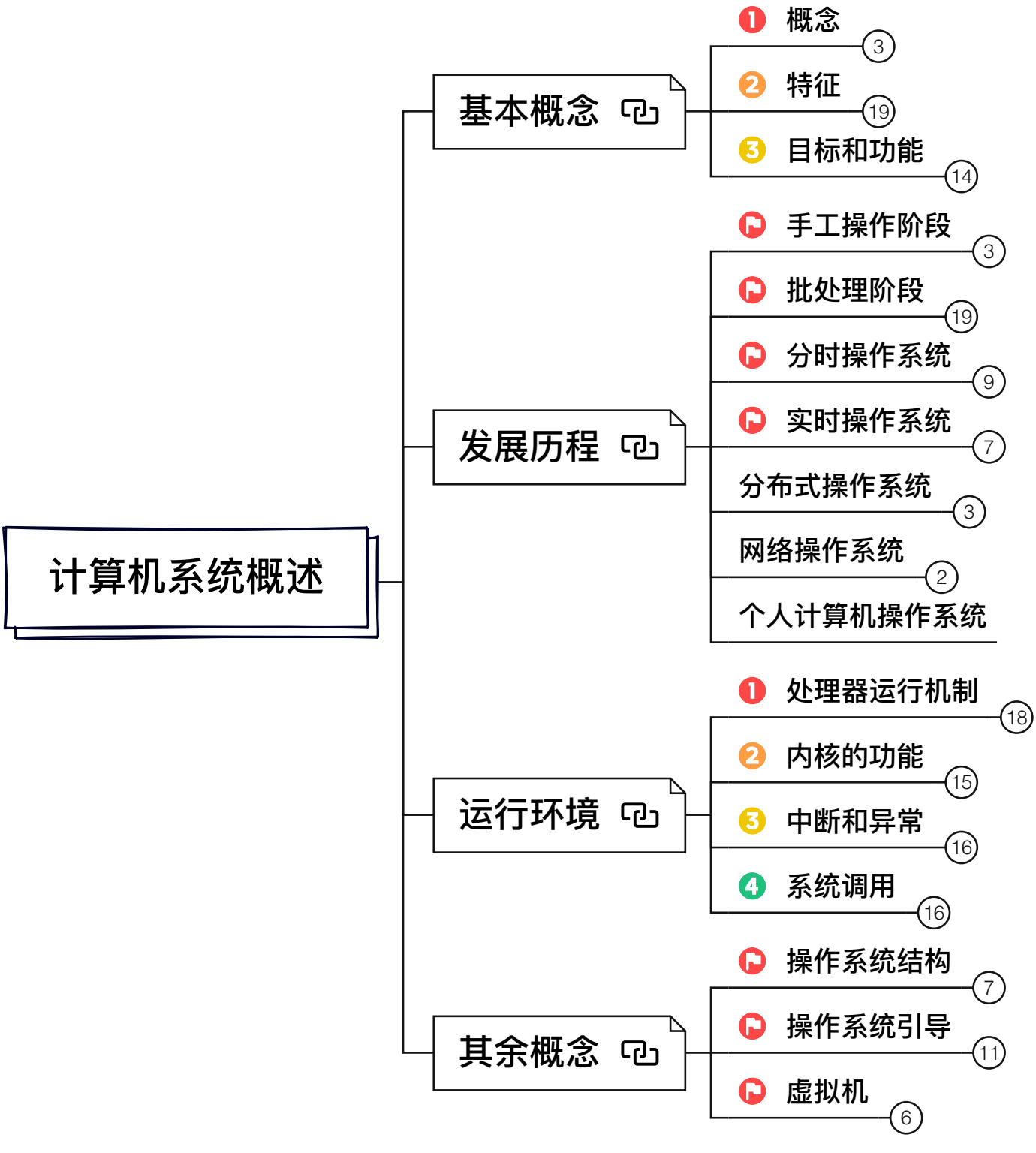
