面试总结-操作系统

1,字节，位

字节是Byte，位是bite。字简称为B，位简称为bit

2,进程通信方式有哪些?

1,低级通信方式

进程之间一次只能传输很少的信息，采用信号量的方式就是一种实现进程互斥与同步的低级通信方式，优点：速度快，缺点：传递的信息量少，效率低，通信过程对用户不透明，因此编程复杂。

2, 高级通信

进程间一次通信可以传输大量的信息，优点是通信效率高，通信过程对用户透明，编程相对简单。主要分为：共享内存通信方式，共享文件通信方式，消息传递通信方式。

- **共享内存通信方式**

  该方式是指在内存中划分出一块内存区作为共享数据区，要求通信双方将自己的虚拟地址空间映射到共享内存分区上，通信时，发送进程将需要交换的信息写入共享内存分区中，接收进程从该共享内存分区中读取信息，从而实现进程之间的通信。优点：该方式不要求数据移动，两个需要交换的信息的进程通过对同一个共享数据区进行写入和读出操作来达到相互通信的目的。该方式可以实现多个进程之间的信息交换，但如何实现共享内存分区的互斥使用则由编程人员完成。WIndows采用该通信方式来进行进程间的通信。

- **消息缓冲通信方式(消息传递通信方式->直接通信方法)**

  发送进程利用发送命令(原语)直接将信息发送到接收进程的消息队列，而接收进程则利用接收命令(原语)从自己的消息队列中取出信息。进程间信息交换以信息为单位。根据发送者和接收者采取的不同的行为：又可以分为单向通信和双向通信。单向通信就是发送者发送完信息之后不等消息被接受者接收就继续前进。接收者接收消息后也不给发送者发送回答信息。双向通信发送者发送完消息后阻塞自己直到接收者的回答信息后才继续前进，接收者未收到信息前也阻塞自己直到接收到发送者发来的消息。

  理想过程：

  1,发送者在发送消息前，先在自己的内存空间设置一个发送区，把欲发送的消息填入其中。

  2，发送者申请一个消息缓冲区，将已经准备好的消息从发送区送到该消息缓冲区，并将发送者进程的名字，消息的开始地址，以及消息的长度等信息填入该缓冲区内，然后把该缓冲区挂载到接收进程的消息链上。

  3，接收者在接受消息之前先把自己的内存空间设置相应的接收区。

  4，接收者摘下消息链上的第一条消息，将该消息缓冲区复制到接收区，然后释放该消息缓冲区。

- **信箱通信方式(消息传递通信方式->间接通信方式)**

  该方式是指进程之间的通信要借助于称为信箱的共享数据结构实体，来暂时存放发送进程发送进程发送给接收进程的消息。该方式最大的好处就是发送方和接受方不必进行直接建立关系，也没有处理时间上的限制，发送方恶意在任何时候发送信件，接收方可以在任何时间取走信件。目前E-mail就是采用这种方式收发信件的。

- **管道通信方式(共享文件通信方式，共享的文件称为管道)**

由于内存容量的限制，使用共享内存的通信方式交换得到的信息量收到一定的限制，为了传输大量的数据，可以采用管道通信的方式。管道是指连接的两个进程之间的一个打开的共享文件。发送者可以源源不断地从管道的一端写入数据流，且每次写入的长度是可变的；接收进程可以从管道的另一端读出数据，读出的长度也是可变的。UNIX使用该方式来进行进程间通信。

3，为什么32位机最大的物理内存为4G?

