# C++

###### C++特点的理解：

1封装

封装是把客观事物封装成抽象的类，并且类可以把自己的数据和方法只让可信的类和对象操作，对不信任的进行信息隐藏。

2继承

继承是可以使用现有类的所有功能，并无需重新编写原来类的情况下对这些功能进行扩展，重要通过父类，子类（基类派生类）的方式，继承是从一般到特殊的过程，从而扩展已实现的代码

3多态

多态的主要机制是重载和覆盖（重写），重载是非面向对象的，而覆盖也就是通过虚函数实现多态的过程是动态的，面向对象的。虚函数其实就是一个接口，多种实现。使基类与派生类处在同一个级别，能够通过派生类指针赋值给基类类型的指针。

###### 重写（覆盖）和重载的区别：

重写（覆盖）必须继承，重载不用

重写（覆盖）方法名，参数树木都相同，参数类型兼容。重载方法名相同，参数列表不同。

重写（覆盖）方法大于等于父类的方法，重载不同

重写（覆盖）不可以抛出父类没有抛出的异常，可以抛出运行时异常

###### 虚函数和纯虚函数的区别：

纯虚函数是指基类的虚函数并不定义函数体并在后面加上=0；累里声明虚函数是为了让这个函数在子类里能被覆盖。纯虚函数是个函数接口，是个函数声明，需要留到子类中去实现。

带纯虚函数的类叫虚基类，这种基类不能声明对象，只能被继承，并且重写虚函数后才能被使用，这样的类叫抽象类。

（说明）：使用纯虚函数，系统要有一定的空间开销，当一个类带有虚函数时，编译系统会为该类构造一个虚函数表，它是一个数组指针，存放每个虚函数的入口地址。

###### （虚）构造和析构函数：

1.虚析构函数：析构函数的作用是在对象撤销之前把类对象从内存中撤销，通常只执行基类的析构函数，不会执行派生类的析构函数，若将析构函数定义为虚函数，则可以解决这个问题。

2.构造函数不能为虚函数的原因：

从存储空间的角度来说，虚函数对应一个指向虚函数表的指针，这个指向虚表的指针是存储在对象的内存空间的，如果构造函数是虚，就要通过虚表调用，但是对象还没实例化，内存空间还没有，找不到虚表。

从使用角度来说，虚函数是作用于信息不全，能使重载函数得到对应的调用的。构造函数是要初始化实例的，使用虚函数没有实际意义。

基类和派生类构造函数的调用顺序是先从基类再到派生类，而虚函数是根据具体情况调用派生类的相应函数，这违反了正常的调用顺序。

###### 友元函数的优缺点：

优点：可以使定义为友元函数或者友元类的函数直接访问另一个类中私有成员和受保护成员，提高效率，方便编程。

缺点：破坏了类的封装性，提供了安全漏洞。

###### Static：

全局变量：全局变量的作用域不仅可以在本文件中使用，在其他文件中也可以使用

静态全局变量（static全局数据区分配）：静态全局变量只能在本文件中使用，在声明它的整个文件都是可见的，在文件外不可见。

局部变量：局部变量作用域为局部作用域，结束后自动销毁。

静态局部变量（static全局数据区分配）：静态局部变量的生存期在程序运行期间都存在，作用域为局部作用域，下一次会引用上一次访问的值。

静态函数：静态函数，只在声明它的文件可见，不能被其他文件使用。其他文件中可以定义同名函数，不发生冲突。

###### Static和const关键字的作用：

Static关键字：

1、函数体内static变量的作用域为该函数体，内存只分配一次，下次调用维持上一次的值。

2、模块内的static全局变量和static函数作用域为模块内。

3、类中的static成员变量属于整个类所有，对类中的对象只有一份拷贝。在类中的static成员函数属于整个类所有，这个函数没有this指针只能调用类的static成员函数。

Const关键字：

想要阻止一个变量改变，用const关键字，在定义const变量时，必须要初始化，以后就不可以改变了。

可以定义指针为const，指针所指数据为const，二者都为const。

对一个函数，参数可以定义为const，表明输入参数，函数内部不能改变其值。

类的成员函数可以定义为常函数，不能改变类的成员变量

类的成员函数，返回值可以定义为const，使返回值不为左值。

###### 拷贝构造函数为什么要const A&可不可以不引用：

如果不引用，相当于按值传递，会无穷无尽的调用拷贝构造函数。

1. CExample(**const** CExample& C)
2. {
3. a = C.a;
4. }

**什么时候调用拷贝构造函数**

1、同类型的一个对象初始化另一个对象。

2、作为实参传递给一个函数。

3、在函数体内返回一个对象。

4、初始化序列容器的元素时。

5、列表的方式初始化数组元素时。

###### 能不能同时用static和const修饰成员函数：

不能，定义为const的成员函数为确保函数不能改变类的状态，会在函数中添加一个隐士的参数const this\* 但是成员定义为static是不能存在this指针的，两者冲突。

###### Const和define的区别：

1、define在预处理阶段执行

Const在编译阶段执行

2、define无类型

Const有类型

3、define仅仅是展开，有多少地方展开替换多少次

Const在内存中分配。

###### 内联函数和define宏定义的区别：

内联函数在运行时可调试，宏不行。

编译器会对内联函数参数类型做安全检测或自动类型转换，宏不会。

内联函数可以访问类的成员变量，宏不行。

在类中声明同时定义的成员函数，自动转化为内联函数。

###### New，malloc和delete，free的区别：

Malloc和free是C语言的库函数，使用时不会调用构造函数和析构函数。

New和delete是C++的运算符，会调用构造函数和析构函数。

New可以自动计算所需大小，malloc需要我们计算字节数。

New返回一个分配类型相同的类型指针，而malloc返回的类型为void需要强行转化为与需要相符的类型指针。

New和malloc都可以检查出内存泄漏，但是new能指明是哪一行。

###### C++编译程序占用内存：

栈区：函数参数，局部变量

堆区：类似链，程序员分配

全局区：全局变量，静态变量

文字常量区：常量字符串

程序代码区

###### 内存溢出和内存泄漏：

内存溢出是申请内存是，没有足够的内存工期是用，出现out of memory

内存泄漏是程序运行完成后，无法释放已申请的内存空间，容易堆积。

防止内存泄漏的方法：

1. 良好的编成习惯。
2. 重载new，delete将内存以链表形式自动管理，使用后从链表删除。
3. 智能指针，定义一个指针类添加一个析构函数，指针释放之前自动释放内存。
4. 使用内存管理工具（vld.h）

