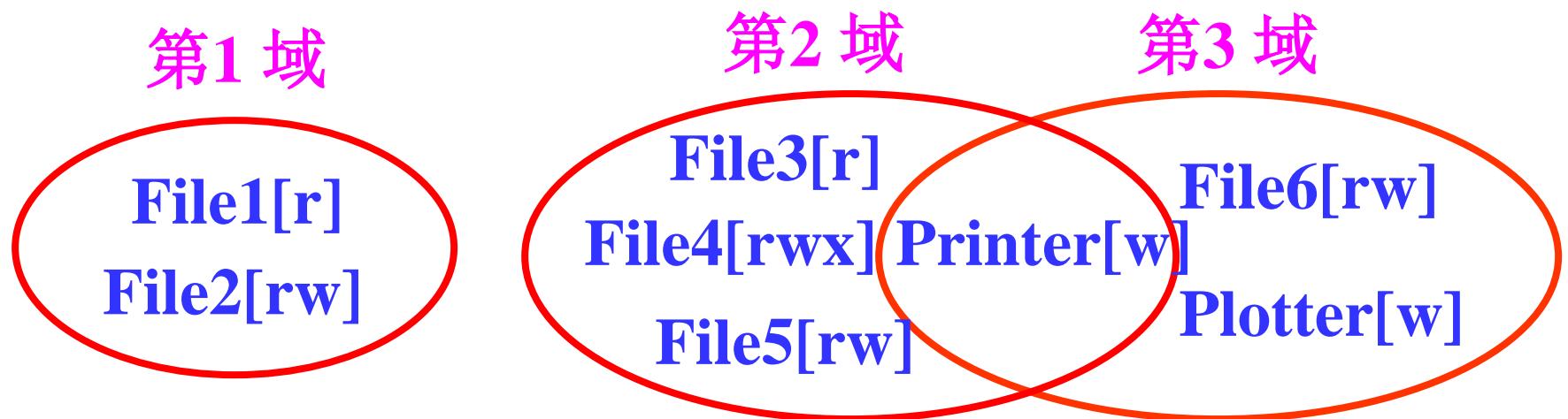
# 文件的存取控制

- 文件存取控制信息规定了各类用户对文件的存取权限。防止核准的用户误用文件和未核准的用户存取文件。
  - 文件的存取控制信息包含在文件目录项中。
- 常用的文件存取控制方法：保护域、存取控制表。

# 保护域(Protection Domains)

- 每个对象都有惟一的名字和允许施加的操作。  
硬件对象(磁盘驱动器、打印机)、软件对象(文件、程序)。
- 一个域是一组 (对象、存取权限) 偶对。即给出它能操作的对象和对每个对象的存取权限。  
防止无权的用户存取对象。
- 每个用户是一个域，每个进程是一个域。
- 核心态的存取权限不同于用户态的，一个系统调用可以引起域的变换。

- 同一个对象可能出现在几个不同的域中，在每个域中可能有不同的存取权限。



具有三个保护域的系统

- ❖ 可用存取控制矩阵来记录各个域对对象的存取权限。

域 \ 对象	F1	F2	F3	F4	F5	print	F6	plot	D1	D2	D3
D1	r	rw									
D2			r	rwx	rw	w					control
D3						w	rw	w			



