Q1:

进程的的基本定义是：在自身的虚拟地址空间运行的一个独立的程序，从操作系统的角度来看，所有在系统上运行的东西，都可以称为一个进程。

Linux是一个多任务的操作系统。 多任务是指在Linux下可以同时执行多个任务，更详细的说，Linux采用了分时管理的方法，所有的任务都放在一个队列中L，操作系统根据每个任务的优先级为每个任务分配合适的时间片，每个时间片很短，用户根本感觉不到是多个任务在运行，从而使所有的任务共同分享系统资源，这就是多任务的概念。

进程是操作系统中最基本、重要的概念。是多道程序系统出现后，为了刻画系统内部出现的动态情况，描述系统内部各道程序的活动规律引进的一个概念,所有多道程序设计操作系统都建立在进程的基础上。

Q2:

代码A父进程fork出子进程后，子进程会继续fork出子进程。

代码B中父进程fork出子进程后，子进程会break，不会再fork出子进程。

原因： fork成功调用一次则返回两个值，子进程返回0，父进程返回子进程ID，因此B中的子进程会break出循环，不再fork。

Q3:

一个进程使用fork创建子进程，如果子进程退出，而父进程并没有调用wait或waitpid获取子进程的状态信息，那么子进程的进程描述符仍然保存在系统中。这种进程称之为僵尸进程。

进程一旦调用了wait，就立即阻塞自己，由wait自动分析是否当前进程的某个子进程已经退出，如果让它找到了这样一个已经变成僵尸的子进程，wait就会收集这个子进程的信息，并把它彻底销毁后返回；如果没有找到这样一个子进程，wait就会一直阻塞在这里，直到有一个出现为止。

Q4:

信号用来通知进程发生了异步事件。在软件层次上是对中断机制的一种模拟，在原理上，一个进程收到一个信号与处理器收到一个中断请求可以说是一样的。信号是进程间通信机制中唯一的异步通信机制，一个进程不必通过任何操作来等待信号的到达，事实上，进程也不知道信号到底什么时候到达。进程之间可以互相通过系统调用kill发送软中断信号。内核也可以因为内部事件而给进程发送信号，通知进程发生了某个事件。信号机制除了基本通知功能外，还可以传递附加信息。

Q5:

未决状态信号的产生主要是因为进程对此信号的阻塞。例如为进程产生一个选择为阻塞的信号，而且对该信号的动作是系统默认动作或捕捉该信号，则为该进程将此信号保持为未决状态，直到该进程对此信号解除了阻塞或者对此信号的动作改为忽略。