地址总线是32位的，32位地址总线可以支持的内存地址代码是 4096MB，也就是有4GB的地址代码，可以编4GB个地址。

4，死锁地四个必要条件

- 互斥条件：任意时刻，一个资源只能有一个进程所占用。
- 请求和保持条件：一个进程当请求的资源无法获取得到的时候，那么就阻塞自己，而且还不释放自己已经占用的资源。
- 不可抢占条件：进程所获得的资源在未使用之前不能被其他进程抢占，而只能占用该资源的进程自己释放。
- 循环等待条件：若干进程（两个或两个以上）形成一个循环等待链，链中每一个进程都在等待该链中下一个进程所占用的资源。

5，死锁的预防

1，可以采用资源与分配策略，即每个进程在运行之前一次性申请它所需要的全部资源，并在资源未得到满足之前不投入运行。进程一旦进入运行，则分配给它的资源就一起归该进程所有且不再提出新的资源需求。优点：安全简单易实现。缺点：系统资源严重浪费，这是因为尽管一次性获得了所需要的资源，但是这些资源可能只存在进程的某个时间段内，在不使用资源的时间段内，资源就白白浪费了；由于要获取到所需要的所有资源，因此有些进程是很长时间段内都是无法进行工作的；很多进程在运行之前系统并不确切的知道它到底需要多少资源。

2，可以采用抢占资源的策略，进程在运行的过程中根据需要逐个提出资源请求，当一个已经占有了某些资源的进程又提出对新的资源请求而又未被满足的时候，则必须释放它已经获得的所有资源然后进行阻塞状态。带以后需要时再重新申请。缺点：有些资源被抢占后很可能会引起错误，这是因为一个资源在使用一段时间后又被强行抢占，有可能造成前一段时间的工作失败，即使采取一些补救措施也有可能使前后两次的执行结果是不一样的。比如打印东西，打印机是资源，一个进程在打印的过程中已经打印了一些东西，但是由于获取新资源失败，被阻塞，当其他进程使用打印机后，该进程又一次使用打印机，但是之前打印的结果已经被其他线程所破坏掉了。

3，可以采用资源有序分配，将系统的资源进行编号，并规定进程申请资源时必须按照资源编号递增（或递减）。一个进程在获取到一个资源之后，只能申请较高编号的资源（或者较低）。本质上是打破了进程之间申请的资源形成环的这种情况。缺点：进程使用的资源的顺序不一定和资源编号的顺序一致，这种情况下会造成资源的浪费，资源的编号的方式对资源的利用率有着很大的影响。

6，死锁的避免（只适用于那些事先已经知道自己需要什么资源的进程）

本质：死锁的避免就是在动态检查前提下动态的分配资源，通过检查来阻止会导致进程进入死锁状态的资源分配，从而避免了死锁的发生。

银行家算法

允许进程动态地申请资源，系统在进行资源分配之前，先计算资源分配的安全性，保证至少有一个进程能够运行到结束，并在安全运行过程中不断地回收已经运行结束地进程所占用的资源，再安全的分配给其他需要改资源的进程，直至全部进程都能够运行结束，只有满足这种情况下，系统才会进行资源的分配。类似于银行家贷款，当多个人来借钱的时候，可以给一些人需要的钱数，给另一些人的钱数不够，当第一部分人用完之后换了贷款，就可以给那些钱数不够的人。

7，关于虚拟内存的理解

- 虚拟内存与物理内存

  物理内存就是CPU的地址线可以直接进行寻址的内存空间大小。比如日常中所说的32位机的运行内存是4G，4G就是2^32得来的。那为什么要有虚拟内存呢？可以很快的想到，肯定是物理内存不能满足进程的运行需求，此时才出现了虚拟内存。举个例子：比如说：有个计算机的物理内存是1G的，但是一个程序的运行是要2G，此时物理内存就不能满足当前程序运行的条件，就是说物理内存无法装满该程序所需要的数据。这里插入一点，一个程序需要2G的内存，并不是说每时每刻这个2G的资源都是被该程序锁需要的，有些资源并不是处处都用到，所以此时势必要有一些数据要放到其他介质，比如说硬盘(外存)，带进程需要访问的时候，再通过调度把需要的资源引入到内存当中。所以说，虚拟内存是进程运行时所有内存空间的总和，并且可能一部分并不在内存当中。其实虚拟内存它并不是内存，而是对内存管理的一种抽象。