表示在域D2中运行的进程可以改变在域D3中运行的进程对各对象的任何访问权

同一个目标可能出现在几个不同的域中，在每个域中可能具有不同的存取权限。在任何一个时刻，每个进程运行在其中的某个保护域中。

下面以UNIX系统为例，看一看域的具体实现。

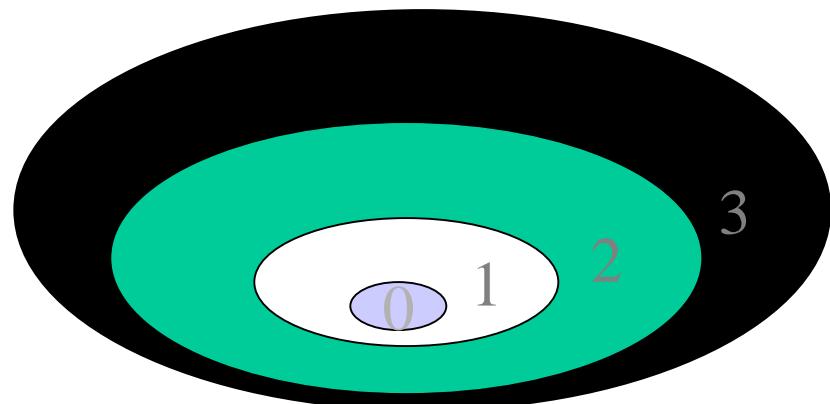
在UNIX系统中，每个进程的域是由它的uid(用户标识)和gid(组标识)定义的。给定一个(uid, gid)组合，就可以列出这组用户能存取(读、写、执行)的所有目标。

具有相同的(uid, gid)的两个进程对任何一组目标都具有相同的存取权限。

具有不同( $uid$ ,  $gid$ )值的进程对同一组目标具有不同的存取权限。

在UNIX系统中，每个进程可能工作在两种不同状态，即核心态，用户态。当一个进程进行一个系统调用时，它将从用户态转换到核心态。核心态的存取目标权限与用户态完全不同。

核心态的进程可以存取物理存储器的所有页、整个磁盘和系统所有的其它被保护资源。因此，一个系统调用可以引起域的转接。



MULTICS系统中的进程四个环状保护域

UNIX系统中的进程划分为核心态和用户态。

在MULTICS系统中使用了功能更强的域转接机制。硬件对每个进程不仅有核心，用户两个域，它可以支持64个不同的域。

MULTICS中一个进程由一组过程组成，每一个过程运行在某一个域中，这个域又称作环(ring)。

最内层环是操作系统内核，它具有最大的权限，从核心向外的各环权力越来越小。

当一个环中的过程运行中要调用另一个环中的过程时，产生一个失陷(trap)，此时系统改变进程的保护域。因此，MULTICS进程在它的生命期内可以操作在多至64个不同的域中。各个域对文件的存取权限记录在它的保护矩阵中。

# 存取控制矩阵

- 它是一个二维矩阵，一维列出使用该文件的所有用户；另一维列出存入系统中的全部文件，矩阵中的每一个元素是用户对文件的存取权限。
- 当一个用户向文件系统提出存取请求时，由存取控制验证模块根据这个存取控制矩阵，将本次请求和该用户对这个文件的存取权限进行比较，如果不匹配就拒绝执行。

这个方法虽然在概念上比较简单，但当文件和用户较多时，存取控制矩阵将变得非常庞大，将占据很大的存储空间，查找如此大的表需要花费很多时间，因此往往采用某些辅助措施以减少时间和空间上的开销。

# 存取控制表(Access Control List)

- 存取控制矩阵很少用，因为矩阵中的空项很多，浪费存储空间。
- 为存取控制矩阵中的每一列建立一张存取控制表(ACL)，用一有序对(域, 权集)表示。

- 通常，一个文件只与特定的几个用户有关，而与其他用户无关。因此，可以按用户对文件的存取权限将用户分成若干组，同时规定每一组用户对文件的访问权限。这样，所有用户组存取权限的集合组成了该文件的存取控制表。
- 这种方法实际上是对存取控制矩阵的一种改进，它不像存取控制矩阵那样，对整个系统中所有文件的访问权限进行集中控制，而是对系统中的每个文件设立一个存取控制表。

域\对象	F1	F2	F3	F4	F5	print	F6	plot	D1	D2	D3
D1	r	rw									
D2			r	rwx	rw	w					control
D3						w	rw	w			

一些对象的存取控制表:

**File1: (D1, r--)**

**File2: (D1, rw-)**

**File4: (D2, rwx)**

**printer: (D2, -w-), (D3, -w-)**

**UNIX系统:** 每个文件: 拥有者权限、小组成员权限、其它成员权限。如: **111101001**。

## 5.8 文件的操作命令

### 1) 创建(Create)文件

主要功能：在指定设备上为指定路径名的文件建立一个目录项，并设置文件的有关属性。

### 2) 删除(Delete)文件

主要功能：根据文件的路径名找到指定的目录项，回收其占用的各个物理块，再将该目录项置为空。

### 3) 打开(Open)文件

- 根据文件路径名找到目录项，将文件的目录项复制到主存一个专门区域，返回文件在该区域的索引。建立进程与文件的联系。
- 目的：避免多次重复地检索文件目录。
- 系统维护了一个系统当前打开文件表。当读/写文件时，通过这个表的索引找到文件的主存目录项。不需要重复地对磁盘进行检索。