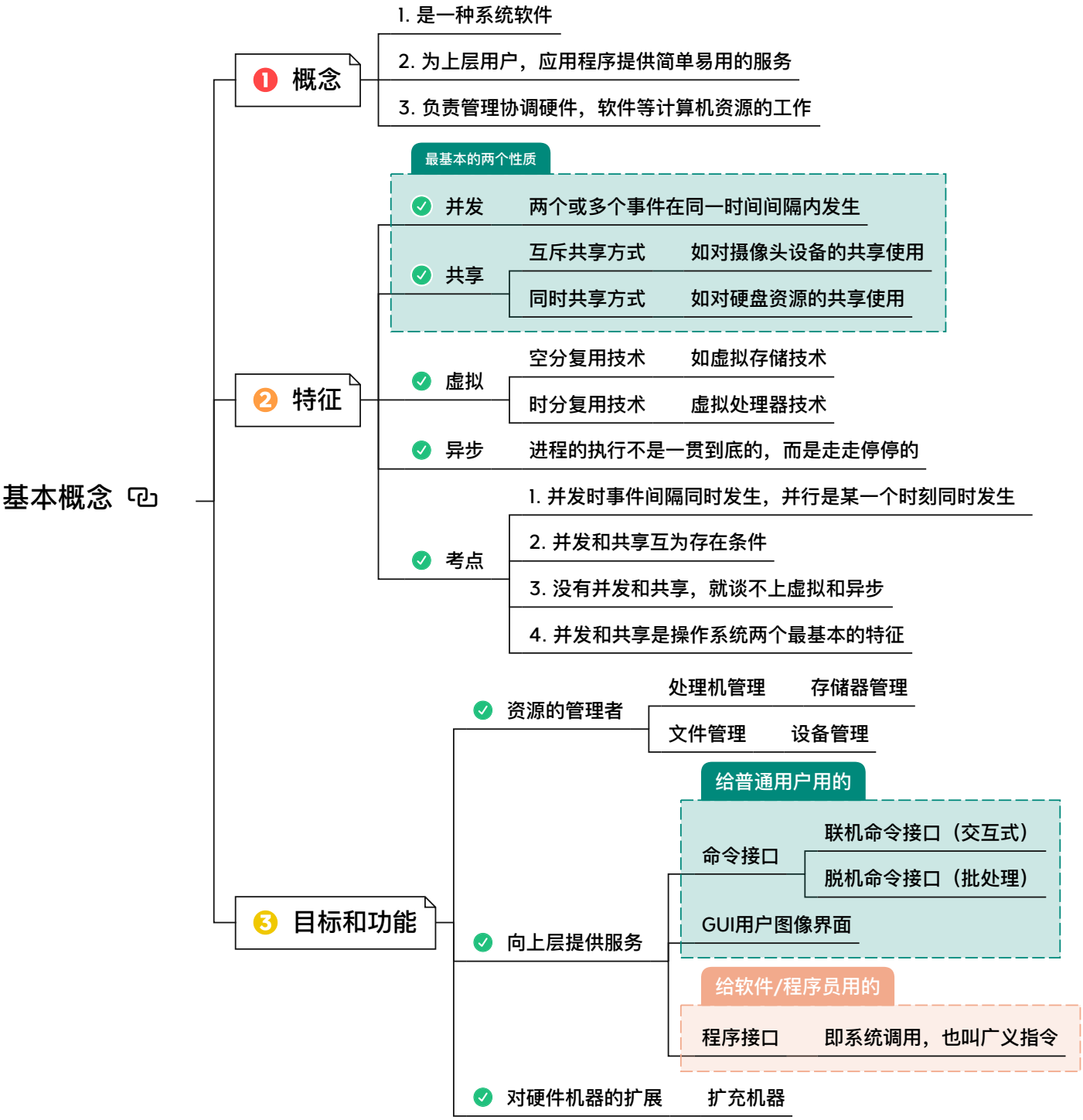
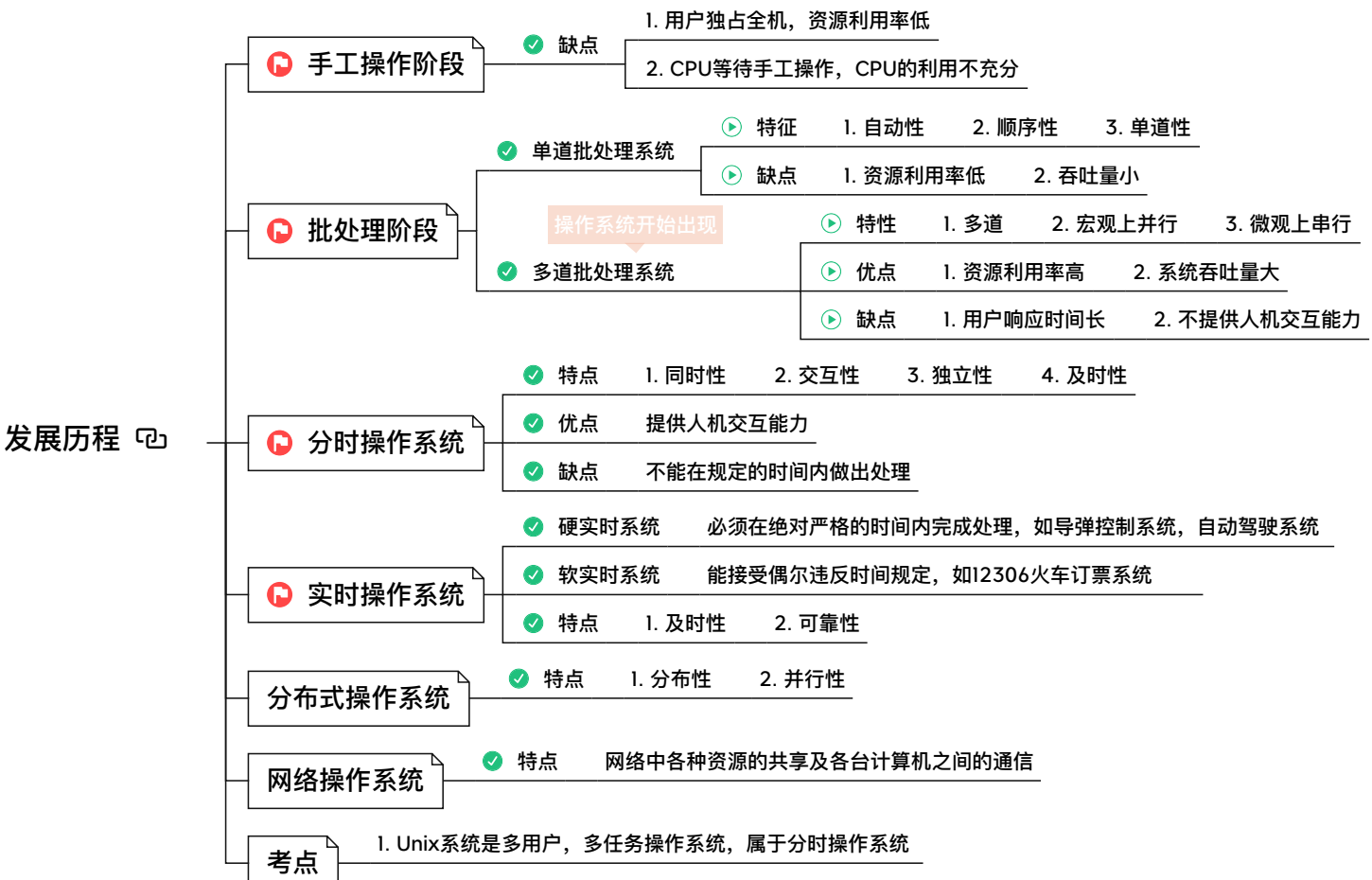
