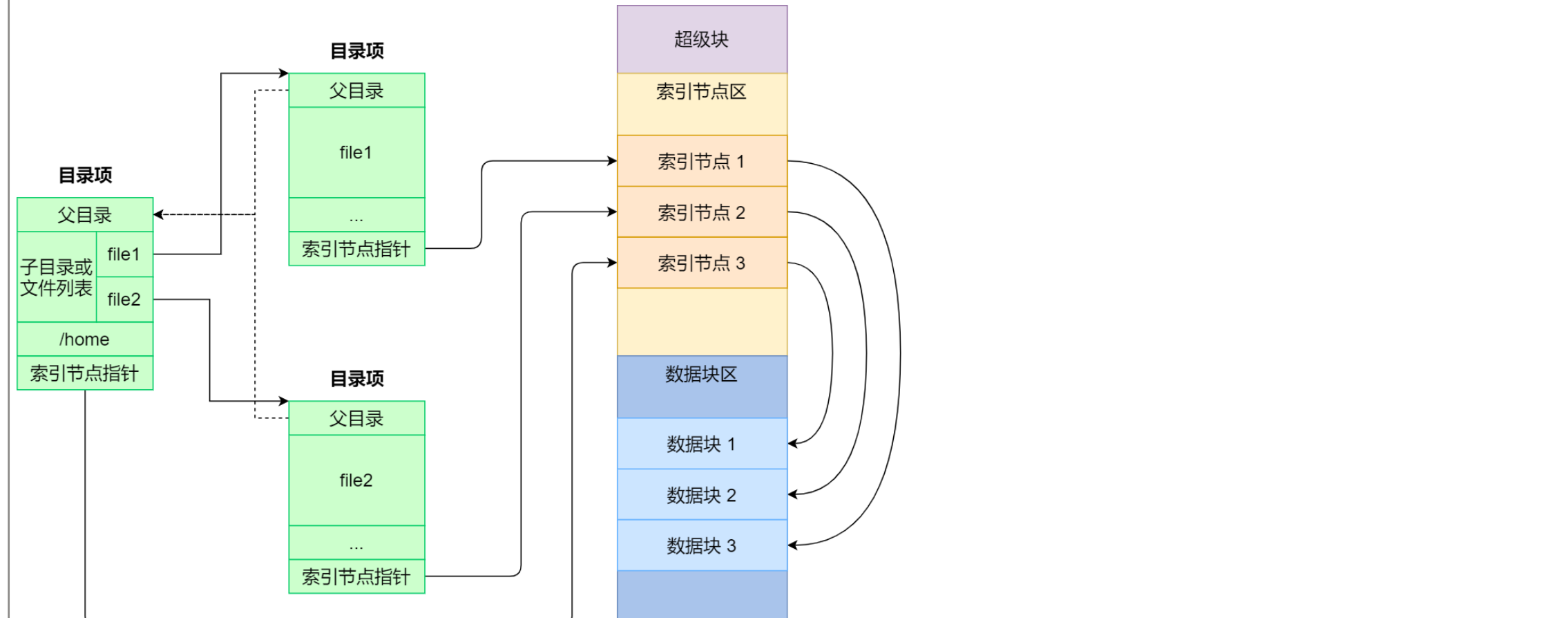
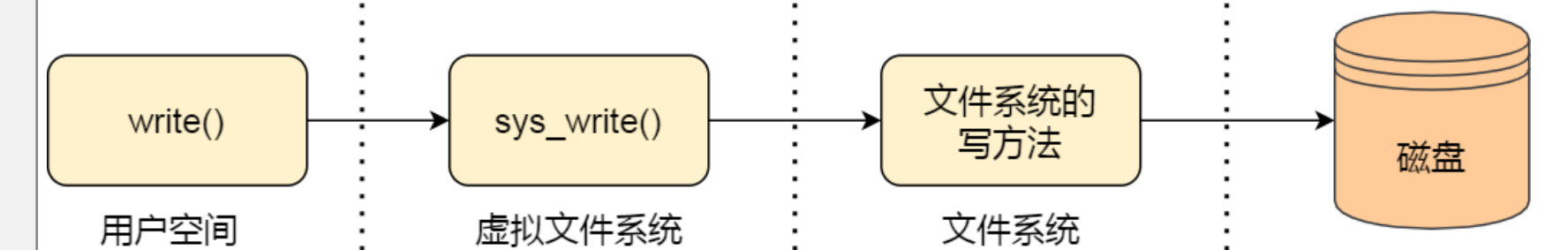