###### 结构体和共同体的区别：

都是由多种不同数据类型成员组成，但在任何同一时刻，共同体中只存放一个被选中成员，而结构体所有成员都在。

结构体长度为总长度，共同体长度为最长的元素的长度，结构体每个元素有自己的内存空间，共同体所有元素不能同时占用他的内存空间。

对共同体不同成员赋值，会对其他成员重写，而结构体则互不影响。

###### 结构体和类的区别：

1.struct只是作为一种数据类型定义，不能用于面向对象编程。

2.对于成员访问权限及继承方式，class中默认的是private，而struct中默认的是public。除此之外，class还可以用于表示摸板类型，struct则不行。

###### Strcpy和memcpy的区别

strcpy字符串复制，不仅会复制其字符串就连其结尾的字符‘\0’也会被复制过去，其复制遇到\0后就结束了；

而memcpy就不一样了，他是内存复制，他不仅可以复制字符串还可以复制任意内容，如字符串数组，结构体等；而且memcpy并不会遇到\0就结束，而是复制你在第三个参数中指定的字节数。

相对而言，memcpy要比strcpy用途要广泛的多，一般我们只需要复制字符串就可以选择strcpy，但是数组或者结构体，那就不要犹豫了，直接上memcpy吧

void \*memset(void \*s, int ch, size\_t n);

函数解释：将s中当前位置后面的n个字节 （typedef unsigned int size\_t ）用 ch 替换并返回 s 。

memset：作用是在一段内存块中填充某个给定的值，它是对较大的结构体或数组进行清零操作的一种最快方法

###### Hash函数：

Hash函数的构造方法：

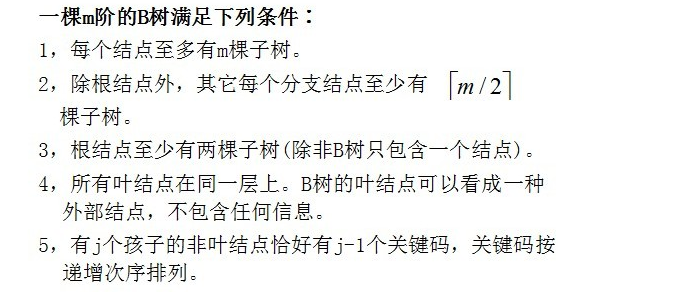
1. 除留取余法
2. 随机数法
3. 平方取中法

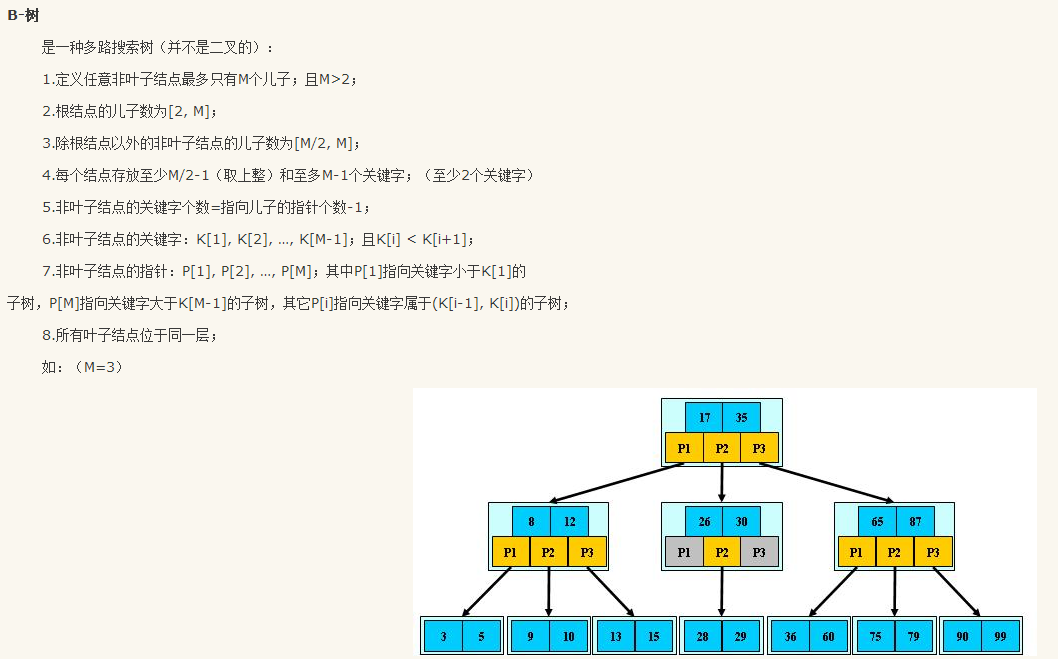
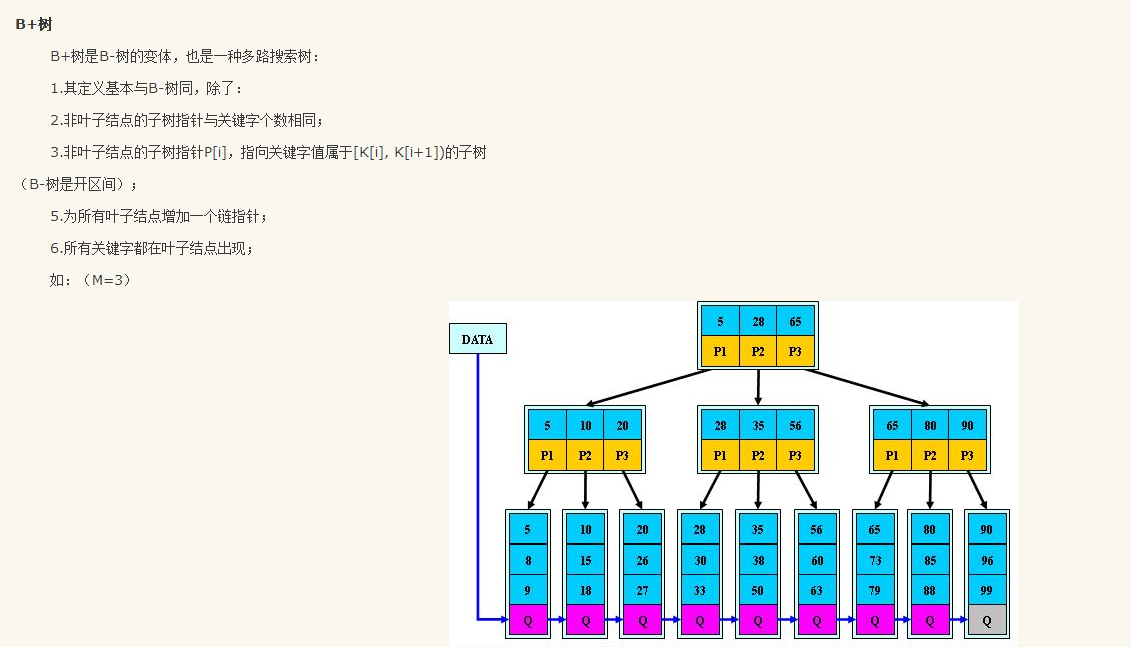
解决Hash冲突的方法：