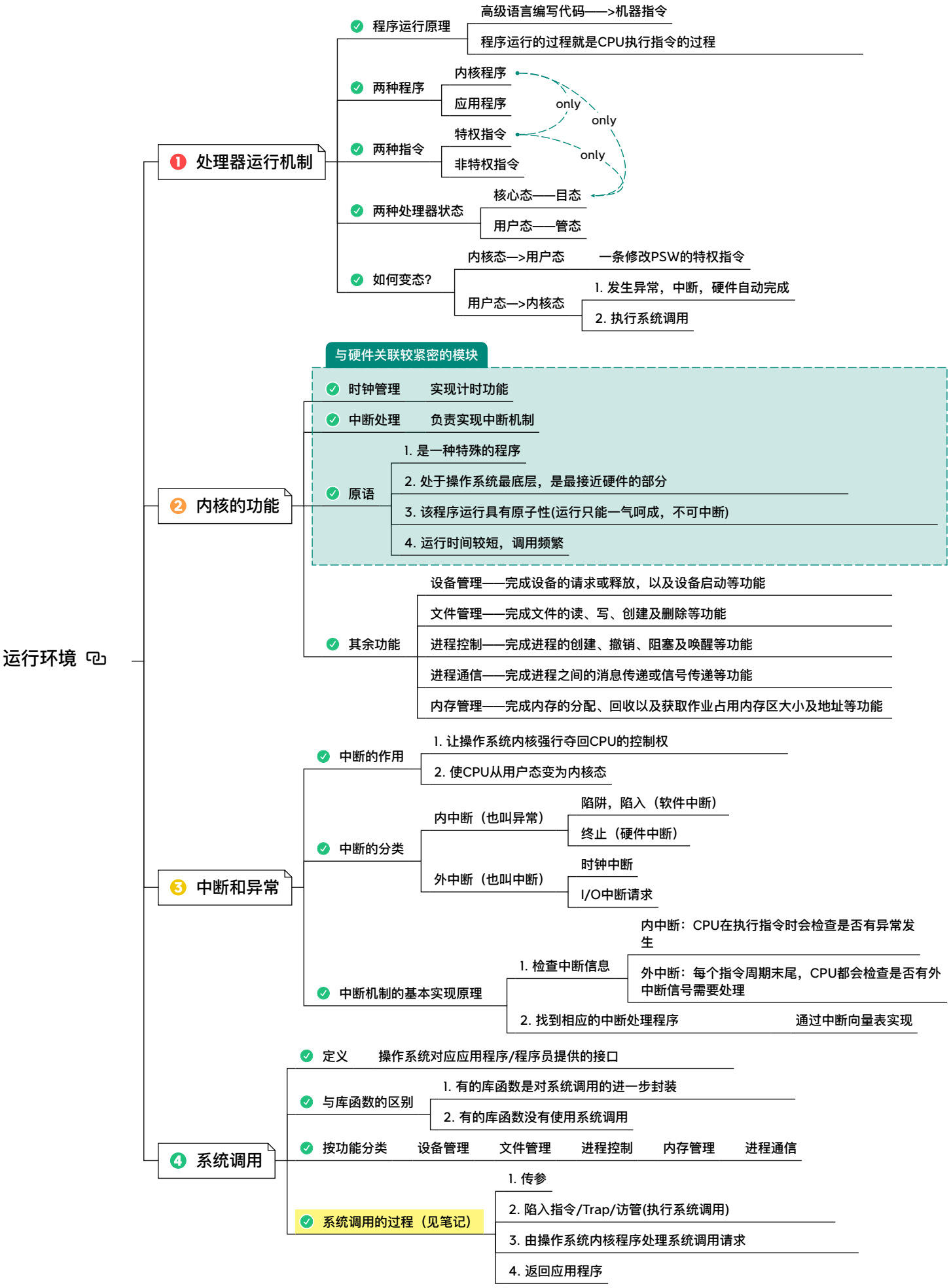
