**作业3：**操作系统主要管理计算机哪些事务？总结其中涉及分配调度的事务，并简介其中算法；总结并介绍涉及虚拟的事务；总结及介绍涉及中间件的事务。

**1. 操作系统主要管理以下几个方面的计算机事务：**

（1）进程管理：操作系统负责创建、调度、执行和终止进程。它分配和管理进程的资源，还提供进程间通信的机制，允许不同进程之间的数据交换和同步。

（2）内存管理：操作系统负责管理计算机的内存资源。它负责将进程所需的内存分配给它们，并跟踪内存的使用情况。操作系统还通过虚拟内存技术将主存扩展为更大的虚拟内存空间，并进行页面交换和内存回收等操作。

（3）文件系统管理：操作系统负责管理计算机的文件系统，包括文件的创建、读取、写入和删除等操作。它提供了一个层次化的文件目录结构，并通过文件系统管理文件的访问和分配存储空间。

（4）设备管理：操作系统负责管理计算机的设备资源，包括输入输出设备、磁盘驱动器等。它通过设备驱动程序控制设备的访问和操作，并提供设备的抽象接口供应用程序使用。

（5）除上述事务外，操作系统还负责处理中断和异常，进行系统调度和资源分配，提供安全和保护机制，以及进行错误检测和恢复等工作。

**2. 总结其中涉及分配调度的事务，并简介其中算法**

在这些事务中，涉及到分配和调度的主要有进程管理和内存管理。

**2.1 进程管理算法**

（1）进程调度算法：用于选择下一个要执行的进程。常见的算法包括先来先服务（FCFS）、最短作业优先（SJF）、优先级调度（PR）、时间片轮转调度（RR）、多级反馈队列调度、实时调度等。

（2）死锁避免和恢复算法：用于检测、避免和解决进程之间的死锁情况。常见的算法包括银行家算法、资源分配图算法等。

（3）进程通信算法：用于实现不同进程间的通信和同步。常见的算法包括共享存储器系统、管道通信系统、消息传递系统、客户机-服务器系统、信号量机制、管程机制等。

**2.2 内存管理算法**

（1）静态分区分配算法：将主存分为固定大小的分区，每个分区可以分配给一个进程。常见的算法包括等分分区、最佳适应算法、最坏适应算法等。

（2）动态分区分配算法：根据进程的大小需求动态地分配内存。常见的算法包括首次适应算法、循环首次适应算法、最佳适应算法、最坏适应算法、快速适应算法、伙伴系统、哈希算法等。

（3）页面置换算法：当物理内存不足时，将不常用的内存页面置换到磁盘中。常见的算法包括最佳置换（OPT）、先进先出（FIFO）、最近最久未使用（LRU）、最少使用（LFU）算法等。

**3. 总结并介绍涉及虚拟的事务**

在计算机操作系统中，"虚拟"的主要目的是为了提供抽象，使得系统的使用和管理变得更加方便和高效。以下是一些关于虚拟化在操作系统中的主要应用：

（1）虚拟存储器：虚拟存储器是一种内存管理技术，其基本原理是局部性原理。它通过将当前需要的页面调入内存，而将其余部分保留在磁盘上，从而实现了逻辑上扩展内存的功能。虚拟存储器的主要目的是允许运行更大的程序和处理更多的数据，而不受物理内存的限制。

（2）虚拟处理器：多线程处理器可以在单个物理核心上提供虚拟处理器。每个线程在操作系统中作为一个独立的处理器出现，尽管它们可能共享底层的物理资源。这使得操作系统能够运行更多的并行进程。

（3）虚拟设备：操作系统中的设备驱动程序常常会提供虚拟设备的抽象。例如，硬盘驱动程序可能会提供一个虚拟文件系统，使得应用程序可以以文件和目录的方式来操作硬盘，而不需要知道硬盘的物理结构和操作方式。

（4）虚拟操作系统（虚拟机）：虚拟机（VM）是一种模拟计算机系统的软件实现，它能够在一个物理机器上运行多个虚拟机实例，每个实例都有自己的操作系统。虚拟机提供了一种隔离的环境，可以在这个环境中运行应用程序，这使得应用程序的测试、部署和管理变得更加方便。

**4. 总结及介绍涉及中间件的事务**

涉及中间件的事务包括缓存、磁盘缓存和虚拟 I/O 设备。中间件是指位于操作系统内核和应用程序之间的一层软件，用于实现不同应用程序之间的通信和数据传输。

（1）在涉及中间件的事务中，缓存起到了重要的作用。缓存是指将频繁访问的数据或计算结果存储在高速存储器中，以加快数据访问速度。中间件可以将一部分数据存储在缓存中，当应用程序需要访问这些数据时，可以直接从缓存中获取，避免了频繁的磁盘访问，提高了事务的执行效率。

（2）磁盘缓存是指将频繁访问的数据存储在磁盘上的缓存区域中。如果需要访问磁盘上的数据，中间件可以先检查磁盘缓存中是否已经存在这些数据，如果存在，则可以直接从缓存中获取，而不需要进行实际的磁盘读写操作，从而提高了事务的执行速度。

（3）虚拟 I/O 设备是一种虚拟化技术，它将物理设备模拟成为了虚拟设备，应用程序可以通过相应的操作系统接口或中间件接口来进行访问。中间件可以将虚拟 I/O 设备用作替代物理设备的访问接口，从而屏蔽了底层硬件的差异性，提供了更加统一和高层次的访问方式。这使得应用程序可以更加方便地进行数据读写操作，并且能够在不同的物理设备之间进行灵活切换，提高了事务处理的效率。