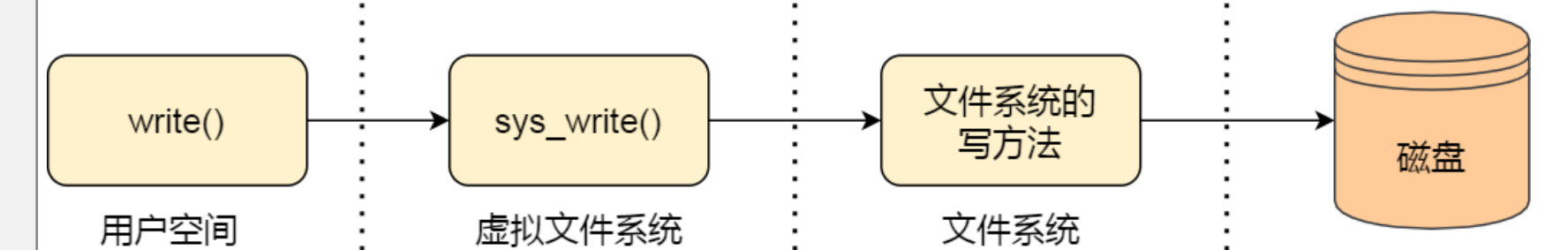
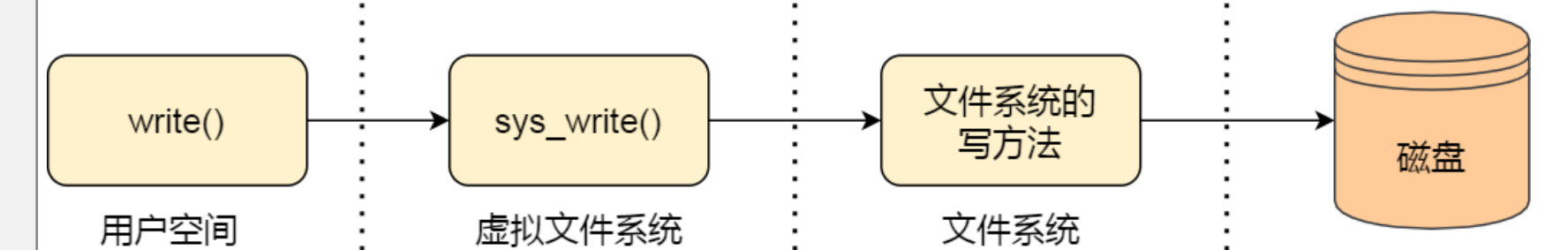


