操作系统备考冲刺

Day 3 习题卷

专题：进程同步与互斥 · PV操作

题型	题量	分值	建议时间
填空题	8 空	8 分	5 分钟
选择题	6 题	12 分	10 分钟
判断题	5 题	10 分	5 分钟
代码题	4 题	50 分	40 分钟
合 计		**80 分**	**约 60 分钟**

⚠ 代码题是本章重点。生产者-消费者、读者-写者每年必考一道。请仿照模板写完整代码，包括 cobegin/coend、信号量初值。

一、填空题（每空 1 分，共 8 分）

1. 进程同步是指多个进程为完成共同任务而 _____ 彼此的执行顺序；进程互斥是指多个进程竞争使用 _____ 资源。

2. P 操作的作用是 _____ 资源；V 操作的作用是 _____ 资源。

3. 信号量 s 的值为 −2，表示 _____。

4. 用 PV 操作实现互斥时，互斥信号量 mutex 的初值应设为 _____。

5. 在生产者-消费者问题中，3 个信号量的初值分别是：empty = _____，full = _____。

二、单项选择题（每题 2 分，共 12 分）

1. 下列关于信号量的说法中，正确的是（ ）

A. 信号量的值可以任意修改

B. 信号量只能通过 P、V 操作改变其值

C. 信号量初值可以为负数

D. P、V 操作是普通函数调用

2. PV 操作必须成对出现，对于互斥操作，P、V 操作应该（ ）

A. 位于不同的进程中

B. 位于同一个进程中

C. 只用 P 不用 V

D. 顺序任意

3. 在生产者-消费者问题中，若 P(empty) 和 P(mutex) 顺序颠倒，会导致（ ）

A. 缓冲区溢出

B. 产品丢失

C. 死锁

D. 程序正常运行，但效率降低

4. 信号量 s 的当前值为 −3，表示（ B ）

A. 资源还剩 3 个

B. 有 3 个进程在等待该资源

C. 该资源已被使用 3 次

D. 系统出错

5. 在读者-写者问题中，第一个读者进入时执行 P(W) 操作的目的是（ A ）

A. 让其他读者等待

B. 阻止写者进入

C. 计数器加一

D. 唤醒等待的写者

6. 下列哪种情况属于"同步"而不是"互斥"？（ C ）

A. 两个进程都要用打印机

B. 多个进程访问同一全局变量

C. 消费者必须等生产者放好产品才能取

D. 多个进程争抢 CPU

三、判断题（每题 2 分，共 10 分，对的打 ✓，错的打 ✗）

1.（ ✓ ）进程互斥是进程同步的一种特殊情况。

2.（ ✗ ）互斥信号量的初值必须设为 0。

3.（ ✓ ）P、V 操作是原语，执行过程中不能被中断。

4.（ ✗ ）信号量的初值可以被多次修改。只能设置一次，后只能通过PV改变

5.（ ✓ ）当 PV 操作使用不当时，可能产生死锁。

四、代码题

1.（8分）最简单的互斥问题

两个进程 P1 和 P2 都需要使用同一台打印机。请用信号量和 PV 操作实现两个进程对打印机的互斥使用。

> 提示：互斥模板，1 个信号量，初值为 1。请写出：（1）信号量定义与初值；（2）P1 和 P2 的完整代码框架（用 cobegin/coend）。

```
semaphore mutex
mutex:=1
cobegin
  process P1:
    begin
      P(mutex)
      使用打印机
      V(mutex)
    end
  process P2
    begin
      P(mutex)
      使用打印机
      V(mutex)
    end
coend
```

2.（10分）最简单的同步问题

进程 A 负责计算数据，进程 B 负责打印数据。要求进程 B 必须在进程 A 完成计算之后才能开始打印。请用信号量和 PV 操作实现这一同步。

> 提示：同步模板，1 个信号量，初值为 0。请写出：（1）信号量定义与初值；（2）A 和 B 的完整代码，并标注 P、V 在哪里。

```
semaphore S
S:=0;
cobegin
  process A
    begin
      计算数据
      V(S)//通知B
    end
  process B
    begin
      P(S)//由A控制
      打印数据
    end
coend
```

