§0.Linux 知识补充 - 进程间的通信

查阅相关资料， 在 RHEL7.4 下初步掌握进程间通信的相关知识点并将答案写成文档：

【本次作业背景】：在 Linux 编程的实际应用中，我们往往会有在不同的进程间传递信息的需求，需要传递的信息既可能只是一个简单的通知，也可能是一批数据的单向/双向传输，还有可能是多个进程共享数据；而不同的进程可能是父子进程，也可能是毫无关系的两个独立进程

在 RHEL7.4 下，使用 C/C++语言，按要求完成以下小题（每个小题放在一个子目录下）：

0、每个人的目录结构要求如下（假设学号为 1551234，各人按实修改）：首先建立"学号-000110"子目录，作业目录可位于任意子目录下，下面再建立若干空的子目录，示例如下：

```
1551234-000110
  |-- 01
  |-- ..
  |-- 07
```

1、无名管道（01 子目录）

- 写测试程序 test1-1，再 fork 子进程，父子进程间建立无名管道，父进程向子进程发送数据
- 写测试程序 test1-2，再 fork 子进程，父子进程间建立无名管道，子进程向父进程发送数据
- 写测试程序 test1-3，再 fork 子进程，父子进程间建立无名管道，能否双向传递数据？
- 无名管道方式传递的数据的类型，长度是否有限制？
- 能否在独立进程间用无名管道通信？

2、有名管道（02 子目录）

- 写测试程序 test2-1，再 fork 子进程，父子进程间建立有名管道，父进程向子进程发送数据
- 写测试程序 test2-2，再 fork 子进程，父子进程间建立有名管道，子进程向父进程发送数据
- 写测试程序 test2-3，再 fork 子进程，父子进程间建立有名管道，能否双向传递数据？
- 一对测试程序 test2-4-1/test2-4-2，两个进程间建立有名管道，test2-4-1 向 test2-4-2 发送数据
- 一对测试程序 test2-5-1/test2-5-2，两个进程间建立有名管道，双向传递数据，能否做到？
- 有名管道方式传递的数据的类型，长度是否有限制？和无名管道相比是否有区别？

3、信号方式（03 子目录）

- 一对测试程序 test3-1-1/test3-1-2，在 test3-2 中约定截获几个信号用自定义函数处理，其中部分是截获后打印信息，程序继续；部分是截获后打印信息，程序退出。test3-1 中定时向 test3-2 发送相应的信号；同时可以通过控制台手工向 test3-2 发送相应的信号
- 信号能否带数据？
- 哪几个信号不能被截获并重定义？（kill -l 可查看系统支持的信号值）
- 一对测试程序 test3-2-1/test3-2-2，要求 3-2-1 写文件（此时 3-2-2 为延时等待状态）后用自定义的特定信号通知 3-2-2 去读，自己变为延时等待状态；3-2-2 收到此信号后去读文件的内容，读取完成后清空文件，写入新内容，再用自定义的特定信号通知 3-2-1 去读，自己变为延时等待状态，循环往复（提示：用什么方法可以知道对方的 pid 值并传递信号）

4、消息队列方式（04 子目录）

- 一对测试程序 test4-1-1/test4-1-2，建立消息队列方式，然后从 test4-1-1 向 test4-1-2 单向传递数据
- 一对测试程序 test4-2-1/test4-2-2，建立消息队列方式，然后双向传递数据，能否做到？
- 消息队列方式传递的数据的类型，长度是否有限制？ 和无名/有名管道相比是否有区别？

5、共享内存方式（05 子目录）

- 一对测试程序 test5-1/test5-2，要求共享一段内存，test5-1 向这段内存写入内容后，test5-2 能读到写入的内容，反之亦然
- 是否只能在父子进程间共享内存？还是可以独立进程间共享？
- 如果两个进程同时写，共享内存内容是否会乱？如何防止共享内存内容乱？

