**集美大学计算机工程学院实验报告**

|  |  |  |
| --- | --- | --- |
| **课程名称**： 嵌入式系统应用 | **班级**： 计算2114 | **实验成绩**： |
| **指导教师**： 刘晋明 | **姓名**： 庄佳强 |  |
| **实验项目名称**： 文件IO编程 | **学号**： 202121331104 | **上机实践日期**： 4.28 |
| **实验项目编号**： 003 | **组号**：003 | **上机实践时间**： 4.28 |

**一、实验目的**（本次实验所涉及并要求掌握的知识点）

1.掌握文件读写及上锁操作

2.掌握open()、read()、write()、fcntl()等函数的使用

1. **实验内容与设计思想**（设计思路、主要数据结构、主要代码结构、主要代码段分析、电路图等）
2. 编写程序实现文件写入锁和读取锁的设置和运行
3. 通过给文件进行加写锁来控制读写互斥。
4. 通过给文件加读锁来实现读共享，写互斥。
5. 编写程序使用文件操作，仿真FIFO（先进先出）结构以及生产者-消费者运行模型
6. 通过写锁来实现生产者的的互斥操作，用sleep来控制周期。
7. 因为消费者为先进先出的类似管道问题，不会存在临界资源访问冲突问题，但消费者一次需要读取多个数据，就不能用lock,但在文件进行更新的时候还是需要上锁。
8. 编程程序实现文件多路复用操作。