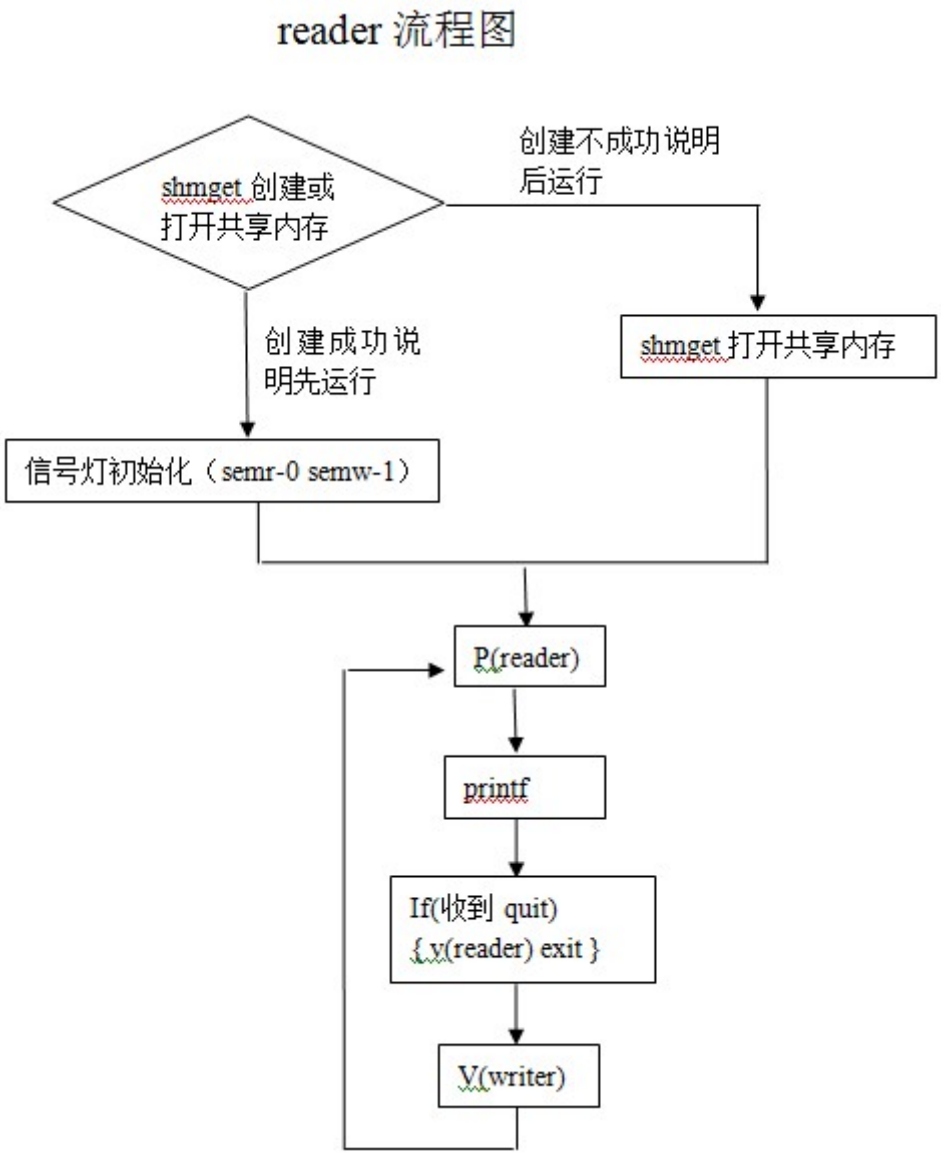
未决状态可以使信号被内核递送时进程再决定对其的处理方式，而不是信号产生时。

Q6:

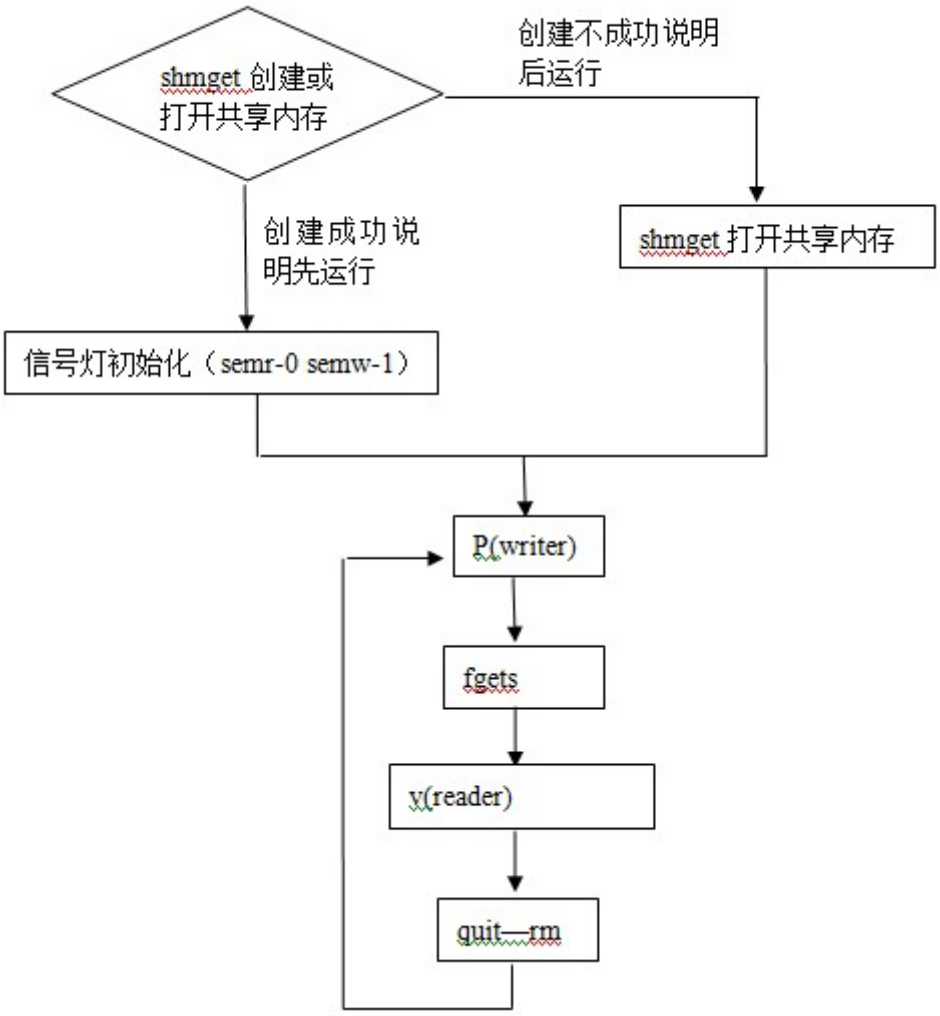
利用POSIX无名信号灯实现共享内存的同步

共享内存定义以下的结构体。

```
typedef struct
    {
        sem_t semr;//控制能否读，初始化为0
        sem_t semw;//控制能否写，初始化为1
        char buf[MAXSIZE];
    }SHM;
```



writer 流程图


shmget 创建或
打开共享内存
创建不成功说明
后运行
创建成功说
明先运行
shmget 打开共享内存
信号灯初始化（semr-0 semw-1）
P(writer)
fgets
v(reader)
quit—rm