6、Unix 套接字方式（06 子目录）

- 一对测试程序 test6-1-1/test6-1-2，要求在两个进程间建立 Unix 类型的 Socket（类似 TCP 方式），然后通过 socket 读写来实现进程的双向通信（通信内容自行定义即可）
- 一对测试程序 test6-2-1/test6-2-2，要求在两个进程间建立 Unix 类型的 Socket（类似 UDP 方式），然后通过 socket 读写来实现进程的双向通信（通信内容自行定义即可）
- Unix 类型 Socket 的建立、使用等与 TCP/UDP Socket 相比有什么相同和不同点？
- Unix 类型 Socket 是否有阻塞和非阻塞方式？是否能通过 select 来读写？写满后是返回不可写还是继续可写而导致数据丢失？在测试程序中表现出来

7、文件锁机制（07 子目录）

- 本小题要求文件的打开及读写方式为 open/close/read/write，不准用 fopen/fclose 系列
- 一对测试程序 test7-1-1/test7-1-2，要求 7-1-1 对某个特定文件新建/打开后加写锁，延时一段时间（此时启动 7-1-2）后再向文件中写入一些内容，写入完成后释放锁，进入死循环等待状态；7-1-2 同样打开该文件后加读锁，此时会被阻塞，直到 7-1-1 释放写锁后，7-1-2 才会退出阻塞状态，读取 7-1-1 刚才写入的内容后进入死循环等待状态
- 一对测试程序 test7-2-1/test7-2-2，在程序中打开文件后将 fd 设置为非阻塞方式，其余要求同 test7-1-1/test7-1-2
- 锁定文件有几种方法？不同的方法对阻塞/非阻塞方式的 fd 是否有区别？
- 在一个程序对文件加写锁后，另一个程序不加锁而直接读写（都不设置为非阻塞方式），是阻塞在 read/write 上，还是 read/write 直接返回失败？
- 在一个程序对文件加写锁后，另一个程序不加锁而直接读写（都设置为非阻塞方式），是阻塞在 read/write 上，还是 read/write 直接返回失败？

【本次作业目录结构要求及批改方法：】

1、每个人的目录结构要求如下（假设学号为 1551234，**各人按实修改**）：首先建立"1551234-000110"子目录（可位于任意子目录下），下面再建立 01-07 的子目录，示例如下：

```
1551234-000110
|-- 01
|-- 02
|-- ...
`-- makefile   （每位同学的总 makefile 文件，make 后能生成所有子目录下的可执行文件）
```

2、提交作业时，每位同学上交一个 linux-ipc.tar.bz2 文件，解压后能得到上述的完整目录结构，截止时间到后，会从每人的交作业目录中复制出来，全部放在 total-000110 目录中

示例如下：

```
total-000110
|-- 1551234-linux-ipc.tar.bz2      （第 1 位同学的作业压缩包）
...
`-- 1554321-linux-ipc.tar.bz2      （最后 1 位同学的作业压缩包）
```

依次解压后，能得到如下目录结构：

```
total-000110
|-- 1551234-000110                      （第 1 位同学的作业目录）
...
```

```
`-- 1554321-000110                    （最后 1 位同学的作业目录）
```

3、进入 total-000110 目录，进行一次 make，就能生成所有可执行文件，示例如下：

```
total-000110
|-- 1551234-000106                    （第 1 位同学的作业目录）
...
|-- 1554321-000106                    （最后 1 位同学的作业目录）
`-- makefile          （老师事先建好的 makefile 文件，准备编译所有同学的本次作业，具体的实现方式
                        是进入到每个学号对应的目录后调用该目录下的总 makefile）
```

4、无法顺利编译则不能得分，对应学号及子目录名错则不能得分

5、作业提交时清除所有的中间文件及生成的可执行文件、源程序备份文件等

【作业要求：】

1、11 月 19 日前网上提交

2、每题所占平时成绩的具体分值见网页

3、超过截止时间提交作业会自动扣除相应的分数，具体见网页上的说明