1. 开放地址法：线性探测法，线性补偿探测法，伪随机探测
2. 拉链法
3. 再散列（双，重复Hash）

拉链法处理冲突简单，相同的key放在一个链上，且无堆积现象，即非同义词决不会发生冲突，因此平均查找长度较短；  
　  
  由于拉链法中各链表上的结点空间是动态申请的，故它更适合于造表前无法确定表长的情况

###### B数（平衡多路查找树）





红黑树

性质1. 。节点是红色或黑色。

性质2. 根节点是黑色。

性质3 每个叶节点（NIL节点，空节点）是黑色的。

性质4 每个红色节点的两个子节点都是黑色。(从每个叶子到根的所有路径上不能有两个连续的红色节点)

性质5. 从任一节点到其每个叶子的所有路径都包含相同数目的黑色节点。

###### Hash函数相对B树的：

优点：速度快，无论哈希表中有多少数据，查如何删除理论上最优需要O（1），B树是O(logN)级别

缺点：1，扩展性差，需要提前预测数据量大小

2，不能有序遍历数据

3，B树磁盘I/O次数少

###### C++中为什么使用模板类？

（1）可用来创建动态增长和减小的数据结构和数据类型

（2）它在编译时而不是运行时检查数据类型，保证了类型安全

（3）它是平台无关的，具有可移植性，与类型无关具有可复用性

###### 指针和引用的区别?

1. 引用必须被初始化，指针不必；

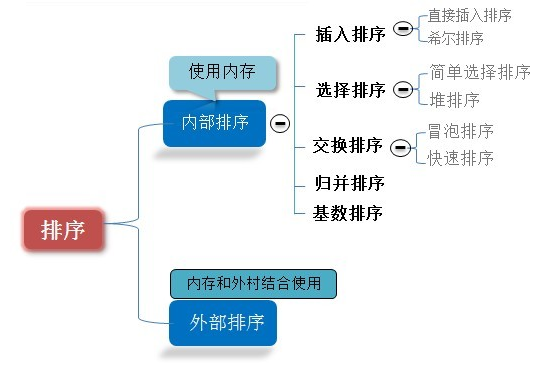
2. 引用初始化后不能改变，指针可以改变所指的对象；

3. 不存在指向空值的引用，但是存在指向空值的指针。

###### 引用和多态有什么关系？

引用是除了指针外另一个可以产生多态效果的手段，一个基类的引用可以指向它的派生类的实例。

###### 八大排序算法：



###### Vector中size（）和capacity（）的区别

1. size（）是可存储元素的总数。
2. capacity（）是必须分配内存空间之前可存储元素个数。

###### 顺序性容器

1. vector：快速随机访问，插入删除操作效率慢。
2. deque：双端队列，小片的连续内存，支持双端插入数据。
3. list：双向链表：内存空间不连续。随机存储很慢，插入删除很快。

###### 关联容器

1、map：map内部有一颗红黑树（自平衡 二叉树），自动排序。

2、unordered\_map : Hash结构。

3、set ： 红黑树，元素唯一，不存重复的。要改变元素值必须先删除旧的再插入新的。

4、queue 5、stac

**红黑树的五大特点：**

I、红黑树的五个性质：  
1）每个结点要么是红的，要么是黑的。  
2）根结点是黑的。  
3）每个叶结点，即空结点（NIL）是黑的。  
4）如果一个结点是红的，那么它的俩个儿子都是黑的。  
5）对每个结点，从该结点到其子孙结点的所有路径上包含相同数目的黑结点。

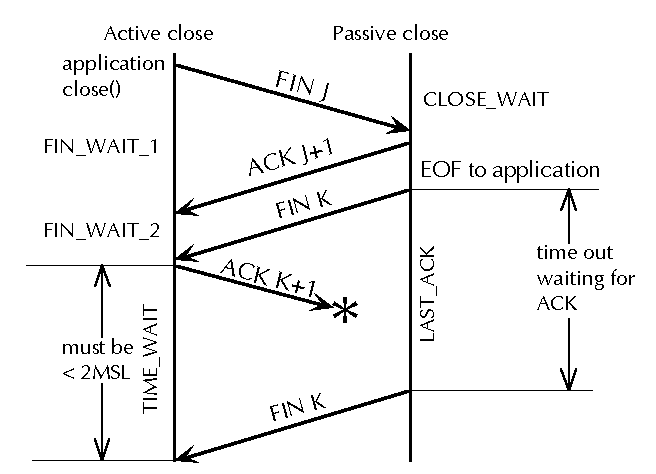
### 堆和栈究竟有什么区别



# 计算机网络

###### TCP/IP三次握手：

1. 客户端发送syn（建立连接时同步信号用）包到服务器，并进入syn\_send状态，等待服务器确认。
2. 服务器收到syn包，确认客户的syn包，同时自己发送一个syn+ACK包，此时达到syn\_receive状态
3. 客户端收到syn+ack包再向服务器确认，此时包发送完毕，客户端和服务器进入到established状态，完成三次握手。



MSL数据报在网络中单向传递认为其失效的时间，由于发送确定失效，接受确定失效是往返过程所以用2倍的MSL时间

###### TCP/UDP的区别：

1. TCP提供面向连接的、可靠的数据流传输，而UDP提供的是非面向连接的、不可靠的数据流传输。
2. TCP传输单位称为TCP报文段，UDP传输单位称为用户数据报。
3. TCP注重数据安全性，UDP数据传输快，因为不需要连接等待，少了许多操作，但是其安全性却一般。

###### TCP的拥塞控制：

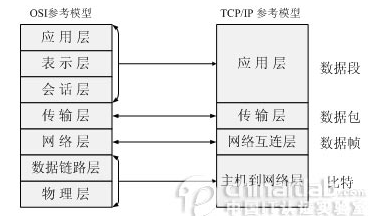
计算机网络中的带宽，交换节点中的缓存和处理机都是网络资源。在某段时间，若对网络中某一资源的需求超过了网络所能提供资源的可用部分，网络就会变坏，发生拥塞。