文件

基本概念	
文件是什么？	<ul style="list-style-type: none">文件是以硬盘为载体的存储在计算机上的信息集合文件可以是文本文档，图片， 程序用户进行的输入输出中，以文件为基本单位
文件由什么组成？	<ol style="list-style-type: none">一块存储空间分类和索引的信息关于访问权限的信息
文件有什么特性？	<ol style="list-style-type: none">可以长期存储在硬盘中允许可控制的进程间共享访问能够被组织成复杂的结构
文件的属性/元数据	<ul style="list-style-type: none">文件名：文件类型创建者：所有者位置：大小；保护；创建信息

文件的数据结构					
目录项	<table><tr><td>相关概念</td><td><ul style="list-style-type: none">目录项/FCB = 用来记录文件的名字，索引节点指针以及其他目录项的层级关联关系目录 = 目录项的集合目录也被视作一个文件，该文件叫做目录文件目录项是义内核维护的一个数据结构， 缓存存在内存目录项文件存储在磁盘</td></tr><tr><td>包含信息</td><td><ul style="list-style-type: none">基本信息：文件名；文件物理位置，文件逻辑结构，文件物理结构存取控制信息：文件主或核准用户或一般用户的存取权限使用信息：文件创立时间，上次修改时间</td></tr></table>	相关概念	<ul style="list-style-type: none">目录项/FCB = 用来记录文件的名字，索引节点指针以及其他目录项的层级关联关系目录 = 目录项的集合目录也被视作一个文件，该文件叫做目录文件目录项是义内核维护的一个数据结构， 缓存存在内存目录项文件存储在磁盘	包含信息	<ul style="list-style-type: none">基本信息：文件名；文件物理位置，文件逻辑结构，文件物理结构存取控制信息：文件主或核准用户或一般用户的存取权限使用信息：文件创立时间，上次修改时间
相关概念	<ul style="list-style-type: none">目录项/FCB = 用来记录文件的名字，索引节点指针以及其他目录项的层级关联关系目录 = 目录项的集合目录也被视作一个文件，该文件叫做目录文件目录项是义内核维护的一个数据结构， 缓存存在内存目录项文件存储在磁盘				
包含信息	<ul style="list-style-type: none">基本信息：文件名；文件物理位置，文件逻辑结构，文件物理结构存取控制信息：文件主或核准用户或一般用户的存取权限使用信息：文件创立时间，上次修改时间				
索引节点	<table><tr><td>概念</td><td><ul style="list-style-type: none">索引节点是用来记录文件的元信息，是文件的唯一表示索引节点同样占用磁盘空间</td></tr><tr><td>分类</td><td>磁盘索引节点<ul style="list-style-type: none">指存放在磁盘上的索引节点，每个文件有一个唯一的磁盘索引节点包含内容：文件主标识符；文件类型；文件存取权限；文件物理地址；文件长度；文件链接计数；文件存取时间内存索引节点<ul style="list-style-type: none">指存放在内存中的索引节点，文件打开后，将磁盘索引节点复制到内存中新增内容：索引节点编号；状态；访问计数；逻辑设备号；链接指针</td></tr></table>	概念	<ul style="list-style-type: none">索引节点是用来记录文件的元信息，是文件的唯一表示索引节点同样占用磁盘空间	分类	磁盘索引节点 <ul style="list-style-type: none">指存放在磁盘上的索引节点，每个文件有一个唯一的磁盘索引节点包含内容：文件主标识符；文件类型；文件存取权限；文件物理地址；文件长度；文件链接计数；文件存取时间 内存索引节点 <ul style="list-style-type: none">指存放在内存中的索引节点，文件打开后，将磁盘索引节点复制到内存中新增内容：索引节点编号；状态；访问计数；逻辑设备号；链接指针
概念	<ul style="list-style-type: none">索引节点是用来记录文件的元信息，是文件的唯一表示索引节点同样占用磁盘空间				
分类	磁盘索引节点 <ul style="list-style-type: none">指存放在磁盘上的索引节点，每个文件有一个唯一的磁盘索引节点包含内容：文件主标识符；文件类型；文件存取权限；文件物理地址；文件长度；文件链接计数；文件存取时间 内存索引节点 <ul style="list-style-type: none">指存放在内存中的索引节点，文件打开后，将磁盘索引节点复制到内存中新增内容：索引节点编号；状态；访问计数；逻辑设备号；链接指针				
两者的关系图					

文件的操作							
基本操作	<ul style="list-style-type: none">创建文件；写文件；读文件；重新定位文件；删除文件；截断文件						
文件打开和关闭关联的信息	<ul style="list-style-type: none">文件指针；文件打开计数；文件磁盘位置；访问权限 <p>文件描述符是打开文件的标识</p>						
读取文件的过程【举例】	<table><tr><td>例图</td><td></td></tr><tr><td>代码</td><td><pre>fd=open(name,flag)# 打开文件 write(fd,...)# 写数据 close(fd)# 关闭文件</pre></td></tr><tr><td>解释</td><td><ul style="list-style-type: none">首先用 open 系统调用打开文件，open 的参数中包含文件的路径名和文件名使用 write 写数据，其中 write 使用 open 所返回的文件描述符，并不使用文件名作为参数使用完文件后，要用 close 系统调用关闭文件，避免资源的泄露</td></tr></table>	例图		代码	<pre>fd=open(name,flag)# 打开文件 write(fd,...)# 写数据 close(fd)# 关闭文件</pre>	解释	<ul style="list-style-type: none">首先用 open 系统调用打开文件，open 的参数中包含文件的路径名和文件名使用 write 写数据，其中 write 使用 open 所返回的文件描述符，并不使用文件名作为参数使用完文件后，要用 close 系统调用关闭文件，避免资源的泄露
例图							
代码	<pre>fd=open(name,flag)# 打开文件 write(fd,...)# 写数据 close(fd)# 关闭文件</pre>						
解释	<ul style="list-style-type: none">首先用 open 系统调用打开文件，open 的参数中包含文件的路径名和文件名使用 write 写数据，其中 write 使用 open 所返回的文件描述符，并不使用文件名作为参数使用完文件后，要用 close 系统调用关闭文件，避免资源的泄露						

