一．关于进程状态回答下面的问题：(8分)

(1) 处于就绪态的进程为什么不会直接变成阻塞态？

**只有执行了某些动作，才可能引发阻塞。就绪态做不出任何动作。**

(2) 由谁将处于等待磁盘读写完成的阻塞态进程唤醒？为什么是它？

**内核。更具体说，可能是磁盘驱动，可能是I/O模块。在微内核操作系统中，还可能是担当磁盘I/O任务的进程。**

(3) 程序调试的时候有断点功能。请应用操作系统所学知识给出一种实现方法，不能改变现有硬件的功能。

**设计一个断点系统调用，在断点处调用它，将进程置于阻塞态，并提供系统调用来获得进程各种状态信息和唤醒之**

二．针对给定的信号量P操作实现，回答下面的问题：(8分)

P(s) {

s--;

while s<0

;

}

(1) 信号量数值为3和-3各表示什么含义？

**3表示可以有3次调用通过P。-3表示有三次调用正在等待**

(2) 进程P1要对信号量执行P操作，叙述这个操作可能发生的结果，并给出原因。

**如S>0，会通过；如S<=0，会等待**

(3) 进程P1在对信号量执行P、V操作时，是否属于临界区，为什么？需要做什么？

**这个实现没有保护s，所以是临界区。需要对s提供保护，比如开关中断等**

(4) 可以实现信号量数值不出现负数的信号量吗？如果可以，如何做？如果不可以，解释原因。

**可以。While s <= 0; s--;**

三．关于死锁，回答问题：(8分)

(1) 简述死锁的含义和死锁造成的后果；

**进程互相纠缠等待，谁都不能前进，就死锁了。这些进程的功能都失效了。**

(2) 对于银行家算法，如果当前剩余资源向量(Available)比某个进程的Need向量(还需要的资源向量)小，则必会发生死锁，这个论断是否对，说明理由；

**不对。比如，其它进程没有need了，然后退出释放资源，使available大于need**

(3) 为什么通常的PC机操作系统采用死锁忽略技术；

**不值得费那事儿**

(4) 一个操作系统要支持死锁恢复技术，需要操作系统提供什么支持，该如何实现，请简要说明。

**比如要能检测到死锁，保存进程恢复点的状态。发现死锁后，恢复之前的状态**

四．针对下面的文件系统要求，回答问题：(8分)

(1) 一块磁盘只用来存放词库文件，供词汇查找程序（如金山词霸）使用，此时的文件适合使用什么样的存储结构，简要说明原因。

**什么结构都有一定合理性，但需要体现出词库是随机访问的这个特点，分析要合理**

(2) 采用索引的文件存储结构在其文件控制块（FCB）中有2个索引项，第1个索引项直接存放数据块，第2个索引项存放1阶间接索引项，索引块和数据块大小都是16字节，且用4个字节表示一个盘块，问此时最大能存放文件大小是多少字节？简要写出计算过程

**（1+4）\* 16 = 80**

(3) 假设某文件大小已经达到最大值。那么第1次顺序完全读入此文件的磁盘IO次数是多少？第2次顺序完全读入此文件的磁盘IO次数又是多少？（初始化时只有FCB已经读入到内存，且系统支持磁盘缓存。）简要写出计算过程。

**5个数据块+1个inode = 6次。  
取决于cache刷新没有，答案可能是0-7次。**

五．关于CPU调度回答下面的问题：(8分)

(1) 简述CPU调度的含义；

**决定谁是下一个运行的进程**

(2) 进程的优先级随着其处于就绪队列的等待时间增加而如何变化，为什么？

**一般优先级会增加，避免饿死**

(3) 给出轮转调度(RR)的工作原理，并描述它的一个优点和一个缺点；

**每个进程运行一小会，大家轮着来。比较公平，但切换太多，且缺少实时性**

(4) 已知四个进程P1,P2,P3,P4，其到达时间分别为0,2,3,5，其执行时间分别为8,6,4,3，时间片大小为2，用RR调度算法给出调度结果（什么时候谁执行，可以画图方式表达），并计算出平均周转时间。

**设优先级相同，轮流跑位，(21+(19-2)+(14-3)+(15-5))/4=14.75**

六. 关于内存管理回答下面的问题：(10分)

(1) 什么是请求调页；给出请求掉页的两个好处；

**访问到某页面，才加载到内存。节约内存空间，减少不必要的I/O，同时容纳更多进程，提高CPU利用率**

(2) 页面置换算法中LRU算法为什么可以有效工作？什么情况下的不能有效工作？

**局部性原理决定了有效性。如程序不走寻常路，就不能有效工作了**

(3) 针对下面的双指针Clock算法（两个指针都是每秒钟移动一次，且两者指针的夹角总保持为3个页面，前一个指针负责清空访问位，后一个指针负责换出页面），假设页面访问也是1秒钟发生一次，设当前时刻的状态如下图所示，现在开始有一个页面访问序列是10, 5, 77, 4, 10（假定每秒都是先完成页面访问然后再执行Clock算法），画出5秒以后下图的样子。



**顺时针还是逆时针转；clock触发时是先移动指针，再做动作，还是先做动作再移指针，结果都不同：**