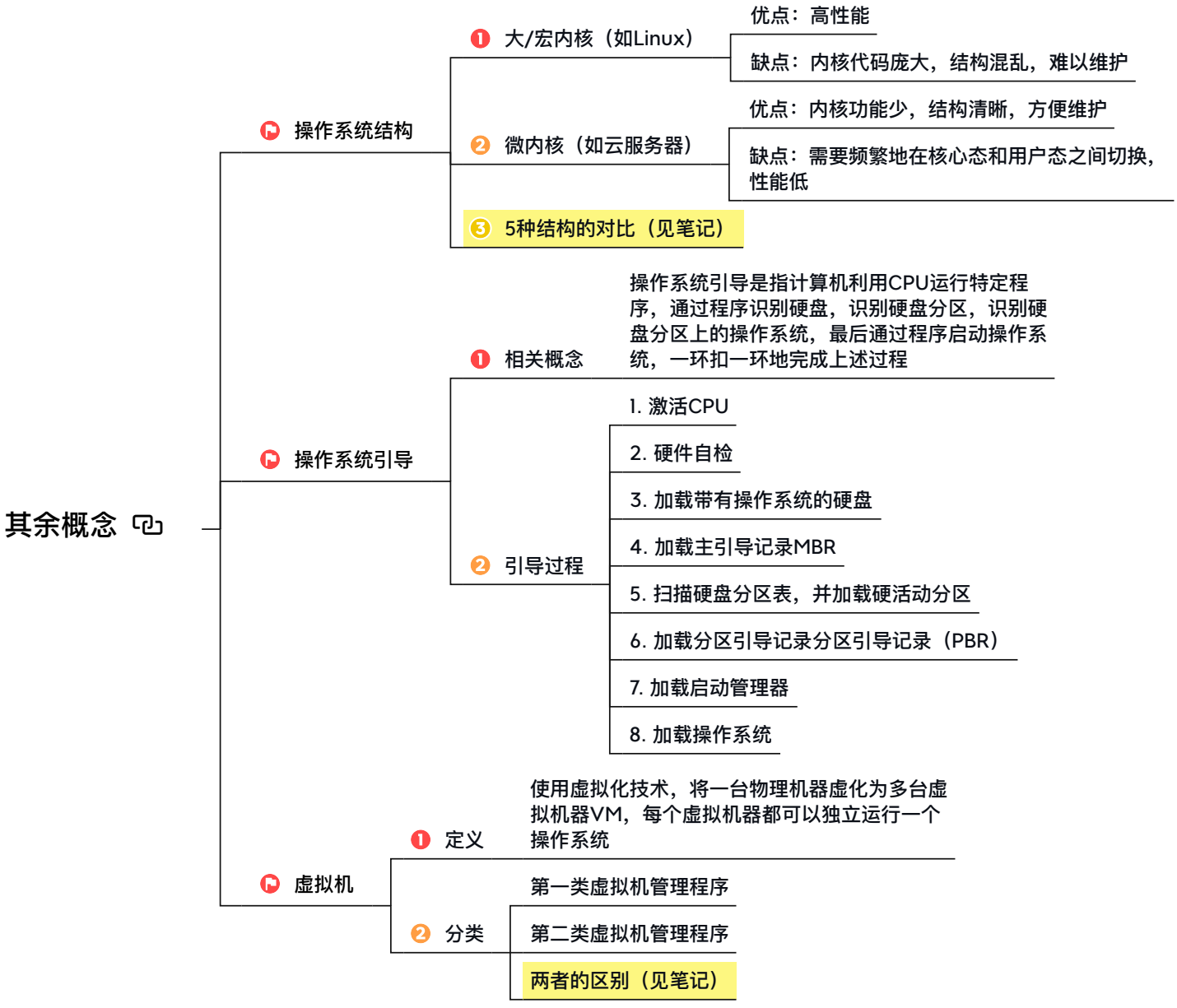