文件的保护

- 保护的**目的**
 - 解决对文件的读，写，执行的许可问题
- 保护的**方式**

非访问控制方法		1.口令	2.加密保护
	定义	<ul style="list-style-type: none">用户建立一个文件时需要提供口令用户请求访问时必须提供相应口令	<ul style="list-style-type: none">对文件进行加密，访问时需要密钥
	优点	<ul style="list-style-type: none">时间空间开销不多	<ul style="list-style-type: none">保密性强，节省了存储空间
访问控制方法	缺点	<ul style="list-style-type: none">口令直接存在系统内部，不安全	<ul style="list-style-type: none">编码和译码需要时间
	访问控制的目的：用于控制用户对文件的访问方式 访问控制的对象：读；写；执行；添加；删除；列表清单		
		方法一	方法二
	定义	<ul style="list-style-type: none">为每个文件和目录增加一个访问控制列表ACL该表规定每个用户名及其所允许的空间管理	<ul style="list-style-type: none">采用精简的访问列表该列表采用所有者、组和其他三种用户类型
	优点	<ul style="list-style-type: none">可以使用复杂的访问方法	<ul style="list-style-type: none">只需要三个域即可列出访问表中这三类用户的访问权限
	缺点	<ul style="list-style-type: none">长度无法预计并且可能导致复杂的空间管理	

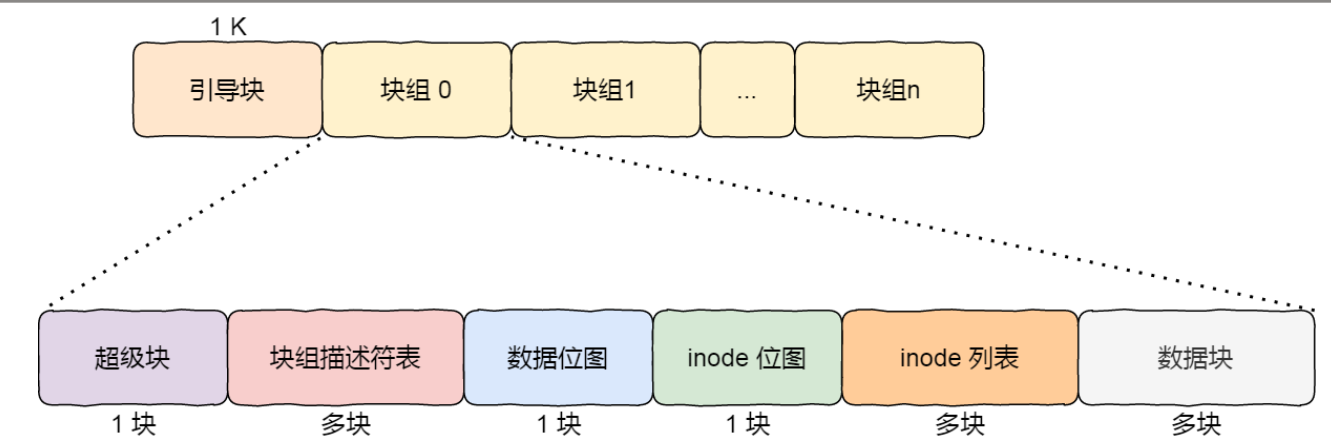
文件的逻辑结构【用户角度的文件组织形式】							
定义	<ul style="list-style-type: none">即文件中的数据在逻辑层面是如何组织起来的						
无结构文件/流式文件	<ul style="list-style-type: none">最简单的文件组织形式，是有序相关信息项的集合，以字节为单位对基本信息单元操作不多的文件适合该方式，如源代码文件，目标码文件						
有结构文件/记录式文件	<table><tr><td>顺序文件</td><td><ul style="list-style-type: none">串结构：只能按顺序查找，费时顺序结构：可采用折半查找，检索效率高</td></tr><tr><td>索引文件</td><td><ul style="list-style-type: none">提高了存取速度，但索引表增加了存储空间</td></tr><tr><td>索引顺序文件</td><td><ul style="list-style-type: none">提高了存取速度，但索引表增加了存储空间</td></tr></table>	顺序文件	<ul style="list-style-type: none">串结构：只能按顺序查找，费时顺序结构：可采用折半查找，检索效率高	索引文件	<ul style="list-style-type: none">提高了存取速度，但索引表增加了存储空间	索引顺序文件	<ul style="list-style-type: none">提高了存取速度，但索引表增加了存储空间
顺序文件	<ul style="list-style-type: none">串结构：只能按顺序查找，费时顺序结构：可采用折半查找，检索效率高						
索引文件	<ul style="list-style-type: none">提高了存取速度，但索引表增加了存储空间						
索引顺序文件	<ul style="list-style-type: none">提高了存取速度，但索引表增加了存储空间						

