1. ：导论（一般1个填选）

操作系统的概念：操作系统是管理计算机硬件的程序，它还为应用程序提供基础，并且充当计算机硬件和计算机用户的中介。

多处理器系统的三个主要优点：增加吞吐量、规模经济、增加可靠性。

用户模式（普通模式）和监督程序模式（管理模式/系统模式/特权模式）。用户模式位为1，监督程序模式位为0。

操作系统五大基本功能：用户接口、存储管理、设备管理、文件管理、处理器管理。

1. ：操作系统结构

用户界面：命令行界面CLI和图形用户界面GUI

系统调用和应用程序接口API。常用API有：Win32的API,POSIX API,Java API（Java虚拟机）

API编程的优点：程序的可移植性，实际的系统调用比API更为注重细节和困难。

系统调用可分为五大类：进程控制（MS-DOS单任务系统，FressBSD多任务系统）、文件管理、设备管理、信息维护和通信。

通信：两大模型。消息传递模型和内存共享模型。

系统程序，位于操作系统之上，应用程序之下。

策略和机制的区分：机制决定如何做，策略决定做什么。策略可能会随时间或位置而有所改变。

操作系统的结构：简单结构（如MS-DOS，没有很好地区分接口和功能层次），分层结构（构造和调试的简单化，但是效率较差），微内核结构（主要功能是使客户程序和运行在用户空间的各种服务之间进行通信，便于扩充操作系统，但是必须忍受由于系统功能总开销的增加而导致系统性能的下降。如TRU64 UNIX,QNX），模块化结构（允许内核提供核心服务，也能动态的实现特定的功能）

虚拟机：VMWare和Java虚拟机

1. ：进程

进程是现代分时系统的工作单元

进程一般包括代码段、程序计数器、堆栈段和数据段。

进程的五种状态及其转换关系

进程控制块PCB

进程调度：

多道程序设计的目的是无论何时都有进程在运行，从而使CPU利用率达到最大化。

分时系统的目的是在进程之间快速切换CPU以便用户在程序运行时能与其进行交互。

清楚作业队列、就绪队列和设备队列

长期调度程序（作业调度程序）：从池中选择进程，并装入内存以准备执行。

短期调度程序（CPU调度程序）：从准备执行的进程中选择进程并为之分配CPU。

分时系统引入的中期调度程序概念：将进程从内存中移出，从而降低多道程序设计的程度。

上下文切换的概念：将CPU切换到另一个进程需要保存当前进程的状态并恢复另一个进程的状态。

进程间的通信机制：

共享内存（允许以最快的速度进行方便的通信，生产者-消费者问题）和消息传递机制（通常用系统调用来实现）

消息传递机制：

直接通信（对称寻址和非对称寻址）和间接通信

阻塞（同步）和非阻塞（异步）

1. ：线程

进程是资源分配的基本单位，线程是CPU调度的基本单位

线程由线程ID、程序计数器、寄存器集合和栈组成。它与同属于同一进程的其它线程共享代码段、数据段和其他资源。

多线程编程的优点：响应度高、资源共享、经济、多处理器体系结构的利用

用户线程受内核支持，而无须内核管理，而内核线程由操作系统直接支持和管理。

多线程模型：多对一、一对一、多对多。（前者为用户线程，后者为内核线程）

系统调用fork（）的两种形式：如果调用fork（）之后立即调用exec（），那么没有必要复制所有线程，因为exec（）参数所指定的程序会替换整个进程。如果在fork（）之后另一进程并不调用exec（），那么另一进程就应复制所有线程。