计算机系统概述
基本概念
1 概念
3
2 特征
19
3 目标和功能
14
发展历程
手工操作阶段
3
批处理阶段
19
分时操作系统
9
实时操作系统
7
分布式操作系统
3
网络操作系统
2
个人计算机操作系统
运行环境
1 处理器运行机制
18
2 内核的功能
15
3 中断和异常
16
4 系统调用
16
其余概念
操作系统结构
7
操作系统引导
11
虚拟机
6


基本概念
1 概念
1. 是一种系统软件
2. 为上层用户，应用程序提供简单易用的服务
3. 负责管理协调硬件，软件等计算机资源的工作
2 特征
最基本的两个性质
并发
两个或多个事件在同一时间间隔内发生
共享
互斥共享方式
如对摄像头设备的共享使用
同时共享方式
如对硬盘资源的共享使用
虚拟
空分复用技术
如虚拟存储技术
时分复用技术
虚拟处理器技术
异步
进程的执行不是一贯到底的，而是走走停停的
考点
1. 并发时事件间隔同时发生，并行是某一个时刻同时发生
2. 并发和共享互为存在条件
3. 没有并发和共享，就谈不上虚拟和异步
4. 并发和共享是操作系统两个最基本的特征
3 目标和功能
资源的管理者
处理机管理
存储器管理
文件管理
设备管理
向上层提供服务
给普通用户用的
命令接口
联机命令接口（交互式）
脱机命令接口（批处理）
GUI用户图像界面
给软件/程序员用的
程序接口
即系统调用，也叫广义指令
对硬件机器的扩展
扩充机器