拥塞控制就是防止过多的数据注入网络，这样可以使网络中的路由不至过载

1. 慢开始(慢启动)算法：不要一开始就发大量数据，先探测网络的拥塞程度，从小到大逐渐增加***拥塞窗口（cwdn）***的大小。（发送方维持一个叫拥塞窗口的状态变量，大小取决于网络的拥塞程度）的大小。慢启动一点也不慢，只不过是起点低。
2. 拥塞避免：拥塞窗口可以很快的增长上来，从而最大程度的利用网络带宽资源，但是并不能让它一直增长下去，TCP使用了一个叫做慢启动门限的变量来控制，慢启动阶段结束后到达拥塞控制阶段。
3. 快重传：快重传即接收方在收到一个失效报文段后就立刻发出重复确认，而不是自己发送数据时稍带着确认。发送方一旦收到3个重复确认就立刻重传对方未收到的报文段，不必等到重传计时器重新发出重传指令。
4. 快恢复：当发送端收到连续三个重复的ack时，重新设置慢开始门限。

###### 快重传，慢启动





###### 1.       OSI ， TCP/IP ，五层协议的体系结构

**OSI分层**（7层） ：物理层、数据链路层、网络层、传输层、会话层、表示层、应用层。

**TCP/IP分层**（4层） ：主机到网络层、网络互联层、传输层、应用层。

**五层协议**（5层） ：物理层、数据链路层、网络层、运输层、应用层。

每一层的作用如下 ：

**物理层**：该层为上层协议提供了一个传输数据的物理媒体。

**数据链路层**：数据链路层在不可靠的物理介质上提供可靠的传输。该层的作用包括：物理地址寻址、数据的成帧、流量控制、数据的检错、重发等。

**网络层** ：网络层负责对子网间的数据包进行路由选择。此外，网络层还可以实现拥塞控制、网际互连等功能。

**传输层** ：第一个端到端，即主机到主机的层次。提供端到端的、可靠的或不可靠的传输。此外，还有差错控制和流量控制问题。

**会话层** ：会话层管理主机之间的会话进程，即负责建立、管理、终止进程之间的会话。会话层还利用在数据中插入校验点来实现数据的同步。

**表示层** ：表示层对上层数据或信息进行变换以保证一个主机应用层信息可以被另一个主机的应用程序理解。表示层的数据转换包括数据的加密、压缩、格式转换等。

**应用层** ：为操作系统或网络应用程序提供访问网络服务的接口。

###### 2.       IP 地址的分类

A类地址：以0开头，第一个字节范围：0~127；

B类地址：以10开头，第一个字节范围：128~191；

C类地址：以110开头，第一个字节范围：192~223；

D类地址：以1110开头，第一个字节范围为224~239；

###### 3.       ARP 协议的工作原理

首先，每台主机都会在自己的ARP缓冲区中建立一个 ARP列表，以表示IP地址和MAC地址的对应关系。当源主机需要将一个数据包要发送到目的主机时，会首先检查自己 ARP列表中是否存在该 IP地址对应的MAC地址，如果有，就直接将数据包发送到这个MAC地址；如果没有，**就向本地网段发起一个ARP请求的广播包**，查询此目的主机对应的MAC地址。此ARP请求数据包里包括源主机的IP地址、硬件地址、以及目的主机的IP地址。网络中所有的主机收到这个ARP请求后，会检查数据包中的目的IP是否和自己的IP地址一致。如果不相同就忽略此数据包；如果相同，该主机首先将发送端的MAC地址和IP地址添加到自己的ARP列表中，如果ARP表中已经存在该IP的信息，则将其覆盖，然后给源主机发送一个 ARP响应数据包，告诉对方自己是它需要查找的MAC地址；源主机收到这个ARP响应数据包后，将得到的目的主机的IP地址和MAC地址添加到自己的ARP列表中，并利用此信息开始数据的传输。如果源主机一直没有收到ARP响应数据包，表示ARP查询失败。

###### 4.       路由设备与相关层

物理层 ：中继器（Repeater，也叫放大器），集线器。

数据链路层 ：网桥，交换机。

网络层 ：路由器。

网关 ：网络层以上的设备。

###### 5.       常见的路由选择协议，以及它们的区别

常见的路由选择协议有：RIP协议、OSPF协议。

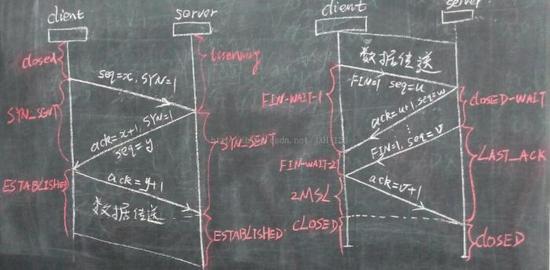
RIP协议 ：底层是贝尔曼福特算法，它选择路由的度量标准（metric)是跳数，最大跳数是15跳，如果大于15跳，它就会丢弃数据包。

OSPF协议 ：底层是迪杰斯特拉算法，是链路状态路由选择协议，它选择路由的度量标准是带宽，延迟。

###### 7.       TCP 的可靠性如何保证？

TCP的可靠性是通过顺序编号和确认（ACK）来实现的。

###### 8.       TCP 三次握手和四次挥手的全过程



###### 10.   在浏览器中输入 [www.baidu.com](http://www.baidu.com/)后执行的全部过程

**事件顺序**

(1) 浏览器获取输入的域名www.baidu.com

(3) 域名系统DNS解析出百度服务器的IP地址

(4) 浏览器与该服务器建立TCP连接(默认端口号80)

(5) 浏览器发出HTTP请求，请求百度首页

(6) 服务器通过HTTP响应把首页文件发送给浏览器

(7) TCP连接释放

(8) 浏览器将首页文件进行解析，并将Web页显示给用户。

**涉及到的协议**

(1) 应用层：HTTP(WWW访问协议)，DNS(域名解析服务)

(2) 传输层：TCP(为HTTP提供可靠的数据传输)，UDP(DNS使用UDP传输)

(3) 网络层：IP(IP数据数据包传输和路由选择)，ICMP(提供网络传输过程中的差错检测)，ARP(将本机的默认网关IP地址映射成物理MAC地址)

###### HTTP 协议包括哪些请求？

**GET**：请求读取由URL所标志的信息。

**POST**：给服务器添加信息（如注释）。

**PUT**：在给定的URL下存储一个文档。

**DELETE**：删除给定的URL所标志的资源。

###### HTTP 中， POST 与 GET 的区别

(1)Get是从服务器上获取数据，Post是向服务器传送数据。

(2)Get是把参数数据队列加到提交表单的Action属性所指向的URL中，值和表单内各个字段一一对应，在URL中科院看到。

(3)Get传送的数据量小，不能大于2KB；post传送的数据量较大，一般被默认为不受限制。

(4)根据HTTP规范，GET用于信息获取，而且应该是安全的和幂等的。

I.所谓 安全的 意味着该操作用于获取信息而非修改信息。换句话说，GET 请求一般不应产生副作用。就是说，它仅仅是获取资源信息，就像数据库查询一样，不会修改，增加数据，不会影响资源的状态。

