1. 概念解释（10个）

**Multipprograming：**

**即多道程序设计**

**指在内存中同时存放若干个作业，并使它们同时运行的一种程序设计技术。**

**在单处理机环境下，仅在宏观上这些作业在同时运行，而在微观上它们是在交替执行。**

**即每一时刻只有一个作业在执行，其余作业或处于阻塞状态，或处于就绪状态。**

**管程：**

一种程序结构，结构内的多个子程序（对象或模块）形成的多个工作

线程互斥访问共享资源。这些共享资源一般是硬件设备或一群变量。管程实

现了在一个时间点，最多只有一个线程在执行管程的某个子程序。管程提供

了一种机制，线程可以临时放弃互斥访问，等待某些条件得到满足后，重新

获得执行权恢复它的互斥访问

**系统调用：**

运行在使用者空间的程序向操作系统内核请求需要更高权限运行

的服务。 系统调用提供了用户程序与操作系统之间的接口。大多数系统交

互式操作需求在内核态执行

**FAT文件系统：**

* FAT(File Allocation Table,文件分配表)文件系统是Windows操作系统所使用的一种文件系统，他的发展过程经历了FAT12,FAT16,FAT32三个阶段。
* FAT文件系统用 **“簇”** 作为数据单元。一个“簇”由一连串的扇区组成，簇所含的扇区数必须是 **2的整数次幂** 。簇的最大值是64个扇区，即32kb。 **本文中一簇是4kb。**
* 所有簇从2开始进行编号，每个簇都有一个自己的编号。
* 用户文件和目录都存储在簇中。
* 文件系统分配磁盘按照簇进行分配，因此一个文件即使只有1kb，那么也会被分配4kb的空间。

**preemptive scheduling：**

当一个进程正在执行时，调度程序基于某种原则，剥夺已分配给该进程的处理机，将它分配给其他进程并使之执行。剥夺的原则有：

(1)优先权原则；(2)短进程优先原则；(3)时间片原则等

**用户级线程、内核级线程：**

在引入线程的操作系统中，线程是进程中的一个实体，是被操作系统独立调度和分派的基本单位。

线程自己基本上不拥有资源，只拥有在运行中必不可少的资源，如程序记数器、一组寄存器和栈。

但线程可与同属于一个进程的所有进程共享进程所拥有的全部资源。一个线程可以创建和撤消另一个线程。同一进程中的线程可以并发执行。

**Race Condition：**

计算机运行过程中，并发、无序、大量的进程在使用有限、独占、不可抢占的资源，

由于进程无限，资源有限，产生矛盾，这种矛盾称为竞争（Race）

由于两个或者多个进程竞争使用不能被同时访问的资源，使得这些进程有可能因为时间上推进的

先后原因而出现问题，这叫做竞争条件（Race Condition）

竞争条件分为两类:   
-Mutex（互斥）：两个或多个进程彼此之间没有内在的制约关系，但是由于要抢占使用某个临界资源（不能被多个进程同时使用的资源，如打印机，变量）而产生制约关系。   
-Synchronization（同步）：两个或多个进程彼此之间存在内在的制约关系（前一个进程执行完，其他的进程才能执行），如严格轮转法。

**阻塞IO：**

操作系统内核对于I/O只有两种方式：阻塞和非阻塞。调用阻塞I/O时，应用程序需要等待I/O完成才返回结果，阻塞I/O的一个特点是调用之后一定要等到系统内核层面完成所有操作后，调用才结束。

调用非阻塞I/O跟阻塞I/O的差别为调用之后立即返回，返回后，CPU的时间片可以用来处理其他事务，此时性能是提升的。但是非阻塞I/O的问题是：由于完整的I/O没有完成，立即返回的并不是业务层期望的数据，而仅仅是当前调用的状态。为了获取完整的数据，应用程序需要重复调用I/O操作来确认是否完成。这种重复调用判断操作是否完成的技术叫做轮询。

轮询技术主要有以下四种：

1、read

最原始、性能最低，通过重复调用来检查I/O的状态来完成完整数据的读取。在得到数据前，CPU一直好用在等待上。

2、select

改进了read，通过对文件描述符上的事件状态来进行判断。有一个较弱的限制为由于它采用一个1024长度的数组来存储状态，所以它最多可以同时检查1024个文件描述符。