文件的物理结构【文件在外存上的存储组织形式】																			
定义	<ul style="list-style-type: none">研究文件数据在物理存储设备上是如何分布和组织的文件在磁带上---->连续存放方式文件在磁盘上---->不采用连续存放方式文件在内存上---->随机存放方式																		
文件的存储方式	<ul style="list-style-type: none">文件的存储就是对磁盘非空闲块的管理																		
	<table><tr><th>方式</th><th>访问磁盘次数</th><th>优点</th><th>缺点</th></tr><tr><td>顺序分配</td><td>需访问磁盘 1 次</td><td>顺序存取速度快，当文件是定长时可以根据文件起始地址及记录长度进行随机访问</td><td>要求连续的存储空间，会产生外部碎片，不利于文件的动态扩充</td></tr><tr><td>链表分配</td><td>需访问磁盘 n 次</td><td>无外部碎片，提高了外存空间的利用率，动态增长较方便</td><td>只能按照文件的指针链顺序访问，查找效率低，指针信息存放消耗内存或磁盘空间</td></tr><tr><td>索引分配</td><td>m 级需访问磁盘 m+1 次</td><td>可以随机访问，易于文件的增删</td><td>索引表增加存储空间开销，索引表的查找策略对文件系统效率影响较大</td></tr></table>	方式	访问磁盘次数	优点	缺点	顺序分配	需访问磁盘 1 次	顺序存取速度快，当文件是定长时可以根据文件起始地址及记录长度进行随机访问	要求连续的存储空间，会产生外部碎片，不利于文件的动态扩充	链表分配	需访问磁盘 n 次	无外部碎片，提高了外存空间的利用率，动态增长较方便	只能按照文件的指针链顺序访问，查找效率低，指针信息存放消耗内存或磁盘空间	索引分配	m 级需访问磁盘 m+1 次	可以随机访问，易于文件的增删	索引表增加存储空间开销，索引表的查找策略对文件系统效率影响较大		
方式	访问磁盘次数	优点	缺点																
顺序分配	需访问磁盘 1 次	顺序存取速度快，当文件是定长时可以根据文件起始地址及记录长度进行随机访问	要求连续的存储空间，会产生外部碎片，不利于文件的动态扩充																
链表分配	需访问磁盘 n 次	无外部碎片，提高了外存空间的利用率，动态增长较方便	只能按照文件的指针链顺序访问，查找效率低，指针信息存放消耗内存或磁盘空间																
索引分配	m 级需访问磁盘 m+1 次	可以随机访问，易于文件的增删	索引表增加存储空间开销，索引表的查找策略对文件系统效率影响较大																
文件的存储空间管理	<ul style="list-style-type: none">文件的存储空间管理就是对磁盘空闲块的管理																		

文件系统

基本概念和目标	
概念	<ul style="list-style-type: none">文件系统 = OS中负责管理持久数据的子系统文件系统 = 与文件管理有关的软件 + 被管理的文件 + 试试文件管理所需的数据结构文件系统需先挂到某个目录才可正常使用文件的基本操作单位就是数据块
目标	<ol style="list-style-type: none">实现对文件的基本操作 = 按名存储和查找文件 + 组织成合适的结构 + 文件共享 + 文件保护【用户角度】管理与磁盘的信息交换 + 完成逻辑结构和物理结构的变换【OS角度】组织文件在磁盘上的存放 + 采取好的文件排放顺序和磁盘调度方法【OS角度】
分类	磁盘的文件系统 ：它是直接把数据存储在磁盘中，比如 Ext 2/3/4、XFS 等都是这类文件系统 内存的文件系统 ：这类文件系统的数据不是存储在硬盘的，而是占用内存空间 <ul style="list-style-type: none">我们经常用到的 /proc 和 /sys 文件系统都属于这一类读写这类文件，实际上是读写内核中相关的数据 网络的文件系统 ：用来访问其他计算机主机数据的文件系统，比如 NFS、SMB 等等

