作业 1

1. 什么是多道程序设计？多道程序设计与分时系统的区别是什么？

答：

- 指允许多个程序同时进入内存并运行，即同时把多个程序放入内存中（前提是内存放得下），并允许它们交替在CPU中运行，它们共享系统中的各种软硬件资源。当一道程序因I/O请求而暂停运行时，CPU便立即转去运行另一道程序

- 分时系统：把处理机的运行时间分成很短的时间片，按时间片轮流把处理机分配给各联机作业使用

- 区别：多道程序设计的资源分配不固定，根据程序需求动态调整；分时系统采用固定时间片轮转，确保每个用户或任务公平获得CPU时间

2. 什么原因推动了操作系统从批处理发展到多道程序，进而发展到分时系统？

答：

- 每次主机内存中仅存放一道作业，每当它运行期间发出I/O请求后，告诉的CPU便处于等待低俗I/O完成状态，致使CPU空闲。为了进一步提升CPU的利用率，引入了多道程序系统

- 多到系统的平均周转时间长且不能提供交互作用能力，进而引入了支持多用户、多进程的分时系统

3. 什么是陷阱？与中断的区别是什么？什么是系统调用？

答：

- 陷阱：由程序执行过程中产生的一种异常。通常需要操作系统介入处理。陷阱可以由硬件或软件触发，通常与程序的正常执行流程相关

- 陷阱是同步异常，是程序内部有意设置的，如系统调用和信号机制等；中断是异步异常，来自I/O设备的信号，如所有的IRQ中断

- 系统调用是操作系统提供给应用程序使用的接口，应用程序可以通过系统调用来请求操作系统内核的服务

4. 判断：可移植的操作系统可以从一个系统架构移植到另外一个系统架构而无 需修改。

(1) 请解释为什么构建完全可移植的 OS 是不可能的？

(2) 如果需要你设计一个高度可移植的 OS，那么请描述你需要设计的两个 层次?

答：错误

（1）不同系统架构的指令集、寄存器、内存管理方式等存在差异，操作系统需要直接与硬件交互，因此必须针对特定的结构实现底层代码

（2）硬件抽象层：提供统一的接口，隐藏底层硬件的具体实现细节，供通用代码调用；

内核层：实现操作系统的核心功能，避免直接与硬件交互

5. **在设计操作系统时，一些设计指标是相互矛盾的，例如资源利用率、吞吐量、 处理时间、健壮性等。请给出一对相互矛盾的设计实例。**

答：资源利用率 与 处理时间

为了让资源利用率提高，可以减少CPU的空闲时间，即减少任务切换带来的I/O等待时间，让任务更长时间占用CPU。但这会导致交互式任务处理时间变长，用户体验下降。

同样，为了减少处理时间，系统需要频繁进行上下文的切换，确保每个任务都可以及时获得CPU时间。但这会增加上下文切换的开销，降低资源利用率

（吞吐量和处理时间同理，单任务处理时间缩短会导致系统吞吐量减小）

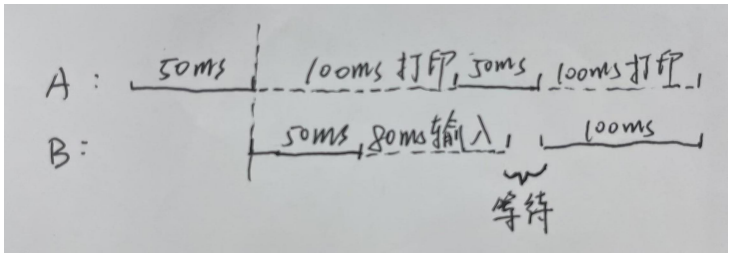
6. **一个计算机系统有输入机一台、打印机两台，现有二道程序同时投入运行， 且程序 A 先开始运行，程序 B 后运行。程序 A 的运行轨迹为：计算 50ms， 打印信息 100ms，再计算 50ms ,打印信息 100ms ,结束。程序 B 运行的轨迹 为：计算 50ms，输入数据 80ms，再计算 100ms，结束。要求：**

(1) 用图画出这二道程序并发执行时的工作情况。

(2) 说明在二道程序运行时，CPU 有无空闲等待？若有，在哪段时间内等待？为 什么会空闲等待？

(3) 程序 A、B 运行时有无等待现象？在什么时候会发生等待现象？

答：

（1） 

（2）CPU有空闲等待时间，在 100 ~ 150ms 时，A打印B输入，都在使用I/O设备，CPU空闲

（3）A没有等待现象，B有等待现象，出现在 180 ~ 200ms 时间段，其在等待A使用CPU进行计算任务