# 1 进程和线程的区别

* 进程：具有一定独立功能的程序在某个数据集上的一次动态执行的过程。是操作系统进行资源分配的基本单位。
* 线程：线程是进程中能划分出来的更小的执行单位，线程是CPU 调度和分派的基本单位
* 线程切换的开销要远远小于进程切换的开销
* 进程之间相互独立，同一进程内的线程共享程序的内存空间和资源。

# 2 死锁

集合中的每个进程都在等待由其他进程所占有的资源，从而陷入无限期僵持的局面

# 3 有了进程为什么还要线程

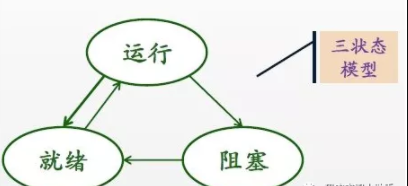
1.进程在同一时间只能干一件事情；

2.进程在执行的过程中如果阻塞，整个进程就会被挂起，即使进程中有些工作不依赖与等待的资源，仍然不会执行。

3.进程切换的代价要远远大于线程切换的代价

# 4 进程的状态切换

进程包括三种状态：就绪态、运行态和阻塞态。



* 就绪 —> 执行：对就绪状态的进程，当进程调度程序按一种选定的策略从中选中一个就绪进程，为之分配了处理机后，该进程便由就绪状态变为执行状态；
* 执行 —> 阻塞：正在执行的进程因发生某等待事件而无法执行，则进程由执行状态变为阻塞状态，如进程提出输入/输出请求而变成等待外部设备传输信息的状态，进程申请资源（主存空间或外部设备）得不到满足时变成等待资源状态，进程运行中出现了故障（程序出错或主存储器读写错等）变成等待干预状态等等；
* 阻塞 —> 就绪：处于阻塞状态的进程，在其等待的事件已经发生，如输入/输出完成，资源得到满足或错误处理完毕时，处于等待状态的进程并不马上转入执行状态，而是先转入就绪状态，然后再由系统进程调度程序在适当的时候将该进程转为执行状态；
* 执行 —> 就绪：正在执行的进程，因时间片用完而被暂停执行，或在采用抢先式优先级调度算法的系统中,当有更高优先级的进程要运行而被迫让出处理机时，该进程便由执行状态转变为就绪状态。

# 5 进程间的通信方式有哪些？

**管道**

* 它是半双工的，具有公共祖先的进程间使用 ；

**消息队列**

* 是消息的链接表，存放在内核中。
* 消息队列是面向记录的，其中的消息具有特定的格式以及特定的优先级；
* 消息队列可以实现消息的随机查询

**信号量**

* 信号量（semaphore）是一个计数器。用于实现进程间的互斥与同步，而不是用于存储进程间通信数据；

**共享内存**

* 共享内存（Shared Memory），指两个以上的进程共享一个给定的存储区；
* 共享内存是最快的一种 IPC，因为进程是直接对内存进行存取。

**套接字Socket**

在客户端进程和服务端进程之间创建一个套接字，对套接字进行读和写操作

6.线程间通信方式

**1.共享内存**

线程之间的共享变量存储在主内存中

**2.wait/notify等待通知**

将处于等待状态的线程将由其它线程发出通知后重新获取CPU资源，继续执行之前没有执行完的任务

**3.join**

基于等待通知机制，主线程调用子线程的join方法，需等待子线程执行完毕，才能执行主线程

**4.管道流**

通过管道来进行线程之间的数据传输

# 6 进程的调度算法

* **先来先服务调度算法 非抢占**

先来先服务调度算法是一种最简单的调度算法，也称为先进先出或严格排队方案。当每个进程就绪后，它加入就绪队列。当前正运行的进程停止执行，选择在就绪队列中存在时间最长的进程运行。该算法既可以用于作业调度，也可以用于进程调度。先来先去服务比较适合于常作业（进程），而不利于段作业（进程）。

* **短作业优先调度算法 非抢占**

短作业优先调度算法是指对短作业优先调度的算法，从后备队列中选择一个或若干个估计运行时间最短的作业，将它们调入内存运行。 短作业优先调度算法是一个非抢占策略，他的原则是下一次选择预计处理时间最短的进程，因此短进程将会越过长作业，跳至队列头。

* **高响应比优先调度 非抢占**

根据“响应比=（进程执行时间+进程等待时间）/ 进程执行时间”这个公式得到的响应比来进行调度。高响应比优先算法在等待时间相同的情况下，作业执行的时间越短，响应比越高，满足段任务优先

* **优先级调度算法 非抢占**

优先级调度算法每次从后备作业队列中选择优先级最髙的一个或几个作业，将它们调入内存，分配必要的资源，创建进程并放入就绪队列。在进程调度中，优先级调度算法每次从就绪队列中选择优先级最高的进程，将处理机分配给它，使之投入运行。

* **最短剩余时间优先调度 抢占**

最短剩余时间是针对最短进程优先增加了抢占机制的版本。在这种情况下，进程调度总是选择预期剩余时间最短的进程。当一个进程加入到就绪队列时，他可能比当前运行的进程具有更短的剩余时间，因此只要新进程就绪，调度程序就能可能抢占当前正在运行的进程。像最短进程优先一样，调度程序正在执行选择函数是必须有关于处理时间的估计，并且存在长进程饥饿的危险。

* **时间片轮转调度 抢占**

时间片轮转调度算法主要适用于分时系统。在这种算法中，系统将所有就绪进程按到达时间的先后次序排成一个队列，进程调度程序总是选择就绪队列中第一个进程执行，即先来先服务的原则，但仅能运行一个时间片。

* **多级反馈队列调度 抢占**

将时间片轮转与优先级调度相结合，把进程按优先级分成不同的队列，先按优先级调度，优先级相同的，按时间片轮转。

# 7 死锁的必要条件

* 互斥条件：一段时间内某资源仅被一个进程所占用。
* 请求和保持条件：当进程因请求资源而阻塞时，对已获得的资源保持不放。
* 不剥夺条件：不能剥夺进程所占有的资源。
* 循环等待条件：各个进程都在等待由其他进程所占有的资源。

# 8 怎么预防死锁

互斥条件一般无法破坏

* 破坏请求保持条件：要么一次性分配所有资源，要么一个资源也不分配。
* 破坏不可剥夺条件：当某进程获得了部分资源，但得不到其它资源，则释放已占有的资源；
* 破坏循环等待条件：通为每个资源编号，当一个进程占有编号为i的资源时，那么它下一次申请资源只能申请编号大于i的资源

# 9 什么是用户态和内核态

CPU的状态

* 内核态：

处于内核态的 CPU 可以访问任意的数据

可以访问外围设备，比如网卡、硬盘等

处于内核态的 CPU 不会发生抢占情况

* 用户态：

处于用户态的 CPU 只能受限的访问内存

不允许访问外围设备

用户态下的 CPU 可以被抢占

## 10.同一个进程中的线程默认资源全共享吗

不是，线程也有自己的私有属性。如局部变量和线程id，这些不会被共享