II. 幂等 的意味着对同一URL的多个请求应该返回同样的结果。

###### TCP/IP 中，每一层对应的协议

**网络层** ：IP协议、ICMP协议、ARP协议、RARP协议。

**传输层** ：UDP协议、TCP协议。

**应用层** ：FTP（文件传送协议）、Telenet（远程登录协议）、DNS（域名解析协议）、SMTP（邮件传送协议），POP3协议（邮局协议），HTTP协议。

###### TCP 对应的协议和 UDP 对应的协议

TCP对应的协议：

（1） FTP ：定义了文件传输协议，使用21端口。常说某某计算机开了FTP服务便是启动了文件传输服务。下载文件，上传主页，都要用到FTP服务。

（2） Telnet ：它是一种用于远程登陆的端口，用户可以以自己的身份远程连接到计算机上，通过这种端口可以提供一种基于DOS模式下的通信服务。如以前的BBS是-纯字符界面的，支持BBS的服务器将23端口打开，对外提供服务。

（3） SMTP ：定义了简单邮件传送协议，现在很多邮件服务器都用的是这个协议，用于发送邮件。如常见的免费邮件服务中用的就是这个邮件服务端口，所以在电子邮件设置-中常看到有这么SMTP端口设置这个栏，服务器开放的是25号端口。

（4） POP3 ：它是和SMTP对应，POP3用于接收邮件。通常情况下，POP3协议所用的是110端口。也是说，只要你有相应的使用POP3协议的程序（例如Fo-xmail或Outlook），就可以不以Web方式登陆进邮箱界面，直接用邮件程序就可以收到邮件（如是163邮箱就没有必要先进入网易网站，再进入自己的邮-箱来收信）。

（5）HTTP协议： 是从 Web 服务器传输超文本到本地浏览器的传送协议。

UDP对应的协议：

（1） DNS ：用于域名解析服务，将域名地址转换为IP地址。DNS用的是53号端口。

（2） SNMP ：简单网络管理协议，使用161号端口，是用来管理网络设备的。由于网络设备很多，无连接的服务就体现出其优势。

（3） TFTP (Trival File Transfer Protocal)，简单文件传输协议，该协议在熟知端口69上使用UDP服务。

###### 特殊的IP地址

**（1）网络地址**

IP地址由网络号（包括子网号）和主机号组成，网络地址的主机号为全0，网络地址代表着整个网络。

**（2）广播地址**

广播地址通常称为直接广播地址，是为了区分受限广播地址。

广播地址与网络地址的主机号正好相反，广播地址中，主机号为全1。当向某个网络的广播地址发送消息时，该网络内的所有主机都能收到该广播消息。

**（3）组播地址**

D类地址就是组播地址。

先回忆下A，B，C，D类地址吧

A类地址以00开头，第一个字节作为网络号，地址范围为：0.0.0.0~127.255.255.255；

B类地址以10开头，前两个字节作为网络号，地址范围是：128.0.0.0~191.255.255.255;

C类地址以110开头，前三个字节作为网络号，地址范围是：192.0.0.0~223.255.255.255。

D类地址以1110开头，地址范围是224.0.0.0~239.255.255.255，D类地址作为组播地址（一对多的通信）；

E类地址以1111开头，地址范围是240.0.0.0~255.255.255.255，E类地址为保留地址，供以后使用。

Notice：只有A,B,C有网络号和主机号之分，D类地址和E类地址没有划分网络号和主机号。

**（4）255.255.255.255**

该IP地址指的是受限的广播地址。受限广播地址与一般广播地址（直接广播地址）的区别在于，受限广播地址之只能用于本地网络，路由器不会转发以受限广播地址为目的地址的分组；一般广播地址既可在本地广播，也可跨网段广播。例如：主机192.168.1.1/30上的直接广播数据包后，另外一个网段192.168.1.5/30也能收到该数据报；若发送受限广播数据报，则不能收到。

Notice：一般的广播地址（直接广播地址）能够通过某些路由器（当然不是所有的路由器），而受限的广播地址不能通过路由器。

**（5）0.0.0.0**

常用于寻找自己的IP地址，例如在我们的RARP，BOOTP和DHCP协议中，若某个未知IP地址的无盘机想要知道自己的IP地址，它就以255.255.255.255为目的地址，向本地范围（具体而言是被各个路由器屏蔽的范围内）的服务器发送IP请求分组。

**（6）回环**地址

127.0.0.0/8被用作回环地址，回环地址表示本机的地址，常用于对本机的测试，用的最多的是127.0.0.1。

**（7）A、B、C类私有地址**

私有地址(private address)也叫专用地址，它们不会在全球使用，只具有本地意义。

**A类私有地址**：10.0.0.0/8，范围是：10.0.0.0~10.255.255.255

**B类私有地址**：172.16.0.0/12，范围是：172.16.0.0~172.31.255.255

**C类私有地址**：192.168.0.0/16，范围是：192.168.0.0~192.168.255.255

###### NAT 协议、 DHCP 协议、 DNS 协议的作用

**NAT协议** ：网络地址转换(NAT,Network AddressTranslation)属接入广域网(WAN)技术，

是一种将私有（保留）地址转化为合法IP地址的转换技术，它被广泛应用于各种类型Internet接入方式和各种类型的网络中。原因很简单，NAT不仅完美地解决了lP地址不足的问题，而且还能够有效地避免来自网络外部的攻击，隐藏并保护网络内部的计算机。

**DHCP协议**：动态主机设置协议（Dynamic Host ConfigurationProtocol, DHCP）

是一个局域网的网络协议，使用UDP协议工作，主要有两个用途：给内部网络或网络服务供应商自动分配IP地址，给用户或者内部网络管理员作为对所有计算机作中央管理的手段。

**DNS协议** ：DNS 是域名系统 (Domain Name System) 的缩写，是因特网的一项核心服务，它作为可以将域名和IP地址相互映射的一个分布式数据库，能够使人更方便的访问互联网，而不用去记住能够被机器直接读取的IP数串

# 操作系统

**Linux操作系统启动全过程：**

1. 加载内核：操作系统接管硬件以后首先进入/boot目录下加载内核文件
2. 启动初始化程序：/sbin/init/初始化系统环境
3. 确定运行级别：很多开机启动项，要确定什么级别下初始化哪些开机启动项
4. 加载开机启动项:/etc/init,d
5. 用户登录
6. 命令行界面