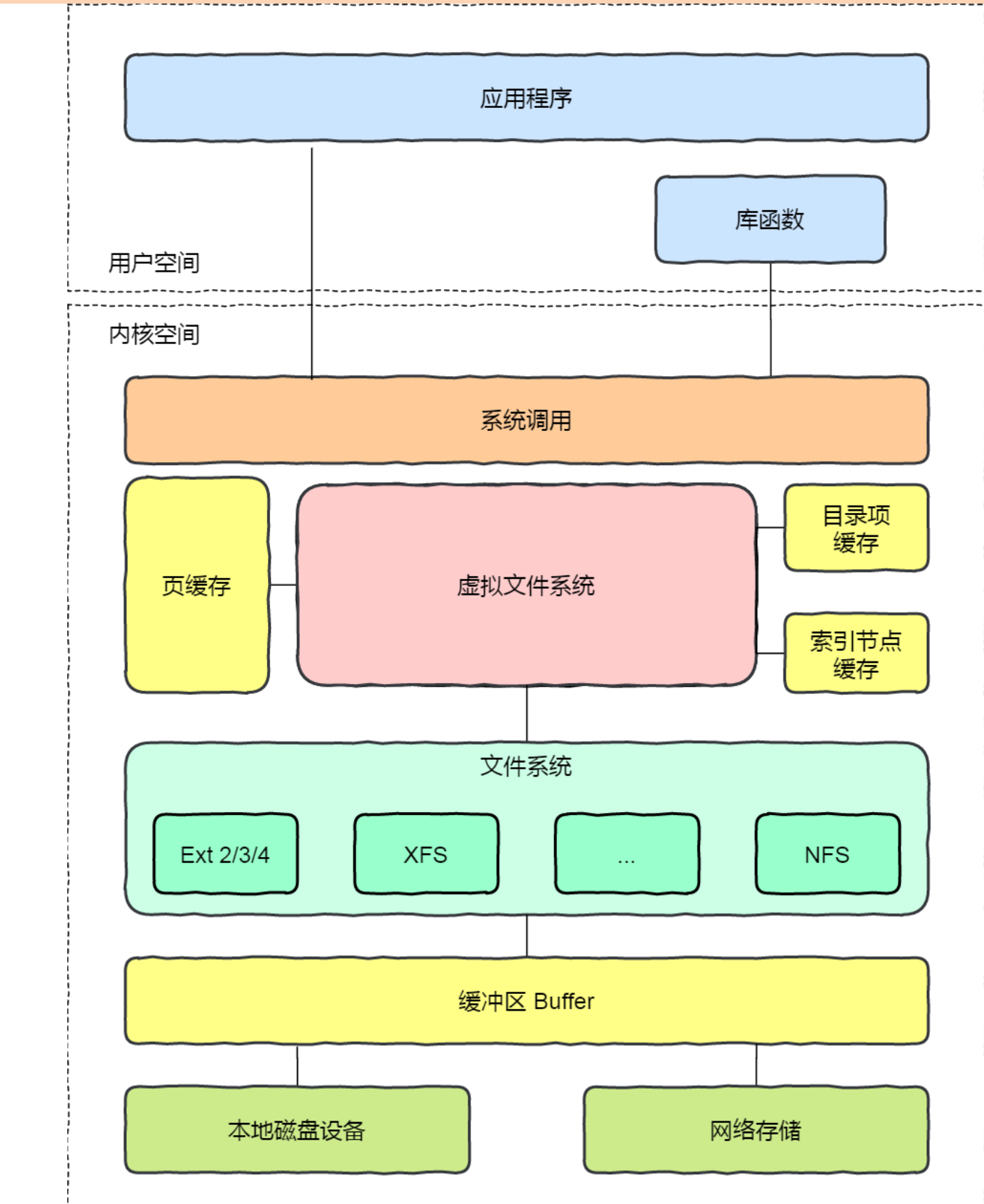
文件系统的层次结构	
	主要功能和介绍
I/O控制	设备驱动程序 ：将输入的命令翻译成底层硬件的特定指令 中断处理程序 ：利用指令使IO设备与系统交互
基本文件系统	<ul style="list-style-type: none">向对应的设备驱动程序发送通用命令，以读取和写入磁盘的物理块管理内存缓冲区，保存各种文件系统，目录和数据块的缓冲
文件组织模块	<ul style="list-style-type: none">组织文件及其逻辑块和物理块可以将逻辑地址转换为物理地址有空闲空间管理器，以跟踪未分配的块，根据需要提供给文件组织模块
逻辑文件系统	<ul style="list-style-type: none">用于管理元数据信息（包括文件系统的所有结构，不包括文件内容）管理目录结构通过FCB维护文件结构负责文件保护