通过系统建立两个管道，通过管道来实现多路复用。

**三、实验使用环境**（本次实验所使用的平台和相关软件）

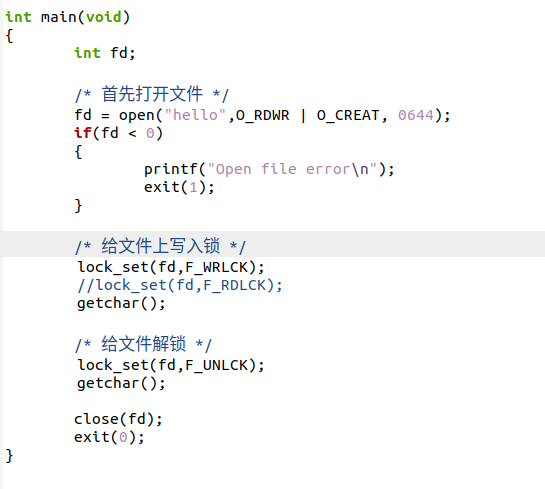
Ubuntu20.04LTS。

**四、实验步骤和调试过程**（实验步骤、测试数据设计、测试结果分析）

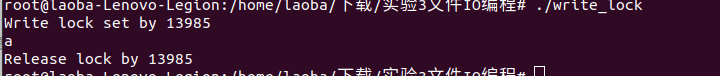
**第一部分:**

1. 编写文件写入锁的测试用例write\_lock.c：创建一个hello文件，之后对其上写入锁，键盘输入任意一个字符后解除写入锁。

先编写write\_lock.c文件



1. 在两个终端上运行./write\_lock，查看运行结果

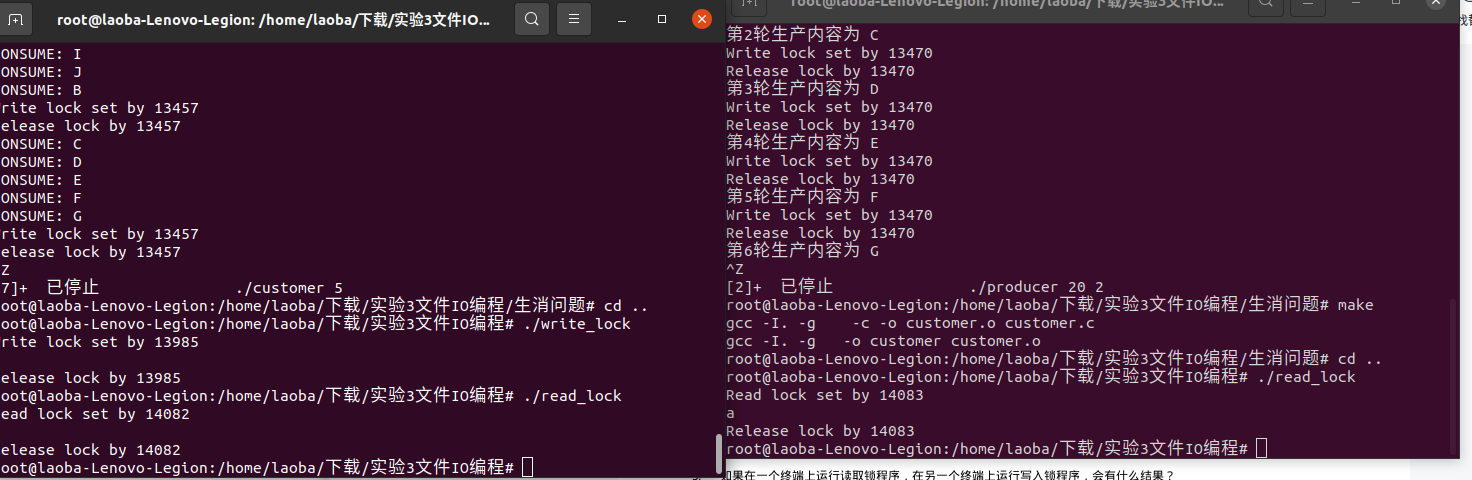


在启动后会对文件加一个互斥写锁，让文件只能该进程访问。

1. 编写文件读取锁的测试用例read\_lock.c：创建一个hello文件，之后对其上读取锁，键盘输入任意一个字符后解除读取锁。

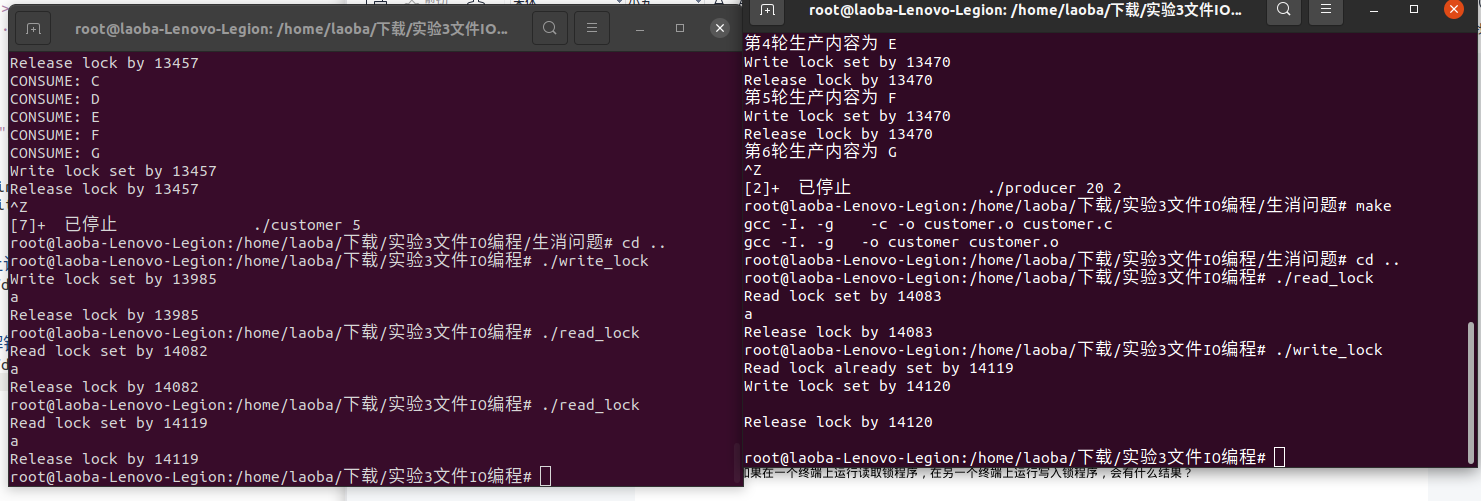


1. 在两个终端上运行./read\_lock，查看运行结果。



写锁之间是共享锁，可以共享的读取文件内容。

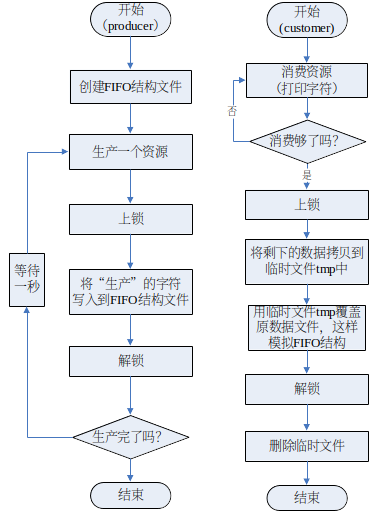
1. 如果在一个终端上运行读取锁程序，在另一个终端上运行写入锁程序，会有什么结果？