**系统调用：**

系统调用是指用户程序向操作系统内核申请更高权限的服务，系统调用提供了用户和操作系统之间的接口，[内核](http://baike.baidu.com/item/%E5%86%85%E6%A0%B8)提供一系列具备预定功能的[多内核](http://baike.baidu.com/item/%E5%A4%9A%E5%86%85%E6%A0%B8)[函数](http://baike.baidu.com/item/%E5%87%BD%E6%95%B0)，通过一组称为系统调用（system call)的接口呈现给用户。系统调用把[应用程序](http://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F)的请求传给[内核](http://baike.baidu.com/item/%E5%86%85%E6%A0%B8)，调用相应的的内核[函数](http://baike.baidu.com/item/%E5%87%BD%E6%95%B0)完成所需的处理，将处理结果返回给应用程序。这里面涉及用户态和内核态两个概念。当一个用户进程执行系统调用而陷入到内核代码中时，此时处理机的特权级别为0级最高级，我们称进程处于内核态。而当进程执行用户程序时，此时处理机处于特权级别3级最低级，进程处于用户态。

**什么是linux内核**

Linux内核是操作系统的核心，包括文件管理系统，IO设备管理系统，内存管理系统，进程管理系统，网络管理系统等，linux内核是一组特权程序，具有访问硬件设备和所有主存权限的功能

**Linux 调用命令的过程**

* 1、shell不断询问是否有键盘输入，用户以enter结束键盘输入后，shell程序以空格为分隔符，得到"ls"，"-l"，"\*.c"，三个字符串。
* 2、shell发现第三个字符串包含"\*"通配符，就会在当前路径(或指定路径)下搜索满足这个通配符的文件。有bar.c和foo.c两个文件，然后展开成{"ls"，"-l"，"bar.c"，"foo.c"}四个字符串，并保存在argv这个二维字符数组中，并赋值argc为4。
* 3、shell然后搜索PATH这个全局变量保存的路径，寻找一个叫"ls"的命令。如果存在，则fork()一个子进程，调用exec()加载运行ls命令。ls的main(int argc, string \*\* argv)函数的参数就是从shell那里得来的。若不存在就会提示用户"command not found"。
* 4、ls程序执行后，会分析argc和argv，"-l"表示这是一个选项开关，表明用户想得到文件的详细信息，并把这个开关量保存，以便最后输出的时候选择信息的格式。
* 5、ls执行完既定的步骤后，退出并返回退出码，若成功则返回0，否则根据错误的不同返回不同的非零数字。
* 6、shell在得到成功退出码后，继续等待用户输入。

###### 进程和线程的区别：

进程：进程是一个程序在一个数据集上执行一次的结果

线程：线程是进程的实体，是比进程更小的CPU调度和分派的独立运行的基本单位。

区别：

一个程序至少有一个进程，一个进程至少有一个线程。

线程的划分尺度小于进程，使得多线程程序并发度高。

进程在执行过程中拥有独立的内存单元，线程是共享内存。

线程虽然有独立的程序运行入口，程序执行序列，程序出口，但是线程不能独立运行，要依存在应用程序中。

多线程集中在一个应用程序中，但是操作系统在进程调度管理以及资源分配上，并没有把多个线程看作是独立运行的应用。

###### 进程的三种状态：

1. 就绪状态：进程已经分配到CPU以外的所有必要资源，只要获得处理机便可以立即执行。
2. 执行状态：进程在处理机上执行。
3. 阻塞状态：正在执行的进程，由于等待某个事件发生无法执行时，便放弃处理机处于阻塞状态。

###### 为什么要使用线程：

1. 耗时的操作使用线程，提高了应用程序的响应。
2. 多CPU系统使用线程提高CPU利用率。
3. 改善程序结构，一个长又复杂的程序可以考虑分为多个线程。
4. 并行操作使用线程，效率更高速度更快

###### 进程间是如何通讯的

**管道**：管道是一种半双工的通信方式，数据只能单向流动，而且只能在具有亲缘关系的进程间使用。进程的亲缘关系通常是指父子进程关 系。

有名管道： 有名管道也是半双工的通信方式，但是它允许无亲缘关系进程间的通信。

**信号量**： 信号量是一个计数器，可以用来控制多个进程对共享资源的访问。它常作为一种锁机制，防止某进程正在访问共享资源 时，其他进程也访问该资源。因此，主要作为进程间以及同一进程内不同线程之间的同步手段。

**消息队列** ： 消息队列是由消息的链表，**存放在内核中并由消息队列标识符标识**。消息队列克服了信号传递信息少、管道只能承载无格式字节流以及缓冲区大小受限等缺点。

**信号** ： 信号是一种比较复杂的通信方式，用于通知接收进程某个事件已经发生。

**共享内存**：共享内存就是映射一段能被其他进程所访问的内存，这段共享内存由一个进程创建，但多个进程都可以访问。共 享内存是最快的 IPC 方式，它是针对其他进程间通信方式运行效率低而专门设计的。它往往与其他通信机制，如信号量，配合使用，来实现进程间的同步和通信。

**套接字** ： 套解口也是一种进程间通信机制，与其他通信机制不同的是，它可用于不同及其间的进程通信。

###### 进程的同步方式

临界区、互斥区、事件、信号量四种方式  
临界区（Critical Section）、互斥量（Mutex）、信号量（Semaphore）、事件（Event）的区别  
**1、临界区**：通过对多线程的串行化来访问公共资源或一段代码，速度快，适合控制数据访问。在任意时刻只允许一个线程对共享资源进行访问，如果有多个线程试图访问公共资源，那么在有一个线程进入后，其他试图访问公共资源的线程将被挂起，并一直等到进入临界区的线程离开，临界区在被释放后，其他线程才可以抢占。

（1）如果有若干进程要求进入空闲的临界区，一次仅允许一个进程进入；

　　（2）任何时候，处于临界区内的进程不可多于一个。如已有进程进入自己的临界区，则其它所有试图进入临界区的进程必须等待；

　　（3）进入临界区的进程要在有限时间内退出，以便其它进程能及时进入自己的临界区；

　　（4）如果进程不能进入自己的临界区，则应让出CPU，避免进程出现“忙等”现象。

**2、互斥量**：采用互斥对象机制。 只有拥有互斥对象的线程才有访问公共资源的权限，因为互斥对象只有一个，所以能保证公共资源不会同时被多个线程访问。互斥不仅能实现同一应用程序的公共资源安全共享，还能实现不同应用程序的公共资源安全共享 .互斥量比临界区复杂。因为使用互斥不仅仅能够在同一应用程序不同线程中实现资源的安全共享，而且可以在不同应用程序的线程之间实现对资源的安全共享。  
**3、信号量**：它允许多个线程在同一时刻访问同一资源，但是需要限制在同一时刻访问此资源的最大线程数目 .信号量对象对线程的同步方式与前面几种方法不同，信号允许多个线程同时使用共享资源，这与操作系统中的PV操作相同。它指出了同时访问共享资源的线程最大数目。它允许多个线程在同一时刻访问同一资源，但是需要限制在同一时刻访问此资源的最大线程数目。