发展历程
手工操作阶段
缺点
1. 用户独占全机，资源利用率低
2. CPU等待手工操作，CPU的利用不充分
批处理阶段
单道批处理系统
特征 1. 自动性 2. 顺序性 3. 单道性
缺点 1. 资源利用率低 2. 吞吐量小
操作系统开始出现
多道批处理系统
特性 1. 多道 2. 宏观上并行 3. 微观上串行
优点 1. 资源利用率高 2. 系统吞吐量大
缺点 1. 用户响应时间长 2. 不提供人机交互能力
分时操作系统
特点 1. 同时性 2. 交互性 3. 独立性 4. 及时性
优点 提供人机交互能力
缺点 不能在规定的时间内做出处理
实时操作系统
硬实时系统 必须在绝对严格的时间内完成处理，如导弹控制系统，自动驾驶系统
软实时系统 能接受偶尔违反时间规定，如12306火车订票系统
特点 1. 及时性 2. 可靠性
分布式操作系统
特点 1. 分布性 2. 并行性
网络操作系统
特点 网络中各种资源的共享及各台计算机之间的通信
考点
1. Unix系统是多用户，多任务操作系统，属于分时操作系统


运行环境
1 处理器运行机制
程序运行原理
高级语言编写代码——>机器指令
程序运行的过程就是CPU执行指令的过程
两种程序
内核程序
应用程序
only
only
两种指令
特权指令
非特权指令
only
两种处理器状态
核心态——目态
用户态——管态
如何变态?
内核态—>用户态
一条修改PSW的特权指令
用户态—>内核态
1. 发生异常，中断，硬件自动完成
2. 执行系统调用
2 内核的功能
与硬件关联较紧密的模块
时钟管理
实现计时功能
中断处理
负责实现中断机制
原语
1. 是一种特殊的程序
2. 处于操作系统最底层，是最接近硬件的部分
3. 该程序运行具有原子性(运行只能一气呵成，不可中断)
4. 运行时间较短，调用频繁
其余功能
设备管理——完成设备的请求或释放，以及设备启动等功能
文件管理——完成文件的读、写、创建及删除等功能
进程控制——完成进程的创建、撤销、阻塞及唤醒等功能
进程通信——完成进程之间的消息传递或信号传递等功能
内存管理——完成内存的分配、回收以及获取作业占用内存区大小及地址等功能
3 中断和异常
中断的作用
1. 让操作系统内核强行夺回CPU的控制权
2. 使CPU从用户态变为内核态
中断的分类
内中断（也叫异常）
陷阱，陷入（软件中断）
终止（硬件中断）
外中断（也叫中断）
时钟中断
I/O中断请求
中断机制的基本实现原理
内中断：CPU在执行指令时会检查是否有异常发生
1. 检查中断信息
外中断：每个指令周期末尾，CPU都会检查是否有外中断信号需要处理
2. 找到相应的中断处理程序
通过中断向量表实现
4 系统调用
定义
操作系统对应应用程序/程序员提供的接口
与库函数的区别
1. 有的库函数是对系统调用的进一步封装
2. 有的库函数没有使用系统调用
按功能分类
设备管理
文件管理
进程控制
内存管理
进程通信
系统调用的过程（见笔记）
1. 传参
2. 陷入指令/Trap/访管(执行系统调用)
3. 由操作系统内核程序处理系统调用请求
4. 返回应用程序


