No.

Date

5.2 a. ① read 读取卡片，将字符串通过rs传给 squash，并且传一个额外的空白符

② squash 在字符串中寻找连续的星号并替换它们，再通过 sp 传给 print

③ print 接收字符流，将它们打印为 125 字符的行。

b. 用三个并发过程

① 第一个进程，读取卡片，将字符传给第二个进程

② 第二个进程修改连续的星号并传给第三个进程

③ 第三个进程打印内容

使用生产者/消费者信号量进行实现

5.7 a. 当一个进程想要进入临界区时，它会得到一个票号。这个票号比当前其他进程的票号都大。票号最小的进程最优先进入临界区，离开临界区时，进程将自己的票号置为0。所有进程

b. 如果每个进程拿到的票号互不相同，那么就有且仅有一个进入临界区的顺序，所以死锁不会发生。

c. 当一个进程在临界区中时，根据算法，其他进程不能进入该临界区访问资源，所以保证了互斥。

5.13 a. 有两个问题：

① 由于被唤醒的进程需要重新进入临界区（第10行），新来的进程可能会在它们之前进入临界区（第5行）。

② 第29行让 ~~must_wait = must_wait~~ must_wait = false 时，可能已经有3个进程被唤醒，但此时有第4个进程通过第6行的判断也进入临界区，就会有4个进程同时在临界区中。

b. 可以解决第②个的互斥问题，但这样可能让后来的进程

"插队"，比先来的进程更快进入临界区，而导致饥饿问题。

5.14 a. 第30行 must_wait = active == 3 不能保证等在block处的进程和新来的进程不会同时进入临界区；
等待的进程也无需重新 SemWait (mutex) 重新进入互斥.

b. 会有插队的情况。若有一个进程执行完第8行时时间片到了，而此时有一个新进程到来并执行到第9行，刚好被唤醒，就会比先来的进程先执行。

c. 由释放 ~~[illegible]~~ 离开临界区的进程为被它唤醒的进程更新全局变量。

5.23
~~5.24~~ ~~#define REINDEER 9~~
~~#define ELVES 3~~

```
// Reindeer process
~~while (true)~~ {
    tan on the beaches.
    SemWait (rmutex)
    rein_ct++
    if (rein_ct == 9) {
        SemSignal (rmutex)
        SemSignal (Santa)
    }
    else {
        SemSignal (rmutex)
        semWait (rein_semWait)
    }
```

```
    SemWait( sleigh)
    deliver toys
    SemWait(done)
}

//elf process
While(true){
    SemWait (only_elves)
    sem Wait(emutex)
    elf_ct++
     if (elf_ct == 3){
       Sem Signal (emutex)
        semSignal (santa)
    }
    else {
      semSignal (mutex)
      SemWait (santa_semSignal)
    }
    SemWait(problem)
    ask Wait(elf_done)
    Sem Signal( only_elves)
}

//Sata process.
While(true){
    SemWait( Sata)
```

```
if (rein_ct == 9) {
  semWait(rmutex)
  rein_ct=0;
  SemSignal(rmutex)
 for (i=0; i<8; i++) {
   Sem Signal (rein_semWait)
  }
 for (i=0; i< 9; i++) {
     SemSignal (sleigh)
  }
  deliver toys
 for (i=0; i<REINDEER; i++) {
    semSignal (done)
  }
else {
  for (i=0; i<2; i++)
    SemSignal (santa semSignal)
    SemWait (emutex)
    elf_ct=0
    SemSignal (emutex)
   for(i=0; i<3; i++) {
     SemSignal (problem)
   }
   Answer question
 } SemSignal (elf done)
}
```

No.

Date

5.28 会出现饥饿问题。若一直有读者请求对进行读操作，readcount
就一直不会回到0，写者就一直不能执行写操作。