读锁和写锁之间是互斥锁。

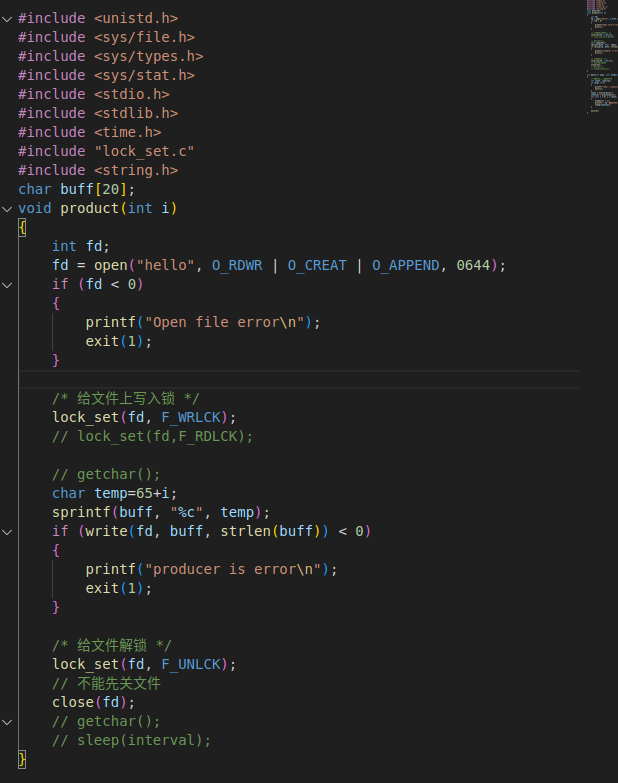
总结为：

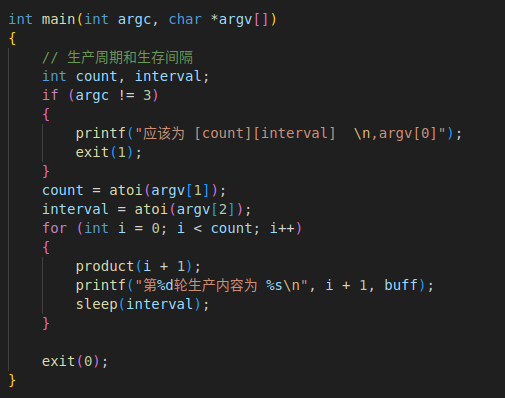
**第二部分:**



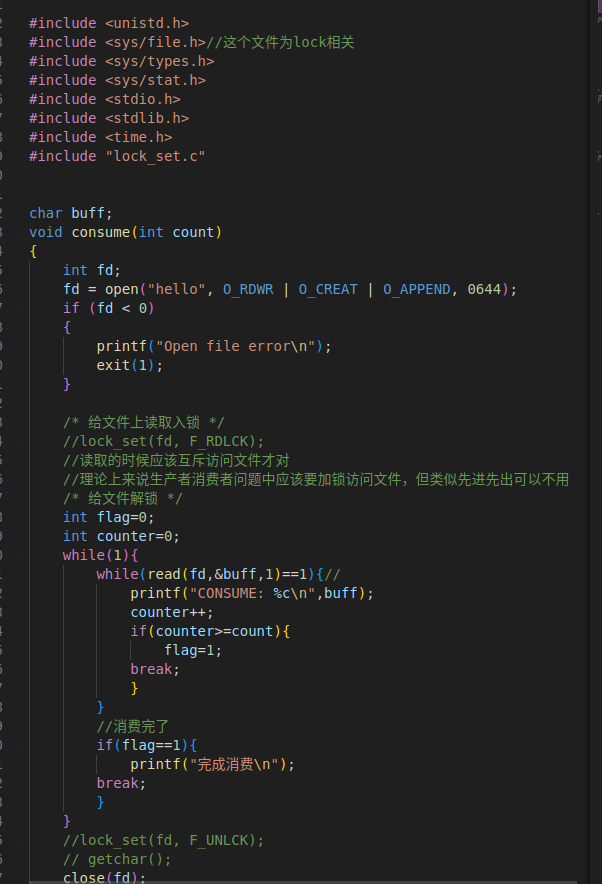
只需要依照这个写代码即可。

编程实现生产者程序producer.c，创建仿真FIFO结构文件（普通文件），按照给定的时间间隔向FIFO文件写入自动生成的字符（自定义），生产周期及生产的资源数通过参数传递给进程。