- 虚拟内存种对内存的分配方式是分页的，物理内存中对内存的分配也是分页的，虚拟内存中叫页，物理内存中叫页帧，两者之间的大小是一样的。且两者之间是一对一映射。

- 虚拟内存的运作方式：首先虚拟内存中的页和物理内存中页帧是一对一映射的，那么此时就会有个问题：虚拟内存中的页数是要大于进程在物理内存中的页帧数的，那这样的话岂不是会导致虚拟内存中有些数据用不到？事实上并不是这样，当需要访问的资源不再物理内存中的时候，此时操作系统中有个页面失效的功能，它会将最少使用的页帧，让它失效，然后将所需要的数据引入进来。这样就能保证所有的页都可以被进程所访问。

- 决定因素

  虚拟内存大小取决于地址总线数目有关，物理内存地址的大小取决于物理内存条的容量有关。

8，对于分页，分段的理解

- 两者描述 打个比方，比如说你去听课，带了一个纸质笔记本做笔记。笔记本有100张纸，课程有语文、数学、英语三门，对于这个笔记本的使用，为了便于以后复习方便，你可以有两种选择。 第一种是，你从本子的第一张纸开始用，并且事先在本子上做划分：第2张到第30张纸记语文笔记，第31到60张纸记数学笔记，第61到100张纸记英语笔记，最后在第一张纸做个列表，记录着三门笔记各自的范围。这就是分段管理，第一张纸叫段表。 第二种是，你从第二张纸开始做笔记，各种课的笔记是连在一起的：第2张纸是数学，第3张是语文，第4张英语……最后呢，你在第一张纸做了一个目录，记录着语文笔记在第3、7、14、15张纸……，数学笔记在第2、6、8、9、11……，英语笔记在第4、5、12……。这就是分页管理，第一张纸叫页表。你要复习哪一门课，就到页表里查寻相关的纸的编号，然后翻到那一页去复习

- 两者的优缺点

  在段式存储管理中，将程序的地址空间划分为若干段，如代码段，数据段，堆栈段，这样每个进程就有一个二维的地址空间，相互独立，互不干扰，段式管理的优点是：没有内存碎片(因为段的大小是可以改变的)，但是段换入换出的时候，会产生外碎片。(比如4K的段换5k的段，会产生1K的外碎片)。

  在页式存储管理中，将逻辑地址划分为固定大小的页，而物理内存也划分为同样大小的页，在程序加载的时候，可以将任意一页的数据放入内存的一个页帧当中，页式存储管理的优点是：没有外碎片(因为页的大小固定)，但是会产生内碎片(数据并没有填满页的容量)。

- 两者的不同点
  - 分页仅仅是由于系统管理的需要而不是用户的需要。比如说虚拟内存的管理就是按照分页的方式来维护的。段则是信息的逻辑单位，它含有一组其意义相对完整的信息，分段的目的是为了更好地满足用户的需求。
  - 页的大小固定且由系统所决定，而短的长度并不确定，决定于用户所编写的程序。
  - 分页的作业地址空间是一维的，即单一的线性地址空间(直接就是页数)，分段是二维的，标识一个地址的时候，不但要给出段名，还要给出段内地址（段号+该段内的具体位置）。

- 两者的结合--**段页式存储管理**
  - 分页系统能有效地地提高内存的利用率(可以从虚拟内存的角度上来看)，而分段能反映程序的逻辑结构，便于段的共享与保护。
  - 在段页式存储管理系统当中，作业的地址空间首先被分成若干个逻辑分段，每段有自己的段号，然后再将每段分成若干个大小相同地页。对于主存空间也分成大小相等的页，主存的分配以页为单位。地址组成:段号，页号，页内位移量。

*活锁

- 指线程1可以使用资源，但它很礼貌，让其他线程先使用资源，线程2也可以使用资源，但它很绅士，也让其他线程先使用资源。这样你让我，我让你，最后两个线程都无法使用资源。

*饥饿

- 是指如果线程T1占用了资源R，线程T2又请求封锁R，于是T2等待。T3也请求资源R，当T1释放了R上的封锁后，系统首先批准了T3的请求，T2仍然等待。然后T4又请求封锁R，当T3释放了R上的封锁之后，系统又批准了T4的请求……，T2可能永远等待。