#### 4) 关闭(Close)文件

- 释放文件在主存专门区域中的目录项，切断用户与文件的联系。
- 若该目录项被修改过，则复制到磁盘。
- 若文件作过某些修改，应将其写回辅存。

#### 5) 读(Read)文件

- 命令中必须指出要读的数据个数，以及存放数据的主存地址。
- 根据文件所在设备、文件类型的不同，系统设置不同的读命令。

## 6) 写(Write)文件

命令中必须指出要写的数据个数，以及存放数据的主存地址，将主存中的数据写到指定的文件中。

## 7) 追加(Append)文件

限制了写文件的形式，将数据追加到文件尾。

## 8) 随机存取(Seek)文件

重新定位文件的读/写位置指针。

## 9) 得到文件属性(Get Attributes)

进程在执行时常常需要了解文件的属性。在UNIX系统中，一个软件开发项目通常由多个源文件组成，make程序用来管理这些软件开发项目。当make被调用时，它检查所有源文件和目标文件的修改时间，并且编排出需要重新编译的文件数。

## 10) 设置文件属性(Set Attributes)

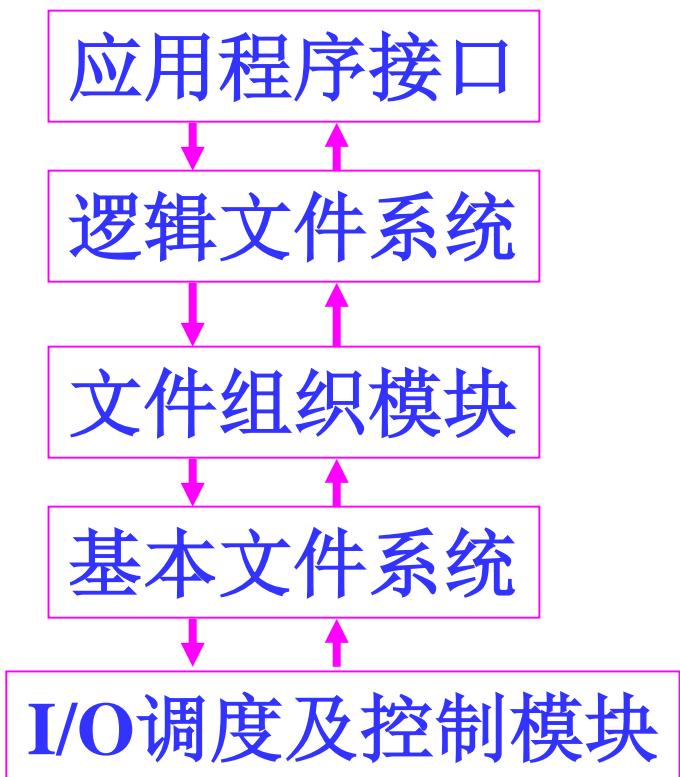
修改文件的一些属性，以适应用户的要求。

## 11) 重命名(Rename)文件

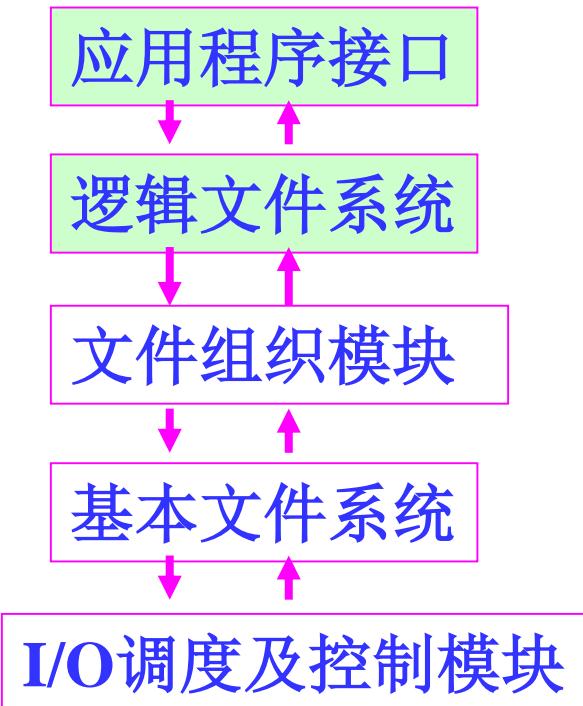
重新命名一个已经存在的文件。

## 5.9 文件系统的组织结构

- ❖ 文件系统是用户和外存设备之间的接口。
- ❖ 引入文件系统后，用户可用统一的文件观点对待和处理各种存储介质上的信息。
- ❖ 文件系统向用户提供许多操作文件和目录的命令

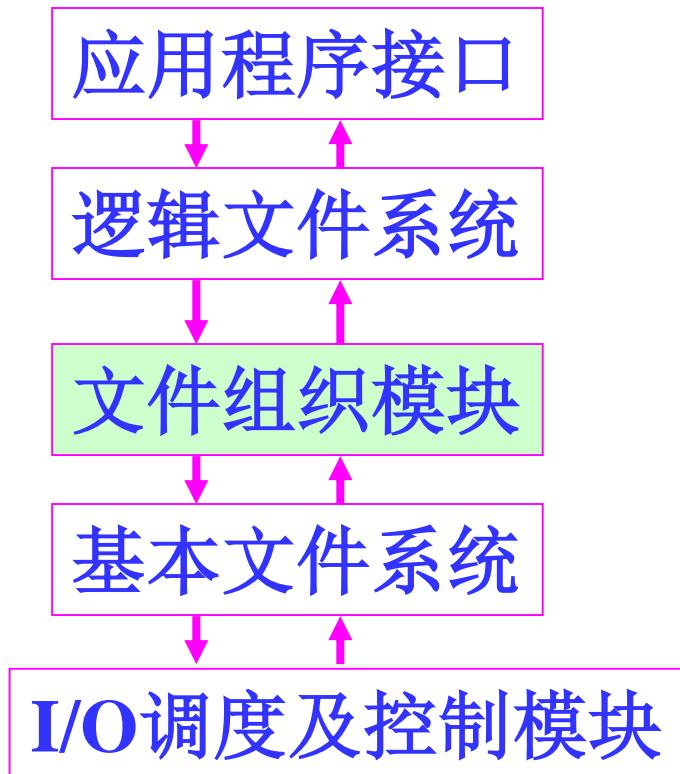


1. **应用程序接口**: 检查由用户提供的命令句法的正确性和参数的合法性。
2. **逻辑文件系统**: 负责目录的管理和维护。按照命令给定的文件名查找目录。
  - **创建文件**: 增加目录项。
  - **打开文件**: 检查各级目录，找到相应的目录项。
  - **读/写文件**: 检查文件是否已经以读/写方式打开。