**4、事 件： 通过通知操作的方式来保持线程的同步，还可以方便实现对多个线程的优先级比较的操作**

###### Windows内存如何管理的：

1. 虚拟内存：最适合来管理大型对象或者数据结构
2. 内存映射文件：最适合用来管理大型数据流
3. 内存堆栈：最适合管理大量小对象

**线程同步的方式包括：互斥锁、读写锁、条件变量、信号量和令牌。**

* **互斥锁和读写锁**： 提供对临界资源的保护，当多线程试图访问临界资源时，都必须通过获取锁的方式来访问临界资源。

[读写锁](http://baike.baidu.com/link?url=P728B9MpRJ5rrQHIZRPgiyl5tcC4d4ekVGHGvVlMWlnnchj3cAcVlNFwgVH0hjtApaLznxJlauOyXYQqXPPfOa)的特点如下：

1）如果有其它线程**读数据**，则允许其它线程执行读操作，但不允许写操作。

2）如果有其它线程**写数据**，则其它线程都不允许读、写操作

互斥锁: 只要被锁住，其他任何线程都不可以访问被保护的资源

* **条件变量**：提供线程之间的一种通知机制，当某一条件满足时，线程A可以通知阻塞在条件变量上的线程B，B所期望的条件已经满足，可以解除在条件变量上的阻塞操作，继续做其他事情。
* **信号量**：提供对临界资源的安全分配。如果存在多份临界资源，在多个线程争抢临界资源的情况下，向线程提供安全分配临界资源的方法。如果临界资源的数量为1，将退化为锁。
* **令牌**：一种高级的线程同步的方法。它既提供锁的安全访问临界资源的功能，又利用了条件变量使得线程争夺临界资源时是有序的。

**线程的通信方式**

# 锁机制：包括互斥锁、条件变量、读写锁  
   \*互斥锁提供了以排他方式防止[数据结构](http://lib.csdn.net/base/datastructure)被并发修改的方法。  
   \*读写锁允许多个线程同时读共享数据，而对写操作是互斥的。  
   \*条件变量可以以原子的方式阻塞进程，直到某个特定条件为真为止。对条件的[测试](http://lib.csdn.net/base/softwaretest)是在互斥锁的保护下进行的。条件变量始终与互斥锁一起使用。  
# 信号量机制(Semaphore)：包括无名线程信号量和命名线程信号量  
# 信号机制(Signal)：类似进程间的信号处理

**死锁：**

死锁的概念：两个或多个进程无限制的阻塞，相互等待的一种状态。

死锁的四个必要条件：

1. 互斥条件：一个资源一次只能被一个进程使用
2. 请求与保持条件：一个资源因请求资源阻塞，对已获得的资源保持不妨。
3. 不剥夺条件：进程所获得的资源，在未使用完之前，不能强行剥夺。
4. 循环等待条件：若干进程之间形成循环等待资源关系。

**中断和轮询的特点**

　　对I/O设备的程序轮询的方式，是早期的计算机系统对I/O设备的一种管理方式。它定时对各种设备轮流询问一遍有无处理要求。轮流询问之后，有要求的，则加以处理。在处理I/O设备的要求之后，处理机返回继续工作。尽管轮询需要时间，但轮询要比I/O设备的速度要快得多，所以一般不会发生不能及时处理的问题。当然，再快的处理机，能处理的输入输出设备的数量也是有一定限度的。而且，程序轮询毕竟占据了CPU相当一部分处理时间，因此，程序轮询是一种效率较低的方式，在现代计算机系统中已很少应用。

　　程序中断通常简称中断，是指CPU在正常运行程序的过程中，由于**预先安排或发生了各种随机的内部或外部事件**，使CPU中断正在运行的程序，而转到为响应的服务程序去处理。

　　轮询——效率低，等待时间很长，CPU利用率不高。

　　中断——容易遗漏一些问题，CPU利用率高

**分页和分段**

分段就是用户可以把自己的作业按逻辑关系划分为若干个段，每个段都是[从0开始](https://www.baidu.com/s?wd=%E4%BB%8E0%E5%BC%80%E5%A7%8B&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YYuAF-m1I-nHR1nj-bnHRY0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1RdPHn3nHms)编址，并有自己的名字和长度。

分页就是将逻辑空间地址分成若干大小相等的页存入内存

段是信息的逻辑单位，用户可见，长度可变  
页是信息的物理单位，用户透明，长度固定  
段式：若干独立的逻辑空间构成进程的非连续逻辑空间，二维地址空间  
页式：一维地址空间  
段式：物理空间不连续，但段内连续  
页式：物理空间不连续

页是信息的物理单位，分页是为实现离散分配方式，以消减内存的外零头，提高内存的利用率；或者说，分页仅仅是由于系统管理的需要，而不是用户的需要。

　　段是信息的逻辑单位，它含有一组其意义相对完整的信息。分段的目的是为了能更好的满足用户的需要。

　　页的大小固定且由系统确定，把逻辑地址划分为页号和页内地址两部分，是由机器硬件实现的，因而一个系统只能有一种大小的页面。段的长度却不固定，决定于用户所编写的程序，通常由编辑程序在对源程序进行编辑时，根据信息的性质来划分。

　　分页的作业地址空间是一维的，即单一的线性空间，程序员只须利用一个记忆符，即可表示一地址。分段的作业地址空间是二维的，程序员在标识一个地址时，既需给出段名，又需给出段内地址。

**作业调度算法**

先来先服务

轮转法

多级反馈队列算法：优先级越低，时间片越长

优先级

短作业优先

# ORACLE

###### Oracle数据库？

1. 支持多用户，大事务量的事务处理（优化器，基于成本解析SQL）
2. 数据安全性和完整性控制：
3. 数据安全：备份恢复，数据审计（audit\_trail）
4. 防止入侵：用户安全管理，权限管理，角色管理（一组权限的集合），profile管理
5. 支持分布式数据处理
6. 可移植性