3、poll

对select改进，采用链表方式避免数组长度的限制，其次它能避免不需要的检查。担当文件描述符较多时，它的性能还是低下。

4、epoll

Linux下效率最高的I/O事件通知机制，在进入轮询的时候如果没有检查到I/O事件，将会进行休眠，直到事件发生将它唤醒。它是真实利用了事件通知、执行回调的方式，而不是遍历查询，所以不会浪费CPU，执行效率较高。

5、kequeue

与epoll类似，仅在FreeBSD系统下存在。

**block device：**

是磁盘指对数据的存取是以数据块为单位的设备，典型的块设备是磁盘。

**抢占：**

**RAID(Redundant Arrays of Independent Disks)：**

其基本思想就是把多个相对便宜的硬盘组合起来，成为一个硬盘阵列

组，使性能达到甚至超过一个价格昂贵、容量巨大的硬盘。根据选择的版本

不同，RAID 比单颗硬盘有以下一个或多个方面的好处：增强数据集成度，增

强容错功能，增加处理量或容量。另外，磁盘阵列对于计算机来说，看起来

就像一个单独的硬盘或逻辑存储单元。简单来说，RAID 把多个硬盘组合成为

一个逻辑扇区，因此，操作系统只会把它当作一个硬盘。RAID 常被用在服务

器计算机上，并且常使用完全相同的硬盘作为组合

**虚拟机：**

第一种是“虚拟主机”的概念，第二种是“运行环境”的概念。这两种概念是不同的。

虚拟机”在作“虚拟主机”讲的时候，指的是操作系统内安装另一个操作系统。你如在Windows里，你可以装一个VMWare，然后在VMWare里安装一个Ubuntu Linux

虚拟机”在做“运行环境”讲的时候，指的是操作系统被安装一个支持其他软件包运行的软件。最明显的例子就是“Java虚拟机”

**Spooling：**

即同时联机外围操作，又称脱机操作。在多道程序环境下，可利用多道程序中的一道程序，

来模拟脱机的输入输出功能。即在联机条件下，将数据从输入设备传送到磁盘，或从磁盘传送到输出设备

**虚拟文件系统：**

虚拟文件系统(VFS)是一种用于网络环境的分布式文件系统，允许使用和操作系统不同的实验接口，他将文件系统通用操作和具体实现分开

**文件系统(file system)：**

包含若干文件以及其属性说明、对文件进行操纵和管理的软件，以及系统向用户提供的使用文件的接口等的集合。文件系统是操作系统的一个重要组成部分。

二、

1. **磁盘空间分配的三种方式-连续分配、链接分配、索引分配的优缺点**

**连续分配：**

**链接分配：**

**索引分配：**

1. **甘特图、平均等待时间**

**甘特图**法观察图像用CPU利用时间/总工作时间=CPU的利用率；

在甘特图中如果各个占用线是连续的则此进程在运行过程中无等待现象，否则就有等待现象，即等待现象发上在间断处

1）抢占最短作业

2）轮转（时间片=5）

3、是否死锁，哪些死锁

|  |  |  |  |
| --- | --- | --- | --- |
|  | allocation | need | available |
| p1 | 0，1，0 | 0，1，0 | 0，0，0 |
| p2 | 2，0，0 | 1，0，0 |  |
| p3 | 0，0，3 | 0，0，0 |  |
| p4 | 2，1，1 | 2，1，0 |  |
| p5 | 0，0，2 | 0，0，2 |  |

**4、外设、CPU、内存，操作系统对并行的保护措施**

1. **局部置换，FIFO置换算法，写出出物理地址顺序**

2帧，现在指向第一条命令，已分配A、B

|  |  |  |
| --- | --- | --- |
| 命令 | 地址（十六进制） |  |
| 1 | F0，F1 |  |
| 2 | EA、EB、EF |  |
| 3 | F8 |  |
| 4 | EA、EE |  |
| 5 | EF、BA |  |

1. **给出一段来自维基百科thead safty的英文，问进程、线程概念，以及为什么没有process safty（= =我把第二段翻译了一边，凑了2点）**

