## 进程线程相关

1. 什么是进程？

程序运行的一个实例。包含程序、输入、输出和状态

1. 什么时候创建进程？
   1. 系统初始化
   2. 批处理作业初始化
   3. 进程调用创建进程的系统调用
   4. 用户请求创建进程
2. 进程退出方式？
   1. 正常退出
   2. 出错退出（如g++ t.c文件不存在）
   3. 严重错误（程序本身出错）
   4. 被其他进程杀死
3. 进程间通信的方式
   1. 低级通信
      1. 信号
   2. 高级通信
      1. 共享内存：建立公共存储区
      2. 共享文件：匿名管道、命名管道
      3. 消息传递：点对点、信箱
   3. 信号量和socket
4. 低级通信和高级通信区别：

低级通信用来传递控制信息，高级通信传递大量数据信息。

1. 信号和信号量的概念和区别：
   1. 信号是一种处理异步事件的方式。信号是复杂的通信方式，用于通知接受事件某种事件发生，可以发送给别的进程，也可以发送给自身。
   2. 信号量是一种处理互斥同步的机制。用于多线程，保证多线程正确合理的处理公共资源
2. 高级通信方式的特点：
   1. 共享内存：建立公共存储区。
      1. 优点：快捷有效，使用系统提供同步机制（信号灯）；
      2. 缺点：不关心写者和读者，存在安全隐患
   2. 共享文件：管道（匿名管道和命名管道），基于缓冲流机制
      1. 优点：使用系统调用read和write操作进行读写
      2. 缺点：单向通信，双向需要两个管道
   3. 消息传递：直接（点对点）、间接（信箱）
      1. 优点：信箱容易实现双向通讯
      2. 缺点：点对点必须指定接收方进程id；信箱必须有一个通讯双方共享的消息队列
3. 匿名管道和命名管道区别：
   1. 匿名管道：用于父子进程间通讯，管道将一个进程的地址空间复制到另一个进程的地址空间
   2. 命名管道：用于任意进程间通讯，通过管道名获取管道
4. 进程是怎么实现的？

操作系统维护一个进程表，每个进程是一个进程表项，该表项包含了进程运行的重要信息，进程管理，存储管理和文件管理三列。

* 1. 当中断发生时，硬件将程序计数器和状态字压入堆栈，跳转到中断服务例程，
  2. 中断服务例程保存当前进程信息到进程表项，从堆栈中删除由硬件保存的那部分信息，并将堆栈指针设置为由进程处理程序所使用的临时堆栈。这部分由汇编完成。
  3. 例程结束后，调用c过程处理剩下的工作。
  4. 调用调度程序决定运行哪个进程。
  5. 调用汇编代码装入寄存器值以及内存映射启动该进程。

1. 进程和线程区别
   1. 进程是资源分配最小单位，线程是cpu执行最小单位
   2. 进程独享地址空间，线程共享地址空间和大部分数据
   3. 线程共享数据，通讯方式方便，需要同步互斥机制
   4. 进程比线程健壮，一个线程死亡导致其所在进程死亡；进程出错不会影响其他进程
   5. 线程有自己的程序入口、执行序列和程序出口，但不能独立运行
2. 为什么需要线程？
   1. 程序设计模型变的简单：共享地址空间和可用数据
   2. 线程比进程过更轻量级
   3. 性能：cpu密集型不能获得性能上提升，io密集型可以
   4. 多线程真正实现了并行
3. 什么是线程安全？

如果多线程执行结果可预期，并且结果与单线程结果一致，就称线程安全

1. 线程的基本概念、状态和状态转换关系

线程是轻量级进程，进程中的一个运行实体。引入线程的目的为了提高系统效率，减少处理机空转时间和调度时间。

基本状态：阻塞、就绪、运行

基本操作：派生、激活、阻塞、调度、结束

1. 线程的典型应用：
   1. 服务器文件管理和通信
   2. 前后台处理
   3. 异步处理
2. 多线程互斥与同步的区别，什么时候分别使用？
   1. 同步是线程间的一种制约关系，当一个线程依赖另一个线程的消息，当他没有得到该消息时等待，直到该消息到达唤醒它
   2. 线程互斥多线程共享数据，对该数据的访问具有排他性，任何时刻只有一个线程被允许访问该数据。互斥是一种特殊的同步
3. 多线程同步与互斥有几种不同的实现方法？

分为用户模式和内核模式，内核模式利用系统内核对象的单一性保证同步，用户模式使用原子操作和临界区保证同步和互斥；内核模式使用事件、互斥量、信号量

* 1. 临界区：保证任何时刻只有一个线程操作共享变量，除非该线程退出，否则其他线程被挂起直到该进程推出，然后抢占。

缺点：只能用来同步本进程的线程

* 1. 事件：使用通知操作同步线程，可用于不同进程内的线程同步
  2. 互斥量：类似于临界区，只有拥有互斥对象的线程可以操作共享数据，保证任何时刻只有一个线程操作共享数据。与临界区不同的是，互斥量可用与不同进程间的线程同步
  3. 信号量：区别于其他三种方式，允许多个线程同时访问共享资源，但是限制可同时访问共享资源的线程数

1. 什么是可重入类的和线程安全的类？

如果一个类的所有成员函数可以被不同的线程调用而互不影响——即使针对同一对象，则称该类线程安全的。

如果类的不同实例被不同线程调用而互不影响，称该类是可重入的。

1. 何时出现进程调度？
   1. 父进程创建子进程时决定谁先运行
   2. 进程退出时
   3. 进程阻塞时
   4. IO中断发生时
2. 调度算法的目标？
   1. 所有系统：
      1. 公平：相似的进程应该得到相似的服务
      2. 平衡：保证任意时刻系统是忙碌的
   2. 批处理系统：
      1. 吞吐量：每小时做大作业数
      2. 周转周期：从提交到终止的最短时间
      3. CPU利用率：保持CPU始终忙碌
   3. 交互式系统：
      1. 响应时间：快速响应请求
      2. 均衡性：满足用户期望
   4. 实时系统：
      1. 满足截止时间：避免丢失数据
      2. 可预测性：在多媒体系统中避免品质下降
3. 调度算法有哪些？

分为三种环境下：批处理、交互式、实时

* 1. 批处理系统：
     1. 先来先服务
     2. 最短作业优先
     3. 最短剩余时间优先（最短作业优先的抢占式版本）
  2. 交互式系统
     1. 轮转调度：将时间片设置短，造成进程切换开销浪费；将时间片设置长，响应得不到保障。通常20ms——50ms比较合理
     2. 优先级调度：优先级可以静态赋予和动态赋予。可能会造成饿死现象。
     3. 多级队列：每次运行完它的时间片增加
     4. 最短进程优先
     5. 保证调度：系统必须跟踪进程自创建以来获得的CPU时间，和1/n对比
     6. 彩票调度
     7. 公平分享调度：调度是考虑及进程所有者因素，每个用户分配到平等的PUC时间
  3. 实时系统

以上所有调度算法都可以用于实时系统

1. 交互式系统与实时系统的区别?

实时系统只运行那些用来推进现有应用的程序，而交互式系统是通用的，他可以运行任意的非协作甚至恶意程序