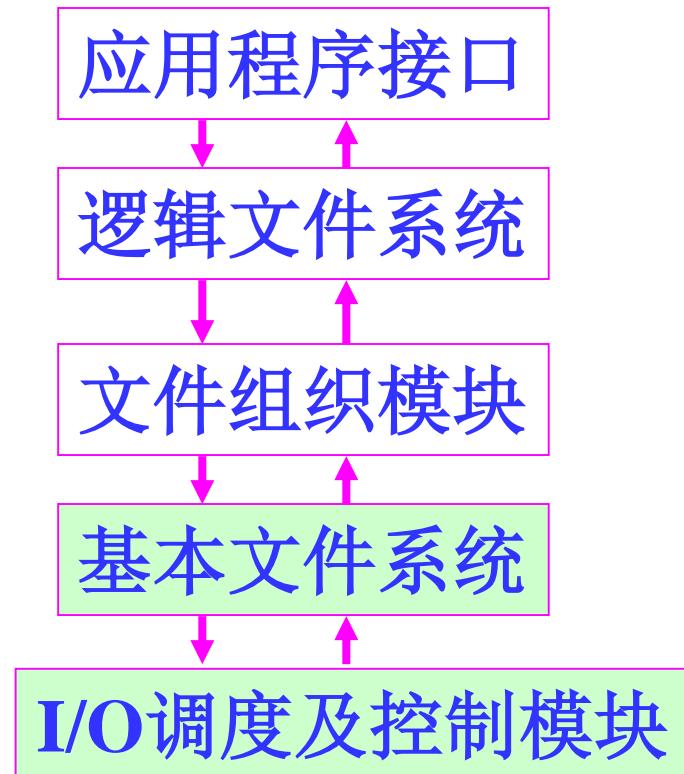


### 3. 文件组织模块:

- 负责将文件的读/写位置转换成文件中的相对块号，进而转换成在存储器中的物理块号。
- 管理磁盘空闲空间。
- 将上层传下来的命令转换成对基本文件系统的调用。



4. 基本文件系统：将上层传下来的命令和物理块转换成对设备驱动程序的调用。
5. I/O调度和控制模块：由设备驱动和中断处理模块组成。将上层传来的命令转换成硬件设备的专用I/O指令和设备地址，控制设备完成与主存之间的信息传输。还负责多个读/写命令的排队和调度。

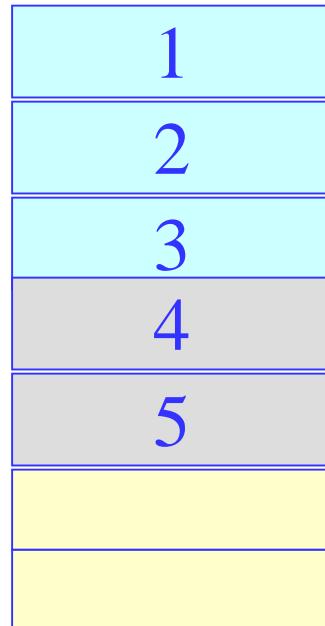


## 5.10 存储器映射文件

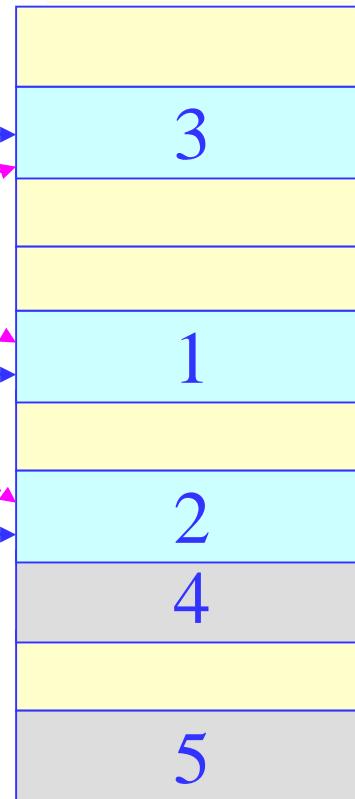
- 文件系统的读文件：先把文件信息块读入系统缓冲区，再送入用户进程指定的位置。
- 存储器映射文件：将文件映射到进程地址空间的一个区域，返回虚拟地址，仅当需要对文件存取时，才传输实际的数据。
- 使用与请求页式虚存管理相同的存取机制。访问的页不在主存时，产生缺页中断读入主存。
- OS提供映射文件的系统调用。

# 内存映射文件

## 进程A虚拟内存



## 物理内存



## 磁盘文件



应用于UNIX, Linux, Windows。

## 5.11 小结

文件系统是OS的信息管理部分，提供文件的按名存取。负责文件的存储、检索和存取保护。

文件是具有符号名的相关信息的集合。

文件的逻辑结构：无结构的字节流式文件，有结构的记录式文件。

文件的存取方法：顺序存取、直接存取。

文件物理结构：连续、链接、索引、索引顺序。

文件目录是文件系统提供按名存取文件的重要数据结构。单级、二级和多级树状目录。二级和多级目录较好地解决文件的重名和共享。

**文件存储空间的管理**: 空白文件目录、位映像表(或位示图)及空白块链。

**对文件的存取控制**: 存取保护域、存取控制矩阵、存取控制表(ACL)、口令和密码等。

**文件的操作命令**: 文件系统向用户提供的接口。

**文件系统的层次结构**: 应用程序接口、逻辑文件系统、文件组织模块、基本文件系统、I/O调度和控制模块。