###### Oracle和Mysql对比

1. Oracle是大型数据库而Mysql是中小型数据库，Oracle市场占有率达40%，Mysql只有20%左右，同时Mysql是开源的而Oracle价格非常高。

2. Oracle支持大并发，大访问量，是OLTP最好的工具。

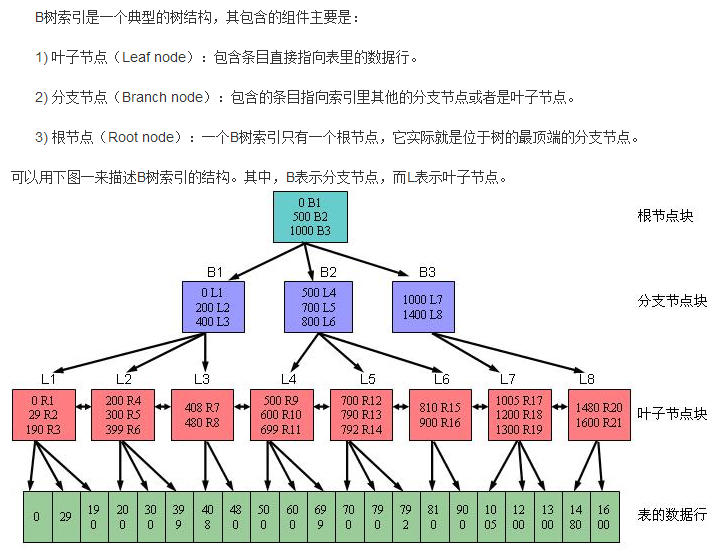
3. 安装所用的空间差别也是很大的，Mysql安装完后才152M而Oracle有3G左右，且使用的时候Oracle占用特别大的内存空间和其他机器性能。

###### 聚促索引

聚簇索引也叫簇类索引，是一种对磁盘上实际数据重新组织以按指定的一个或多个列的值排序。由于聚簇索引的索引页面指针指向数据页面，所以使用聚簇索引查找数据几乎总是比使用[非聚簇索引](http://baike.baidu.com/item/%E9%9D%9E%E8%81%9A%E7%B0%87%E7%B4%A2%E5%BC%95)快。每张表只能建一个聚簇索引，并且建聚簇索引需要至少相当该表120%的附加空间，以存放该表的副本和索引中间页。

###### Oracle中常见的索引：

1. B树索引：



1. 位图索引：对每个键值建立一个位图
2. 散列索引：根据Hash算法构建索引，检索速度很快但是不能范围查找

###### 数据库的事务

数据库的事务是指作为单个逻辑工作单元执行的一系列操作，要么完全执行，要么完全不执行。

原子性：对数据的修改，要么全都执行，要么全都不执行。

一致性：事务在完成时，必须使所有的事务，数据保持一致状态。

隔离性：由并发事务所做的修改与其他并发事务所做的修改隔离。

持久性：事务完成之后，对系统的影响是永久的

###### 存储过程和函数的区别

存储过程是用户自定义的一系列SQL语言集合，涉及表和其他对象任务，，用户可以调用存储过程。而函数通常是数据库已定义的方法，不涉及表。

###### 事前触发器和事后触发器的区别

事前触发器运行于触发事件之前，事后触发器运行于触发事件之后，语句级触发器可以在语句执行前后执行，而行级触发器所影响的每一行触发一次。

###### 索引的原理

对要查询的字段建立索引其实就是把该字段按照一定的方式排序；建立的索引只对该字段有用，如果查询的字段改变，那么这个索引也就无效了，比如图书馆的书是按照书名的第一个字母排序的，那么你想要找作者叫张三的就不能用改索引了；还有就是如果索引太多会降低查询的速度

1、如果每次都需要取到所有表记录，无论如何都必须进行全表扫描了，那么是否加索引也没有意义了。2、对非唯一的字段，例如“性别”这种大量重复值的字段，增加索引也没有什么意义。3、对于记录比较少的表，增加索引不会带来速度的优化反而浪费了存储空间，因为索引是需要存储空间的，而且有个致命缺点是对于update/insert/delete的每次执行，字段的索引都必须重新计算更新。

# LINUX

**linux的特点**  
          - 免费的/开源   
          - 支持多线程/多用户   
          - 安全性好  
          - 对内存和文件管理优越

**linux分区详解**  
硬盘的分区主要分为基本分区（Primary Portion）和扩展分区（Extension Portion）两种。只是针对一个硬盘来讲，**基本分区**和**扩展分区**的数目之和**不能大于4个**，且基本分区可以马上被使用但不能再分区。扩展分区必须再进行分区后才能使用，也就是说它必须还要进行二次分区。那么有扩展分区再分下去的是什么呢？它就是**逻辑分区**（Logical Portion），而且逻辑分区**没有数量上限制**

**shell脚本文件：**  
- 是一个文本文件   
- 命令的集合   
- 有执行的权限   
- 执行方式（./文件名）

**shell通配符**  
- \*代表多个字母或数字   
- ?代表一个字母或数字   
【案例】ls a\* ls a? ls f080[1-6].tif   
- 转义字符\   
【案例】ls /mnt/win1/My\Documents   
- 单引号：不处理任何变量和命令   
【案例】echo ‘Welcome $NAME, the date is date ’   
- 双引号：处理变量但不处理命令   
【案例】echo “Welcome $NAME, the date is date “   
- 反引号：把引号中的每个单词作为一个命令，如果是变量则先求值然后作为一个命令处理   
【案例】echo “Welcome $NAME, the date is `date` “

**常见命令：**

1.date(显示日期) cal（日历） cal 2017 cal 4 2017

2.bc(简单的是计算器)

3.sync(数据同步写入磁盘)

4.shutdown -h now（立即进行关机）      shutdown -r now （现在重新启动计算机）

5.mkdir rmdir(删除空的目录) -p连同上层目录一起删除 touch建立空文件

6.mv移动目录，或者更名

7.cat从第一行开始显示档案内容（tac从后向前） more less一页一页显示档案内容

Head只看头几行 tail只看尾几行 od二进制读取

8.file观看文件类型

9．**ln ‐s** 源目标 建立符号连接，类似于某个文件的快捷方式

9.gzip，zcat压缩读取文件。 Tar -zcvf/-zxvf

**tar -cvf /tmp/etc.tar /etc** <==仅打包，不压缩！  
**tar -zcvf /tmp/etc.tar.gz /etc** <==打包后，以 gzip 压缩  
**tar -jcvf /tmp/etc.tar.bz2 /etc** <==打包后，以 bzip2 压缩

10.startx 进入桌面

11.su- 用户登录

12.fdisk –l 查看分区具体情况

**查阅历史记录**  
- history，查看使用过的命令的历史记录   
- history 5，此项说明会显示最近使用的5个命令   
- !5，此项说明执行历史编号为5的命令

- !ls，此项说明执行最后一次以“ls”开头的命令