E03：并发进程（进程通信）参考答案

参考答案与说明

1. D
2. D
3. A
4. B
5. B
6. C
7. D
8. B
9. D
10. B
11. C
12. D
13. ①一次仅允许一个进程访问的资源　　②进程中访问临界资源的那段程序代码
14. 等待
15. ①P　　②V
16. 1 至-(m-1)
17. s.value<0
18. 互斥
19. ①P 操作　　②v 操作
20. 只有一个
21. 同步
22. ①P 操作　　②v 操作

23. 答：PV 操作是指在信号量上进行的 P 操作和 V 操作。假定信号量为 s，则 P(s)和 v(s)的定义如下：

```
procedure p(Var s:semaphore);
begin
        s.value:=s.value-1;
        ifs.value<0 then sleep(s);
    end;
procedureV(Var s:semaphore);
begin
        s.value:=s.value+1;
        ifs.value<=0 then wakeup (s);
    end;
```

其中，sleep(s)表示将调用 P(s)过程的进程置成“等待信号量 s”的状态，且将其排入等待队列。wakeup(s)表示释放一个“等待信号量 s”的进程，该进程从等待队列退出，并加入就绪队列中。

信号量 S 的物理意义如下：s.value＞0 时，S 表示可使用的资源数或表示可同时使用资源的进程数。s.value＝0 时，表示无资源可供使用和表示不允许进程再进入临界区。s.value＜0 时，｜s.value｜表示等待使用资源的进程个数或表示等待进入临界区的进程个数。

24. 参考伪代码如下：

变量：
waiting：表示等待的顾客数量。

信号量：
mutex：　用于对waiting的互斥访问
customers：有等待复印的顾客
operator：　有等候顾客的操作员

```
int waiting = 0;
semaphore mutex, customers, barbers ;
mutex.value : =1;
customers.value : =0;
barbers.value : =0;
```

```
process operator( )       //操作员进程
{
   while(1)
      {
          p(customers);   //等待顾客到来
          复印;
          v(operator);    //顾客完成复印
      }
}
```

```
process cusotmeri( )     //顾客进程
 {
       p(mutex);
       if(waiting<5)
       {
           waiting++;
           v(customers);
           v(mutex);

           p(operator);

           复印;

           p(mutex);
           waiting--;
           v(mutex);
       }
       else
       {
           v( mutex);
           离开复印室;
       }
}
```

```
main( )
{
    cobegin
     {
          operator( );
          customeri( );
     }
}
```

25. C
26. B
27. A
28. D
29. C
30. B
31. B
32. B
33. ①安全状态　②不安全状态
34. 请求和保持
35. ①死锁的避免　②死锁的预防　③死锁的解除

36. （1）调整表格如下：可用资源（2，1，0，0）

进程	当前已经分配到的资源	最大资源需求	仍需要资源
P1	0，0，1，2	0，0，1，2	0，0，0，0
P2	2，0，0，0	2，7，5，0	0，7，5，0
P3	0，0，3，4	6，6，5，6	6，6，2，2
P4	2，3，5，4	4，3，5，6	2，0，0，2
P5	0，3，3，2	0，6，5，2	0，3，2，0

则存在以下执行序列（安全序列），执行过程列表如下 ：

进程	可用资源数（剩余资源数+已经分配资源数）
P1	2，1，1，2
P4	4，4，6，6
P5	4，7，9，8
P2	6，7，9，8
P3	6，7，12，12

则该状态是安全的。

（2）、假设 P3 发出资源请求（0，1，0，0），系统分配给它，则系统还剩余资源（2，0，0，0），并且状态如下表所示：

进程	当前已经分配到的资源	最大资源需求	仍需要资源
P1	0，0，1，2	0，0，1，2	0，0，0，0
P2	2，0，0，0	2，7，5，0	0，7，5，0
P3	0，1，3，4	6，6，5，6	6，5，2，2
P4	2，3，5，4	4，3，5，6	2，0，0，2
P5	0，3，3，2	0，6，5，2	0，3，2，0

则新的执行过程如下表所示：

进程（完成后）	可用资源数（剩余资源数+已经分配资源数）
P1	2，0，1，2
P4	4，3，6，6
P5	4，6，9，8

P5 执行后，不能继续执行下去，则该状态不安全，系统将拒绝资源请求。