线程取消：异步（立即终止）和延迟（不断检查它是否应终止）

线程池（在进程开始时创建一定数量的线程，并放入到池中以等待工作）：通常用现有线程处理请求要比等待创建新的线程要快，线程池限制了在任何时候可用线程的数量。

轻量级进程LWP的概念。（多线程模型中，用户线程和内核线程中间的数据结构）

1. ：CPU调度

分派程序的概念

清楚几个概念：吞吐量、周转时间（从开始到完成）、等待时间（在就绪队列中所花费时间之和）、响应时间（从提交请求到产生第一响应的时间）

调度算法：FCFS、最短作业优先（SJF）、可抢占SJG（最短剩余时间优先调度）、优先级调度、轮转法调度（专门为分时系统设计）、多级队列调度、多级反馈队列调度

多处理器调度（SMP）：对称多处理，处理器亲和性（SMP努力使一个进程在同一个处理器上运行），负载平衡（设法将工作负载平均分配到SMP系统的所有处理器上）

内核无法看到用户级线程，只能看到核心级线程

1. ：进程同步

临界区问题的解决方案必须满足以下三个要求：互斥、前进、有限等待

信号量：这里区分一下计数信号量和二进制信号量（互斥锁）

三个经典问题：有限缓冲问题、读写问题、哲学家进餐问题

管程

1. ：死锁

死锁的四个必要条件：互斥、占有并等待、非抢占、循环等待

资源分配图

处理死锁的三种方法：

死锁预防：确保至少一个必要条件不成立

死锁避免：资源分配图算法（每个资源一个实例）、银行家算法（重点，每个资源多个实例）

死锁恢复：死锁检测（单个实例-等待图，多个实例-类似于银行家算法），死锁恢复（进程终止、资源抢占）

1. ：内存管理

逻辑地址：CPU生成的地址

物理地址：内存单元看到的地址

编译和加载时的地址绑定方法生成相同的逻辑地址和物理地址，但是执行时的地址绑定方案将导致不同的物理地址和逻辑地址。此时的逻辑地址叫做虚拟地址。

运行时，从虚拟地址到物理地址的映射是由被称为内存管理单元完成的。

动态加载（一个子程序只有在调用时才被加载）：为了获得更好的内存空间使用率

动态链接

交换

连续内存分配：

内存通常分为两个区域：一个用于驻留操作系统，一个用户用户进程。一般来说，操作系统位于低内存。

内部碎片（分配的与进程所需的内存之差）和外部碎片（所有的可用内存之和可以满足请求，但是不连续）问题

内存分配的方法：（动态分区只可能导致外部碎片、静态分区可能导致内部碎片）

静态分区：分配为固定大小的分区

动态分区：孔的概念，选择一个空闲孔的方法：首次适应（分配第一个足够大的）、最佳适应（分配最小的足够大的）、最差适应（分配最大的）

解决外部碎片的两种方法：

紧缩：仅在重定位是动态并在运行时可用

允许物理地址空间非连续：分页和分段

分页（可能产生内部碎片，用户视角的内存和实际物理内存的分离）：

基本思想：将物理内存分为固定大小的块-帧，将逻辑内存也分为同样大小的块-页，通过页表将逻辑地址映射到物理地址。（页号和页偏移）

TLB：解决两次访存的问题

TLB包括页表中的一小部分条目，如果页码不在TLB中，那么就需要访问页表。

有效内存访问时间的计算。

页表中的有效位的含义

页表的三种结构：

层次页表：向前映射页表

哈希页表（通常用来处理超过32位地址空间）

反向页表（64位的UltraAPARC和PowerPC）

分段（支持用户视角的内存管理方案）：

基本思想：逻辑地址空间是由一组段组成的，地址指定了段名称和段内偏移。因此用户通过两个量来指定地址：段名称和偏移，通过段表实现。而在分页中，用户只指定一个地址，该地址通过硬件分为页码和偏移。