文件系统的布局									
在磁盘中的结构	 <ul style="list-style-type: none">最前面的第一个块是引导块，在系统启动时用于启用引导接着后面就是一个一个连续的块组了，块组的内容如下 <table><tr><td>超级块</td><td><ul style="list-style-type: none">包含的是文件系统的重要信息比如 inode 总个数、块总个数、每个块组的 inode 个数、每个块组的块个数</td></tr><tr><td>块组描述符</td><td><ul style="list-style-type: none">包含文件系统中各个块组的状态比如块组中空闲块和 inode 的数目等，每个块组都包含了文件系统中「所有块组的组描述符信息」</td></tr><tr><td>数据位图 inode 位图 inode 列表</td><td><ul style="list-style-type: none">用于表示对应的数据块或 inode 是空闲的，还是被使用中包含了块组中所有的 inode，inode 用于保存文件系统中与各个文件和目录相关的所有元数据</td></tr><tr><td>数据块</td><td><ul style="list-style-type: none">包含文件的有用数据</td></tr></table>	超级块	<ul style="list-style-type: none">包含的是文件系统的重要信息比如 inode 总个数、块总个数、每个块组的 inode 个数、每个块组的块个数	块组描述符	<ul style="list-style-type: none">包含文件系统中各个块组的状态比如块组中空闲块和 inode 的数目等，每个块组都包含了文件系统中「所有块组的组描述符信息」	数据位图 inode 位图 inode 列表	<ul style="list-style-type: none">用于表示对应的数据块或 inode 是空闲的，还是被使用中包含了块组中所有的 inode，inode 用于保存文件系统中与各个文件和目录相关的所有元数据	数据块	<ul style="list-style-type: none">包含文件的有用数据
超级块	<ul style="list-style-type: none">包含的是文件系统的重要信息比如 inode 总个数、块总个数、每个块组的 inode 个数、每个块组的块个数								
块组描述符	<ul style="list-style-type: none">包含文件系统中各个块组的状态比如块组中空闲块和 inode 的数目等，每个块组都包含了文件系统中「所有块组的组描述符信息」								
数据位图 inode 位图 inode 列表	<ul style="list-style-type: none">用于表示对应的数据块或 inode 是空闲的，还是被使用中包含了块组中所有的 inode，inode 用于保存文件系统中与各个文件和目录相关的所有元数据								
数据块	<ul style="list-style-type: none">包含文件的有用数据								
在内存中的结构	<ul style="list-style-type: none">内存中的信息用于管理文件系统并通过缓存来提高信息这些结构有以下类型内存中的安装表内存中的目录结构的缓存包含最近访问目录的信息整个系统的打开文件表每个进程的打开文件表								

外存空闲空间管理	
定义	<ul style="list-style-type: none">是指是对空闲块的组织和和管理，包括空闲块的组织，分配，回收
方法1：空闲表法	<ul style="list-style-type: none">表内容 = 空闲区第一个块号 + 该空闲区的块个数
方法2：空闲链表法	<ul style="list-style-type: none">每个空闲块里有一个指针指向下一个空闲块
方法3：位示图法	<ul style="list-style-type: none">0表示盘块空闲，1表示盘块被分配
方法4：成组链接法	<ul style="list-style-type: none">结合空闲表和空闲链表的优点，克服表长的缺点

虚拟文件系统VFS	
目的	<ul style="list-style-type: none">为用户体用文件系统操作的统一结构，屏蔽了不同文件系统差异和操作细节
特性	<ol style="list-style-type: none">能提高系统性能不是一种实际的文件系统只存在与内存中，不存在与任何外存空间中在系统启动时建立，在系统关闭时消亡
VFS的数据结构	<ol style="list-style-type: none">超级块对象索引节点对象目录项对象文件对象