**进程**：程可以定义为“可与其他程序并发执行的程序J在一个数据集合上的运行过程”。进程具有**动态性、并发性、独立性、异步性**和结构特征

**线程**：在引入线程的操作系统中，线程是进程中的一个实体，是被操作系统独立调度和分派的基本单位。线程自己基本上不拥有资源，

只拥有在运行中必不可少的资源，如程序记数器、一组寄存器和栈。但线程可与同属于一个进程的所有进程共享进程所拥有的全部资源。

一个线程可以创建和撤消另一个线程。同一进程中的线程可以并发执行

1. **磁盘调度算法：已完成144，目前指向143，队列有13、54、83、941、1050、1550、1701问距离**

为了减少对文件的访问时间，应采用一种最佳的磁盘调度算法，以使各进程对磁盘的平均访问时间最少。

由于在访问磁盘时主要是寻道时间。因此，**磁盘调度的目标是使磁盘的平均寻道时间最少**

1）FIFO：先来先服务算法。这种算法的思想比较容易理解。假设当前磁道在某一位置，依次处理服务队列里的每一个磁道，

这样做的优点是处理起来比较简单，但缺点是磁头移动的距离和平均移动距离会很大。

1. SSTF：最短寻道时间算法。这种算法的本质是利用贪心算法来实现，假设当前磁道在某一位置，接下来处理的是距离当前磁道最近的磁道号，

处理完成之后再处理离这个磁道号最近的磁道号，直到所有的磁道号都服务完了程序结束。这样做的优点是性能会优于FIFO算法，但是会产生距离

当前磁道较远的磁道号长期得不到服务，也就是“饥饿”现象，因为要求访问的服务的序列号是动态产生的，即各个应用程序可能不断地提出访问不同的磁道号的请求

1. SCAN：电梯调度算法。先按照一个方向(比如从外向内扫描)，扫描的过程中依次访问要求服务的序列。当扫描到最里层的一个服务序列时反向扫描，

这里要注意，假设最里层为0号磁道，最里面的一个要求服务的序列是5号，访问完5号之后，就反向了，不需要再往里扫。结合电梯过程更好理解，

在电梯往下接人的时候，明知道最下面一层是没有人的，它是不会再往下走的

4) CSCAN：循环扫描算法。CSCAN算法，循环扫描算法，来看一下上一种算法，有什么问题。仔细一看，我们会发现，在扫描到最里面的要求服务的序列时，接着会反向，在接下来的很大一部分时间里，应该是没有要求服务的磁道号的，因为之前已经访问过了。什么意思，就是说从初始磁道号到最里层的一个磁道号之间的所有序列都已经访问过了，所以SCAN会增加等待的时间。为了解决这样的情况，CSCAN算法的思想是，访问完最里面一个要求服务的序列之后，立即回到最外层欲访问磁道。也就是始终保持一个方向。故也称之为**单向扫描调度算法**。从最里面的一个磁道立即回到最外层欲访问的磁道，这步的距离是两者磁道号差的绝对值。

各种算法应用举例：

假设当前磁头在67号，要求访问的磁道号顺序为98,25,63,97,56,51,55,55,6  (电脑随机产生的,设定最外层磁道号为100号)

FIFO算法的服务序列是：98,25,63,97,56,51,55,55,6

磁头移动的总距离distance = (98-67)+(98-25)+(63-25)+(97-63)+(97-56)+(56-51)+(55-51)+(55-55)+(55-6)

SSTF算法的服务序列是: 63,56,55,55,51,25,6,97,98

磁头移动的总距离distance = (67-63)+(63-56)+(56-55)+(55-55)+(55-51)+(51-25)+(25-6)+(97-6)+(98-97)

SCAN算法的服务序列是：63,56,55,55,51,25,6,97,98//先下楼，再上楼

磁头移动的总距离distance = (67-63)+(63-56)+(56-55)+(55-55)+(55-51)+(51-25)+(25-6)+(97-6)+(98-97)

CSCAN算法的服务序列是：63,56,55,55,51,25,6,98,97

distance = (67-63)+(63-56)+(56-55)+(55-55)+(55-51)+(51-25)+(25-6)+|6-98|+(98-97)

8、信号量：两个人，一人拿黑子，一人拿白字，差不超过M，一次只能一只手伸入棋盘