段表：每个条目包含段基地址和段界限

1. ：虚拟内存

虚拟内存的概念：将用户逻辑内存和物理内存分开

按需调页：不是将整个程序调入内存，而是将必需的页调入内存。

支持按需调页的硬件：页表、次级存储器

按需调页的有效访问时间的计算

写时复制（Win XP,Linux，Solaris）：允许父进程和子进程开始时共享同一个页面。如果任何一个进程需要对页进行写操作，就创建一个共享页的副本。

修改位（脏位）的概念和作用

页面置换算法（综合应用）：

FIFO算法：Belady异常

最优置换算法

LRU置换（最近最少使用）

最优置换和LRU置换都没有Belady异常，都属于栈算法。

近似LRU置换（使用引用位），也称为时钟算法

帧分配：

平均分配、比例分配

全局置换（允许一个进程从所有帧集合中选择一个置换帧，而不管该帧是否已分配给其他进程）和局部置换（每个进程只能从自己的分配帧中进行选择）的概念。

系统颠簸：

定义：频繁的页调度行为，如果一个进程在换页的时间上要多于执行时间，则这个进程在颠簸。

原因：CPU使用率太低或太高。

解决方法：

局部模型，缓存的基础。

工作集合模型：基于局部性假设。最近x个引用的页合称为工作集合。工作集合是程序局部的近似。防止了颠簸，并尽可能地提高了多道程序的程度，优化了CPU使用率。

通过固定定时中断和引用位，能近似模拟工作集合模型。

页错误频率：为所期望的页错误率设置上限和下限。

工作集合和页错误率：当一个新的局部按需调页时，页错误率进入波峰，一旦新局部的工作集合在内存内，页错误率开始下降。当进程进入一个新的工作集合，页错误率又一次升到波峰，然后再次降到波谷。从一个波峰的开始到下一个波峰的开始，显示了工作集合的迁移。

1. ：文件系统接口

文件系统：一组文件和目录结构

文件：记录在外存上的相关信息的具有名称的集合，文件是逻辑外存的最小分配单元。

文件属于抽象数据类型。

访问方法：

顺序访问：基于文件的磁带模型

直接访问（相对访问）：基于文件的磁盘模型，可立即访问大量信息。

目录结构：

单层目录结构：便于理解和支持

双层目录结构：解决名称冲突问题，但是对用户加以隔离

树状结构目录：最常用的目录结构，禁止共享文件和目录

绝对路径（从根开始）和相对路径（从当前目录开始）

无环图目录：允许目录含有共享子目录和文件

通用图目录

文件共享：

FTP

DFS分布式文件系统

万维网

一致性语义：评估文件系统对文件共享支持的一个重要准则。

当信息保存在计算机系统中，需要保护其安全，使之不受物理损坏（可靠性）和非法访问。

保护：

访问类型：限制可进行的文件访问类型

访问控制：根据用户身份进行控制

其他保护方式：为每个文件加上密码

1. :文件系统实现

虚拟文件系统VFS：

通过定义一个清晰的VFS接口，以将文件系统的通用操作和具体实现分开。提供了在网络上唯一标识一个文件的机制。

目录实现：

线性列表

哈希表

磁盘空间的分配方法（与文件访问类型综合应用）：

连续分配：每个文件在磁盘上占有一组连续的块，支持顺序访问和直接访问。访问容易，困难的是为新文件找到空间，可能导致外部碎片。

链接分配：支持顺序访问。创建新文件很容易，没有外部碎片，不能有效支持直接访问，同时，指针也需要空间。变种为文件分配表，可支持直接访问。

索引分配：把链接分配的所有指针放到一个索引块中。支持直接访问，而且没有外部碎片问题。

空闲空间管理：

位图（位向量）:掌握计算

链表

1. ：大容量存储器的结构

磁盘的结构

磁盘调度：FCFS、SSTF（最短寻道时间优先算法，会导致饿死的问题）、SCAN算法（电梯算法）、C-SCAN调度、LOOK调度

磁盘格式化：物理格式化和逻辑格式化

交换空间管理：实现虚拟存储

1. ：I/O输入系统

轮询

中断：可屏蔽中断和非屏蔽中断

直接内存访问DMA：改善系统总体性能，但是也破坏了系统的安全和稳定。

阻塞与非阻塞I/O、异步I/O

缓冲：

缓冲区是用来保存两个设备之间或在设备和应用程序之间所传输数据的内存区域。

采用缓冲的三个理由：

处理数据流的生产者和消费者之间的速度差异。

协调传输数据大小不一致的设备

支持应用程序I/O的复制语义（用户内存的缓冲区被分到内核内存中）

假脱机：用来保存设备输出的缓冲区，这些设备不能接收交叉的数据流。

为了防止用户执行非法I/O，定义所有I/O指令为特权指令。