用户空间，系统调用，虚拟文件系统，缓存，文件系统和存储之间的关系



分区和安装

- 一个磁盘可划分为多个区，每个分区都可以创建单独的文件系统，每个分区都可包含不同的操作系统
- 文件在使用前必须先安装（即挂载）

目录

目录管理要求

- 实现“按名存取”
- 要提高目录的检索速度
- 需要提供用于控制访问文件的信息
- 允许不同用户对不同文件采用系统的名字

	定义	优点	缺点
单级目录结构	<ul style="list-style-type: none">整个文件系统只建立一张目录表每个文件占一个目录项		<ul style="list-style-type: none">查找速度慢文件不允许重名不利于文件共享不适合多用户的OS
两级目录结构	<ul style="list-style-type: none">文件目录分为主文件目录MDF和用户文件目录UFDMDF记录用户名UFD所在的存储位置UFD记录用户文件的FCB信息	<ul style="list-style-type: none">解决了多用户之间的文件重名问题文件系统可以在目录上实现访问限制	<ul style="list-style-type: none">缺乏灵活性，不能对文件分类
▲型目录结构	<ul style="list-style-type: none">使用绝对路径，相对路径，当前路径的结构大多OS采用这种目录结构	<ul style="list-style-type: none">可以很方便的对文件进行分类能够有效地进行文件的管理和保护	<ul style="list-style-type: none">利于文件共享查找文件增加了磁盘访问次数，会影响查询速度
无环图目录结构	<ul style="list-style-type: none">在树形目录结构上加入有向边，组成一个有向无环图	<ul style="list-style-type: none">实现了文件共享	<ul style="list-style-type: none">使系统的管理变得更加复杂

目录的操作

搜索文件	创建文件	删除文件		
创建目录	删除目录	移动目录	修改目录	显示目录

目录的查询

概念	<ul style="list-style-type: none">目录查询通过在磁盘上反复搜索完成，需要不断进行I/O操作，开销大可以把当前使用的文件目录复制到内存，从而降低磁盘操作次数，提高系统速度			
实现方法		定义	优点	缺点
	线性列表 [对应线性查找]	<ul style="list-style-type: none">采取线性列表存储文件目录项	<ul style="list-style-type: none">实现简单	<ul style="list-style-type: none">查找费时
	哈希表 [对应散列查找]	<ul style="list-style-type: none">采取哈希表存储文件目录项	<ul style="list-style-type: none">查找迅速插入删除简单	<ul style="list-style-type: none">需要一些措施来避免冲突

文件共享

概念	<ul style="list-style-type: none">文件共享使多个用户共享同一个文件，系统只需保留该文件的一个副本				
文件共享方式	<table><tr><td>基于索引节点的关系方式 【硬链接】</td><td><ul style="list-style-type: none">硬链接就是多个指针指向一个索引节点只要还有一个指针在，索引节点就不会被删除文件的物理地址和其他文件属性信息放在索引节点中硬链接不可用于跨文件系统硬链接查找速度比软链接快</td></tr><tr><td>基于符号链实现文件共享 【软链接】</td><td><ul style="list-style-type: none">软链接相当于重新创建一个文件新文件只包含被链接文件的路径名软链接可以跨文件系统</td></tr></table>	基于索引节点的关系方式 【硬链接】	<ul style="list-style-type: none">硬链接就是多个指针指向一个索引节点只要还有一个指针在，索引节点就不会被删除文件的物理地址和其他文件属性信息放在索引节点中硬链接不可用于跨文件系统硬链接查找速度比软链接快	基于符号链实现文件共享 【软链接】	<ul style="list-style-type: none">软链接相当于重新创建一个文件新文件只包含被链接文件的路径名软链接可以跨文件系统
基于索引节点的关系方式 【硬链接】	<ul style="list-style-type: none">硬链接就是多个指针指向一个索引节点只要还有一个指针在，索引节点就不会被删除文件的物理地址和其他文件属性信息放在索引节点中硬链接不可用于跨文件系统硬链接查找速度比软链接快				
基于符号链实现文件共享 【软链接】	<ul style="list-style-type: none">软链接相当于重新创建一个文件新文件只包含被链接文件的路径名软链接可以跨文件系统				