其余概念
操作系统结构
1 大/宏内核（如Linux）
优点：高性能
缺点：内核代码庞大，结构混乱，难以维护
2 微内核（如云服务器）
优点：内核功能少，结构清晰，方便维护
缺点：需要频繁地在核心态和用户态之间切换，性能低
3 5种结构的对比（见笔记）
操作系统引导
1 相关概念
操作系统引导是指计算机利用CPU运行特定程序，通过程序识别硬盘，识别硬盘分区，识别硬盘分区上的操作系统，最后通过程序启动操作系统，一环扣一环地完成上述过程
2 引导过程
1. 激活CPU
2. 硬件自检
3. 加载带有操作系统的硬盘
4. 加载主引导记录MBR
5. 扫描硬盘分区表，并加载硬活动分区
6. 加载分区引导记录分区引导记录（PBR）
7. 加载启动管理器
8. 加载操作系统
虚拟机
1 定义
使用虚拟化技术，将一台物理机器虚化为多台虚拟机器VM，每个虚拟机器都可以独立运行一个操作系统
2 分类
第一类虚拟机管理程序
第二类虚拟机管理程序
两者的区别（见笔记）

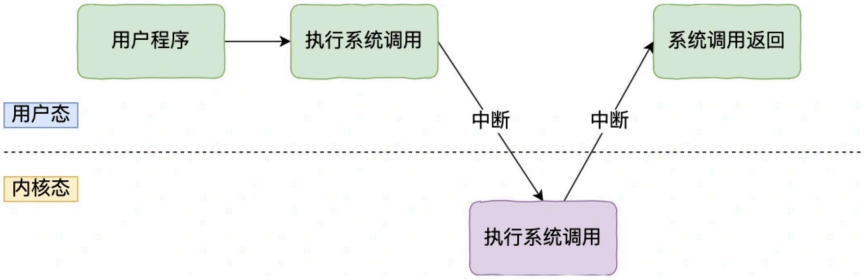
5种系统结构对比

	特性、思想	优点	缺点
分层结构	内核分多层，每层可单向调用更低一层提供的接口	1. 便于调试和验证，自底向上逐层调试验证 2. 易扩充和易维护，各层之间调用接口清晰固定	1. 仅可调用相邻低层，难以合理定义各层的边界 2. 效率低，不可跨层调用，系统调用执行时间长
模块化	将内核划分为多个模块，各模块之间相互协作。 内核 = 主模块+可加载内核模块 主模块：只负责核心功能，如进程调度、内存管理 可加载内核模块：可以动态加载新模块到内核，而无需重新编译整个内核	1. 模块间逻辑清晰易于维护，确定模块间接口后即可多模块同时开发 2. 支持动态加载新的内核模块（如：安装设备驱动程序、安装新的文件系统模块到内核），增强OS适应性 3. 任何模块都可以直接调用其他模块，无需采用消息传递进行通信，效率高	1. 模块间的接口定义未必合理、实用 2. 模块间相互依赖，更难调试和验证
宏内核（大内核）	所有的系统功能都放在内核里（大内核结构的OS通常也采用了"模块化"的设计思想）	1. 性能高，内核内部各种功能都可以直接相互调用	1. 内核庞大功能复杂，难以维护 2. 大内核中某个功能模块出错，就可能导致整个系统崩溃
微内核	只把中断、原语、进程通信等最核心的功能放入内核。进程管理、文件管理、设备管理等功能以用户进程的形式运行在用户态	1. 内核小功能少、易于维护，内核可靠性高 2. 内核外的某个功能模块出错不会导致整个系统崩溃	1. 性能低，需要频繁的切换 用户态/核心态。 用户态下的各功能模块不可以直接相互调用，只能通过内核的"消息传递"来间接通信 2. 用户态下的各功能模块不可以直接相互调用，只能通过内核的"消息传递"来间接通信
外核（exokernel）	内核负责进程调度、进程通信等功能，外核负责为用户进程分配未经抽象的硬件资源，且由外核负责保证资源使用安全	1. 外核可直接给用户进程分配"不虚拟、不抽象"的硬件资源，使用户进程可以更灵活的使用硬件资源 2. 减少了虚拟硬件资源的"映射层"，提升效率	1. 降低了系统的一致性 2. 使系统变得更复杂

两类VMM的对比

	第一类VMM	第二类VMM
对物理资源的控制权	直接运行在硬件之上，能直接控制和分配物理资源	运行在Host OS之上，依赖于Host OS为其分配物理资源
资源分配方式	在安装Guest OS时，VMM要在原本的硬盘上自行分配存储空间，类似于"外核"的分配方式，分配未经抽象的物理硬件	GuestOS 拥有自己的虚拟磁盘，该盘实际上是Host OS 文件系统中的一个大文件。GuestOS分配到的内存是虚拟内存
性能	性能更好	性能更差，需要HostOS作为"中介"
可支持的虚拟机数量	更多，不需要和 Host OS 竞争资源，相同的硬件资源可以支持更多的虚拟机	更少，Host OS 本身需要使用物理资源，Host OS 上运行的其他进程也需要物理资源
虚拟机的可迁移性	更差	更好，只需导出虚拟机镜像文件即可迁移到另一台 HostOS 上，商业化应用更广泛
运行模式	第一类VMM运行在最高特权级（Ring 0），可以执行最高特权的指令。	第二类VMM部分运行在用户态、部分运行在内核态。GuestOS 发出的系统调用会被 VMM 截获，并转化为 VMM 对 HostOS 的系统调用

系统调用的过程



特权指令

①清内存 ②置时钟 ③分配系统资源 ④修改虚存的段页或页表

⑤修改用户的访问权限 ⑥系统调用(广义指令) ⑦输入/输出(涉及到中断),I/O指令

⑧关中断指令

非特权指令

访管指令,trap指令,跳转指令、压栈指令

在用户态发生的事件

①命令解释程序 ②系统调用 ③外部中断 ④缺页

在核心态发生的事件

①缺页处理程序 ②进程调度程序 ③时钟中断处理程序

④进程切换

常见的中断

①访管中断,由访管指令引起,发生在用户态(管态)