1. 多道程序设计技术允许将多个程序一同加载到计算机的内存中，并使它们能够交替执行。这种方式下，多个程序会共享计算机的硬件和软件资源。如果某个程序需要等待输入/输出操作，CPU会立刻切换到另一个程序执行，以提高效率。在分时系统中，CPU的时间被划分成若干小的片段，这些时间片轮流分配给各个程序，从而实现了好像多个程序似乎是在同时执行一样。当程序的时间片用尽时，会触发时钟中断，系统会暂停当前程序的执行，保存其状态，然后根据调度策略选择另一程序继续执行。批处理系统和分时系统的主要差异在于它们的目标和适用场景有所不同：批处理系统旨在优化系统的整体资源利用率和作业处理量，适合运行大型、预先经过测试的作业；而分时系统则注重于提供公平的资源访问，更适合用于开发和调试较小的作业，并且由于它支持多用户作业，调度开销相对较小，资源使用上也显示出不同的特点。
2. 提高处理效率、需求增加、技术进步、交互性需求、公平性和资源利用率的平衡
3. - 陷阱（Trap）是一种由软件引起的中断，通常是程序中的异常情况或者是特定的系统调用指令导致的。例如，当程序执行到一个非法操作（如除以零）或尝试执行一个特权操作（如直接访问硬件设备）时，就会产生陷阱。陷阱也可以由程序显式请求操作系统服务时产生，此时它作为系统调用的一种实现机制。当产生陷阱时，执行流程会从用户模式切换到内核模式，以运行操作系统代码处理该陷阱。
   - 陷阱和中断都是转移控制流到操作系统的机制，但它们的主要区别在于触发源和预定性：1、陷阱通常由程序执行中的事件（如错误或系统调用）触发，是软件产生的；而中断主要由硬件设备产生的事件触发，如键盘输入、鼠标移动或硬件计时器超时等。2、陷阱往往是可预期的，因为它们是由程序代码显式触发的，或者由程序错误引发的；中断通常是异步的，它们可能在程序执行的任何时刻发生，与当前执行的指令无关。
   - 系统调用（System Call）是程序向操作系统请求服务的机制，如文件操作、进程控制或网络通信等。当应用程序需要操作系统提供的服务时，会执行一个特定的系统调用指令，这通常会引发一个陷阱，导致处理器从用户模式切换到内核模式。在内核模式下，操作系统会执行相应的服务并返回结果。系统调用为应用程序提供了一种安全的方式来利用操作系统的功能，而不需要直接访问硬件资源。
   - 系统调用、陷阱和中断共同构成了软硬件之间交互的基础，使得操作系统能够控制资源访问，同时保证了系统的稳定性和安全性。
4. - 每个系统架构都有自己独特的一套指令集，不同架构的设备无法执行相同的程序，反之亦然。同时，不同架构的硬件基础不同，由于这些差异，构建完全可移植的操作系统并不可能。
   - 一个高度可移植的操作系统包含两个层次：机器相关层和机器无关层。机器相关层会考虑不同硬件的不同特性，根据不同的架构提供统一的接口，供机器无关层使用。而机器无关层仅仅需要调用统一的接口即可。
5. 一个设计实例是吞吐量和响应时间之间的关系。响应时间和吞吐量之间的矛盾在于：为了减少响应时间，系统需要更频繁地切换任务，这会增加上下文切换的开销，从而可能降低吞吐量；反之，为了提高吞吐量，系统可能会增加任务的执行时间，减少任务切换，但这又会导致响应时间变长。
6.

打印机1						
打印机2		A	A		A	A
输入			B	B(30ms)		
处理器	A	B		A	B	B
时间间隔	50	50	50	50	50	50

- 有空闲，在 B 程序第一次计算后的 50ms，因为程序 A 在输出，B 在等待输入。
- 程序 A 没有等待，程序 B 在输入结束时 A 在占用处理器，因此发生等待。