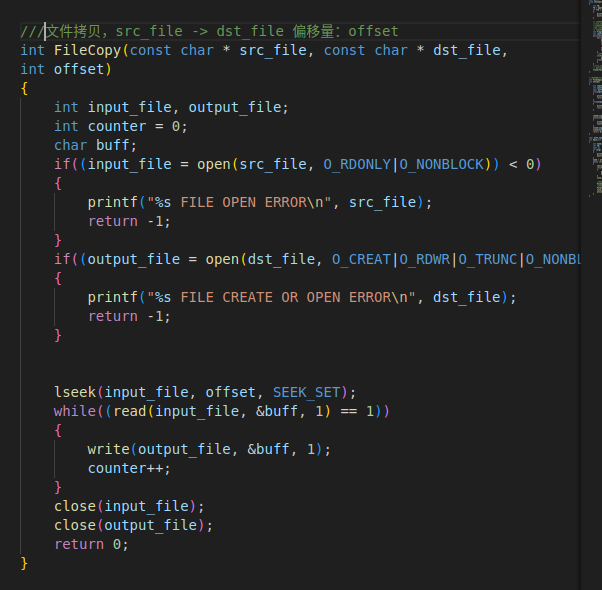


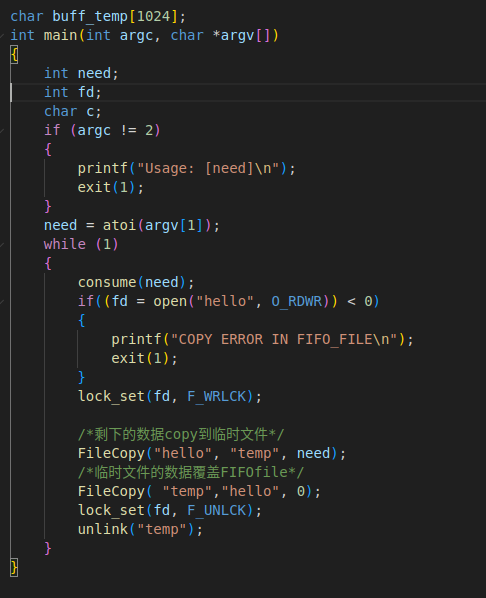


编程实现消费者程序customer.c，从文件中读取相应数目的字符并在屏幕上显示，然后从文件中删除刚才消费过的数据，可通过两次幅值来实现文件内容的偏移，每次消费的资源通过参数传递给进程。

