1. 为什么引入线程？

答：我认为不论是操作系统还是服务器端，包括iOS开发，程序的执行效率和线程都有着密不可分的关系，在操作系统里，线程可以提高程序的并发性和效率。在iOS开发里，我学习的多线程可以帮助用户及时的解决当前发生的事件，多个线程独立的处理任务，提高了响应速度和效率。而且线程和线程之间的数据源和资源都是存在共享的，所以他同时进行多个任务的时候更需要提高效率

1. 为什么说线程只拥有栈和少量寄存器，其他资源都共享进程的资源，都共享哪些资源？假设父亲是进程，你是线程，如果父亲有10个儿子（都是线程），类比一下资源的共享以及他们的干活速度。

答：线程只拥有栈和少量寄存器是因为线程是在进程中运行的，多个线程是共享进程的资源，类似于堆，文件描述等。所以进程不用独立的地址空间和其他资源，它们只需要自己的栈和寄存器来执行自己的代码。

当然，和进程相比，线程的创建和切换销毁的开销更小，线程只需要自己管理自己的栈和寄存器。

假设父进程有10个线程，这些线程会共享父进程的资源，包括打开的文件、信号处理器、进程 ID 等，同时这些线程也共享同一片堆区和数据段。由于线程之间的切换只需保存少量寄存器和栈信息，所以线程的创建、撤销、切换都比进程要快得多。

1. 通过Linux进程和线程创建，说说进程，用户级线程和内核级线程的区别和联系。

答：Linux实现线程由两部分组成：内核的线程支持+用户态的库支持。用户级线程与内核级线程是一对一模型

1. 运行实验楼上实验10

4.1 用户态线程是通过什么创建的，谁管理这些线程？为什么说用户态线程具有可移植性？

答：用户态是程序通过自己的线程库创建的，应用程序管理用户态线程。

可移植性是因为它们存在于自己的线程库，不依赖特定的操作系统或者内核实现，应用程序是一个可以存在于不同系统的东西。

4.2 内核级线程谁管理的？

答：内核级线程是由操作系统内核管理的。在操作系统中，内核级线程由内核管理和调度，内核级线程的创建、销毁、调度等操作都是由内核完成的。

4.3 进程和线程到底共享哪些资源，哪些不能共享？

答：能够共享：内存空间，文件描述，信号处理；

不能共享：寄存器和栈， 线程私有的数据， PCB， 栈

* 1. 运行第6步的代码，分析程序的运行结果，你有什么启发？

答：一行是主线程的信息，一行是新线程的信息。其中，pid是进程ID，ktid是线程ID，utid是线程ID（pthread库中生成的线程ID）。可以看到，不同的线程具有不同的线程ID，但是它们共享了相同的进程ID。 启发是：线程间的UTID不同，说明线程是独立的执行实体，线程之间的调度和同步需要通过系统调用来完成，而PID相同，说明所有的线程都属于同一个进程。