Week13 Assignment

班级：192112
学号：19373073
姓名：何潇龙

1. 简述信号量的作用，如何利用信号量实现同步和互斥?

信号量，是在多线程环境下使用的一种设施，是可以用来保证两个或多个关键代码段不被并发调用。在进入一个关键代码段之前，线程必须获取一个信号量；一旦该关键代码段完成了，那么该线程必须释放信号量。其它想进入该关键代码段的线程必须等待直到第一个线程释放信号量。
同步：
由一个无条件执行的进程开始，信号量资源初始值为0，该进程结束进行V(S)操作，信号量资源+1，此时会通知别的进程。需要该资源的进程，打破阻塞，P(S)操作，去访问临界资源。
互斥：
P(S)操作，资源-1，则其余进程不可再访问该份临界资源。使用完V(S)操作，资源+1，此时别的进程才可进行是否可以访问判断。

2.简述共享内存的作用和用法，共享内存为什么需要与信号量一起使用?

共享内存指在多处理器的计算机系统中，可以被不同CPU访问的大容量内存。由于多个CPU需要快速访问存储器，这样就要对存储器进行缓存。任何一个缓存的数据被更新后，由于其他处理器也可能要存取，共享内存就需要立即更新，否则不同的处理器可能用到不同的数据。
可通过使用以下函数共享内存：
创建共享内存（shmget）
访问共享内存（shmat）
解除绑定（shmdt）
释放共享内存（shmctl）
共享内存并未提供同步机制。也就是说，在第一个进程结束对共享内存的写操作之前，并无自动机制可以阻止第二个进程开始对它进行读取，所以通过信号量和共享内存一起使用，实现进程间的同步或互斥高速通信。

3. 简述消息队列的作用和用法，并将其与管道进行异同点对比。

“消息队列”是在消息的传输过程中保存消息的容器。消息队列管理器在将消息从它的源中继到它的目标时充当中间人,是一个消息的链表,即将消息看作一个记录，并且这个记录具有特定的格式以及特定的优先级。对消息队列有写权限的进程可以按照一定的规则添加新消息到队列的末尾；对消息队列有读权限的
进程则可以从消息队列中读取消息。
可通过使用以下函数共享内存：
创建消息队列(msgget)
发送数据到消息队列(msgsnd)
从消息队列中读取数据(msgrcv)
删除队列(msgctl)

管道一般用于父子进程间通信（有名管道除外，有名管道不限于父子进程通信）。而消息队列可用于机器上的任何进程间通信（只要进程有权操作消息队列）
消息队列与管道以及有名管道相比，具有更大的灵活性，首先，它提供有格式字节流，有利于减少开发人员的工作量；其次，消息具有类型，在实际应用中，可作为优先级使用。这两点是管道以及有名管道所不能比的。同样，消息队列可以在几个进程间复用，而不管这几个进程是否具有亲缘关系，这一点与有名管道很相似；但消息队列是随内核持续的，与有名管道（随进程持续）相比，生命力更强，应用空间更大。

4. 请查阅资料简述进程间通信的System V、POSIX两种标准之间的差异性。

POSIX：
POSIX(Portable Operating System Interface for Computing Systems)是由IEEE 和ISO/IEC 开发的一簇标准。该标准是基于现有的UNIX 实践和经验，描述了操作系统的调用服务接口，用于保证编制的应用程序可以在源代码一级上在多种操作系统上移植运行。它是在1980 年早期一个UNIX 用户组(usr/group)的早期工作的基础上取得的。该UNIX 用户组原来试图将AT&T 的系统V 和Berkeley CSRG的BSD 系统的调用接口之间的区别重新调和集成，从而于1984 年产生了/usr/group 标准。1985 年，IEEE 操作系统技术委员会标准小组委员会(TCOS-SS)开始在ANSI 的支持下责成IEEE 标准委员会制定有关程序源代码可移植性操作系统服务接口正式标准。到了1986 年4 月，IEEE 就制定出了试用标准。第一个正式标准是在1988 年9 月份批准的(IEEE 1003.1-1988)，也既以后经常提到的POSIX.1 标准。

System V:
System V， 曾经也被称为 AT&T System V，是Unix操作系统众多版本中的一支。它最初由 AT&T 开发，在1983年第一次发布。一共发行了4个 System V 的主要版本：版本1、2、3 和 4。System V Release 4，或者称为SVR4，是最成功的版本，成为一些UNIX共同特性的源头，例如 ”SysV 初始化脚本“ (/etc/init.d)，用来控制系统启动和关闭，System V Interface Definition (SVID) 是一个 System V 如何工作的标准定义。
AT&T 出售运行System V的专有硬件，但许多(或许是大多数)客户在其上运行一个转售的版本，这个版本基于 AT&T 的实现说明。流行的SysV 衍生版本包括 Dell SVR4 和 Bull SVR4。当今广泛使用的 System V 版本是 SCO OpenServer，基于 System V Release 3，以及SUN Solaris 和 SCO UnixWare，都基于 System V Release 4。
System V 是 AT&T 的第一个商业UNIX版本(UNIX System III)的加强。传统上，System V 被看作是两种UNIX"风味"之一(另一个是 BSD)。然而，随着一些并不基于这两者代码的UNIX实现的出现，例如 Linux 和 QNX， 这一归纳不再准确，但不论如何，像POSIX这样的标准化努力一直在试图减少各种实现之间的不同。

POSIX 在无竞争条件下，不需要陷入内核，其实现是非常轻量级的; System V 则不同，无论有无竞争都要执行系统调用，因此性能落了下风。

5. 请编写一个管道通信程序实现文件的传输，并请思考采用其他进程通信方式是否可方便实现进程间的文件传输。

```c
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <limits.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <stdio.h>
#include <string.h>
#define MAXSIZE 2000
int main(int argc,char *argv[]){
    int wt;
    int fd_src = open(argv[1],O_RDONLY);
    char buf[MAXSIZE];
    int ret;
    mkfifo("./charlotpipe", 0777);
    wt = open("./charlotpipe", O_WRONLY);
    while(1){
        ret=read(fd_src,buf,MAXSIZE-1);
        if(ret==0)break;
```

```
        write(wt, buf, ret);
    }
    close(wt);
    close(fd_src);
    return 0;
}
```

```
#include <unistd.h>
#include <stdlib.h>
#include <fcntl.h>
#include <limits.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <stdio.h>
#include <string.h>
#define MAXSIZE 2000
int main(int argc,char *argv[]){
    int rd;
    int fd_tgt=open(argv[1],O_WRONLY);
    int ret;
    char buf[MAXSIZE];
    rd = open("./charlotpipe", O_RDONLY);
    while(1){
        ret=read(rd, buf, MAXSIZE-1);
        if(ret==0)break;
        write(fd_tgt,buf,ret);
    }
    close(rd);
    close(fd_tgt);
    return 0;
}
```