1. Unix I/O

1) Unix I/O：将设备映射位文件的方式，允许 Linux 内核引出一个简单、低级的应用接口，称为 Unix I/O
2) 文件类型：
   a. 普通文件
   b. 目录
      a) 目录包含有一组链接，每个链接将一个文件名映射到一个文件
      b) 每个目录至少含有两个条目：一是到该文件自身的链接；二是到目录层次结构中父目录的链接
      c) 命令：mkdir 创建空目录；ls 查看目录内容； rmdir 删除空目录
   c. 套接字
3) 文件操作
   a. 打开文件：open
   b. 关闭文件：close（关闭一个已经关闭的文件会导致程序错误）
   c. 读文件：read（读文件从当前位置复制字节到内存位置，然后更新文件位置）
   d. 写文件：write（写文件从内存复制字节到当前文件位置，然后更新文件位置）
   e. 读、写会出现“不足值”的情况
      a) 出现“不足值”的几种情况：
         i. 读时碰到 EOF
         ii. 从终端读文本行
         iii. 读写网络套接字
      b) 一般不会出现“不足值”：
         i. 读磁盘文件（除了 EOF）
         ii. 写磁盘文件（磁盘满也会）

2. RIO 包

3. 读取文件元数据、共享和重定位

1) 元数据（Metadata）：关于文件的信息，用户通过调用 stat 和 fstat 函数访问元数据
2) 如何表示打开文件：
   a. 描述符表：每个进程一张表。它的表项是由进程打开的文件描述符来索引的，每个打开的描述符表项指向文件表中的一个表项：fd0 表示 stdin；fd1 表示 stdout；fd2 表示 stderr。
   b. 打开文件表：所有进程共享。每个文件表的表项包括当前的文件位置、引用计数（refcnt），以及一个指向 v-node 表中对应表项的指针。
   c. V-node 表：所有进程共享。每个表项包含 stat 结构中的大多数信息。
3) 共享文件：注意描述符表和 refcnt 的变化
4) I/O 重定向：允许用户将磁盘文件和标准输入输出联系起来

a. 通过调用 dup2(oldfd, newfd)函数：注意先后顺序
b. 注意描述符表和 refcnt 的变化

4. 标准 I/O

1) 优点:
   a. 通过减少读和写系统调用的次数，有效增加内存
   b. 自动处理不足值
2) 缺点:
   a. 没有提供访问文件元数据的函数
   b. 标准 I/O 函数不是异步信号安全的, 不适合用于信号处理
   c. 标准 I/O 不适合网络套接字的输入输出操作