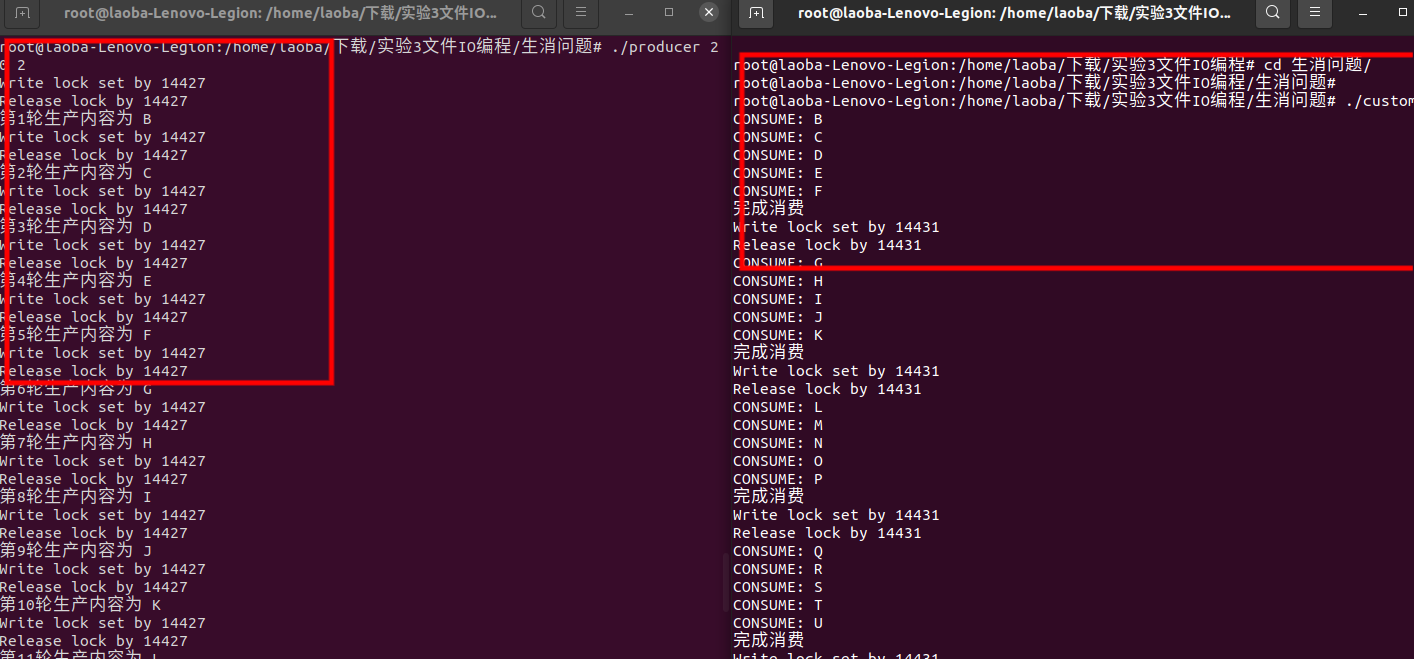


因为用open打开的文件中没法进行覆盖操作，所以只能使用swap这样占空间操作。





在两个终端上分别运行生产者程序producer和消费者程序customer

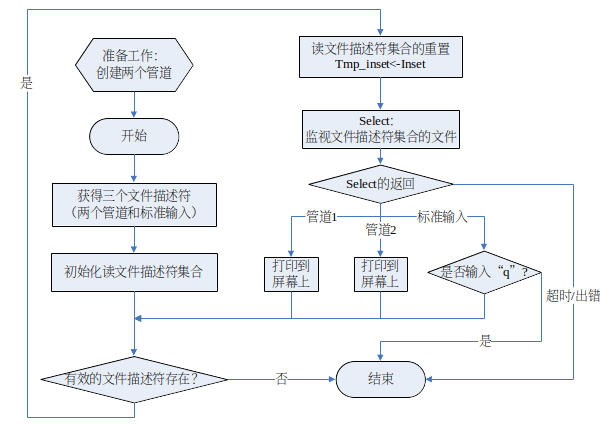


可以看到当生产者生产5歌物品后消费者也消耗完对于的物品后，就会更新FIFO文件，让消费了的文件被覆盖掉，模拟使用的过程，这一过程中是互斥访问。

**第二部分:**

多路复用—I/O操作及阻塞

编程实现文件描述符集合的监听。



运行时，需要打开3个虚拟终端，分别创建两个管道文件in1和in2，运行主程序

终端1：

mknod in1 p

cat >in1

MULTIPLEX CALL

TEST IN1

END

终端2：

mknod in2 p

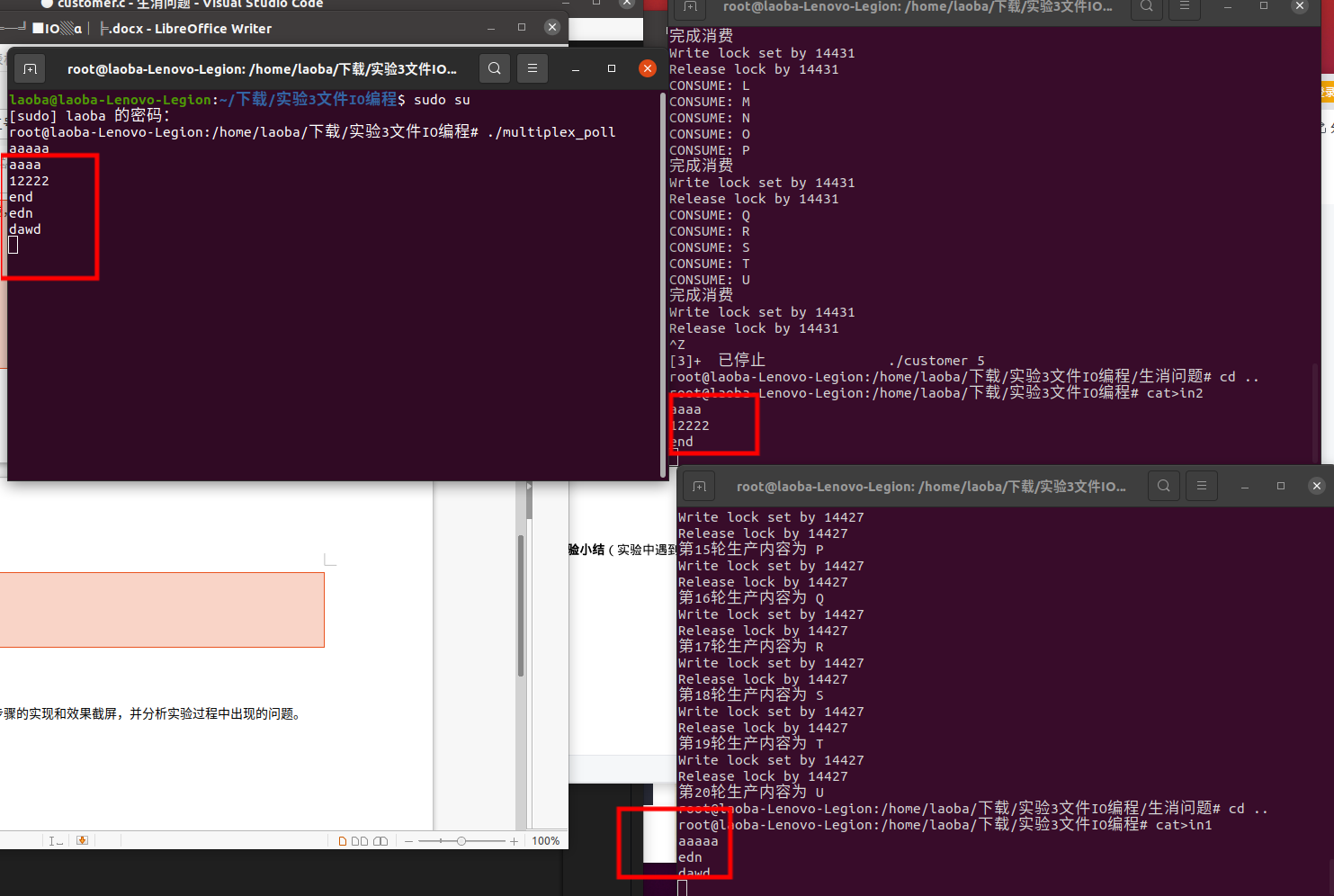
multiplex call

test in2

end

终端3:

./multiplex\_poll



mknod in1 p命令用于在Linux系统中创建一个命名管道（Named Pipe）。其中，mknod是命令本身，in1是要创建的命名管道的名称，p表示该命名管道是一个管道类型。

命名管道是一种特殊的文件类型，允许进程之间进行双向通信。进程可以像读写文件一样读写命名管道，但是数据不会像普通文件一样存储，而是直接传递。因此，命名管道可用于实现进程间通信（IPC）。

使用mknod命令创建命名管道时，如果该管道不存在，则会在指定路径下创建一个新的命名管道文件。若已存在同名的命名管道，则mknod命令将返回错误。创建命名管道后，可以使用标准的文件读写操作进行读写操作。

**五、实验小结**（实验中遇到的问题及解决过程、实验中产生的错误及原因分析、实验体会和收获）

这次实验让我更深入地理解了操作系统的核心概念和机制，比如文件系统、进程管理和线程管理等。通过编写各种实验程序，我掌握了使用各种系统调用和库函数的方法，比如fcntl、mkfifo和poll等，这些函数可以用于管理和操作文件系统以及处理进程和线程。除此之外，我还学会了如何利用并发编程技术来提高程序的性能和可靠性。