操作系统概述：

硬件与用户之间程序集合（核心，控制，接口程序） 工作方式：中断驱动

系统保护：Dual-Mode（用户，内核态）、CPU（进程时间控制）、内存（程序地址控制）、IO保护（全privilege）

特权指令：可能伤害硬件的指令，在内核态执行

引导程序：ROM,EPROM 内存：RAM

发展：单任务~批处理 ~多道程序~分时系统 CPU利用率：非空时间占比

服务：系统内部具有的能力 功能：对外提供的服务 接口：SCI，API

特征：**并发**（Concurrence），**共享**（Sharing），**虚拟**（Virtual），**异步**（Asynchronous）

系统结构：简单~分层~微内核~模块化

进程：

进程：按照顺序结构执行的程序（程序代码，程序计数器PC（执行语句），指令寄存器（临时变量执行状态），堆栈+数据段（全局变量））

特征：**动态性**（Dynamic），**独立性**（Independency），**并发性**（Concurrence），**结构化**（Structure）

进程队列：就绪（等待执行）~设备（等待IO）~其它（存在于内存中）

调度器：Long-term（载入）~Short-term（执行）~Medium-term（交换） 上下文切换：保存PCB，CPU，内存信息（区分进程状态）

子进程：创建（fork()，复制进程代码与状态），更改（exec()，通过PID识别更改子进程功能）

进程通信：RPC（远程进程函数调用 stub-stub），RMI（远程对象方法调用 stub-skeleton）

线程：

进程中的**控制流**，CPU执行的最小单位，多个线程**共享进程地址空间**（线程ID，程序计数器，寄存器组，（独享）堆栈）

线程分类：用户线程（用户线程库管理，进程阻塞），内核线程（内核管理）

多线程模型：用户-内核线程（多对一~多对多~一对一~Two level（多对多，一对一处理紧急线程））

线程池：预先创建内核线程放到线程池中，进程需要时分配（效率高，对线程总数可控）

CPU调度：

CPU空闲时从就绪队列中选取进程（抢占/非抢占Preemptive）

策略评价：CPU利用率，吞吐量（系统），周转时间，等待时间，响应时间（用户）

调度策略：FCFS先到先服务（取队头 护航效应convey effect），SJF最短最优先（剩余CPU数最少 平均周转时间最短），Priority优先级（优先级老化防饥饿），RR轮询（使大多数进程在一个时间片内完成 平均响应时间最短）

实时调度：资源预留确保进程按时完成，无法用老化防止饥饿

算法评价方法：决策（人为决定），队列，模拟，实现

进程同步：

规定进程执行顺序，确保共享数据一致性

临界区：更改共享数据代码段 临界资源：共享数据 临界区协议：进入，临界区，退出

解决方法：**互斥**，**进展性**，**有限等待**（三个条件需**同时满足**，证明满足（反证））

**临界区中进程不一定运行**（可能为IO）**，等待进入临界区进程不一定处于等待状态**

TAS：取出标志位并置为true swap：交换key与lock（测试key开锁）

互斥信号量（临界资源mutex），同步信号量（mutex+同步关系counter）

死锁解决：破坏占有等待（分配全部/全不分配），破坏不可抢占（优先级），破坏循环等待（递增申请资源）

Tips：

1. 批处理，多道程序，分时系统的区别与所解决的问题
2. 分层与模块化的操作系统结构异同（同：实现模块化设计，便于扩展与维护 异：分层需逐层调用，模块之间可以相互调用灵活性高）
3. Difference between program & process & thread（程序是永久保存的静态实体，进程是按照顺序执行的程序，线程是进程控制流是CPU执行的最小单位）
4. Five state translation diagram of a process
5. Fork and exec operation（fork复制代码段与进程执行状态，exec修改子进程的代码段）
6. Difference between user and kernel thread（存在空间不同，管理者不同）
7. 信号与中断异同（异步通信，处理程序，返回，可屏蔽 优先级，用户/内核，及时/延时响应）
8. Preemptive & Non- Preemptive example（多用户,抢占 单一操作资源独占，非抢占）