3.（16分）生产者-消费者问题（必考！）

若干生产者进程和消费者进程通过一个有界缓冲区交换数据。缓冲区大小为 N。生产者往缓冲区中放产品，消费者从缓冲区中取产品。

要求：

① 缓冲区满时，生产者必须等待；

② 缓冲区空时，消费者必须等待；

③ 任一时刻只能有一个进程操作缓冲区。

请用信号量和 PV 操作完成：

（1）写出所有信号量及其初值。（4分）

mutex 1 缓冲区互斥访问 full 0 满缓冲区数

empty N 空缓冲区数

（2）写出生产者 producer 和消费者 consumer 的完整代码。（10分）

semaphore empty full mutex

empty:=N full:=0 mutex:=1

cobegin

process producer

begin

L1: 生产一个产品

P(empty) //同步P：当可供

P(mutex) //互斥P：等用缓

将产品放到缓冲区

V(mutex)

V(full)

goto L1

end

process consumer

begin

L2: P(null)

P(mutex)

从缓冲区拿产品

V(mutex)

V(empty)

消费产品

goto L2

end;

coend;

（3）若把生产者代码中的 P(empty) 和 P(mutex) 顺序颠倒，会出现什么问题？请简要说明。（2分）

会死锁，缓冲区若已满

4.（16分）读者-写者问题（必考！）

多个读者和写者并发访问同一个共享文件 F，要求：

① 允许多个读者同时读文件；

② 任一时刻只能有一个写者写文件；

③ 写者写文件时，不允许其他读者或写者访问。

请用信号量和 PV 操作完成：

（1）写出所有信号量、变量及其初值，并说明每个的含义。（4分）

W 1 控制读写互斥
R 1 控制rc互斥访问
rc 0 计数器

（2）写出读者 read 和写者 write 的完整代码。（10分）

（3）为什么在读者代码中，"rc = rc + 1" 这一行要用 P(R) 和 V(R) 包起来？（2分）

rc是多个读者共享的变量

(2)

```
var rc: integer:=0
    W,R: semaphores
    W:=1  R:=1
cobegin
  procedure read
    begin
      P(R)
      rc:=rc+1
      if rc=1 then P(W)
      V(R)
      读文件
      P(R)
      rc:=rc-1
      if rc=0 then V(W)
      V(R)
    end
coend
```

```
procedure write
  begin
    P(W)
    写文件
    V(W)
  end;
```

① 最简单的互斥（PV同一进程）

```
semaphore mutex=1; (初值为1)
Cobegin
    process Pi:
     begin
       ⋮
      P(mutex);
      临界区;
      V(mutex);
       ⋮
     end;
Coend;
```

② 最简单的同步（2个进程）

```
semaphore S=0 (初值为0!!!)
cobegin
 process P1:                process P2:
    begin                      begin
      ⋮                          P(S) 当P作为
    操作A                        操作B 前A之后
    V(S) 发作为                    ⋮
    end;                       end;
Coend;
```

生产者—消费者

```
semaphore empty=K;
semaphore full =0      } 同步
semaphore mutex=1 互斥
  Cobegin
    process Producer:
     begin
    L1: 生产一个商品;
      P(empty);
      P(mutex);
       B[in]=product;
       in=(in+1)%K;      } 放入商品
      V(mutex);
      V(full);
    goto L1;
   end;
  Coend;
```

```
process: Consumen;
  begin
L2: P(full)
      P(mutex)
      product=B[out]
      out=(out+1)mod K   } 取出商品
      V(mutex);
      V(empty);
    消费商品;
    goto L2;
  end;
```

读者写者

① 读者优先 (只要有一个读者在读，读者都可以加入 写者与读者都隔开)

```
semaphore rw =1;      读写互斥
semaphore mutex =1    保护 readcount
int readcount =0
cobegin
  Reader:
  begin
     P(mutex);              //保护 readcount   ← 出现加 mutex
     if(readcount==0)       //我是第一个读者
       P(rw);               //替全体读者抢占 rw，挡写者
     readcount ++;
     V(mutex);

      读文件

     P(mutex);
     readcount --;
     if(readcount==0)       //我是最后一个读者
       V(rw);               //释放 rw 让写者进
       V(mutex);
   end;
  coend;

  Writer:
  begin;
    P(rw)
    写文件
    V(rw)
  end
```

② 写者优先 (加一个信号量 w：写者来了先占住它，挡住读者)

写优 中实际最外层多Pw

```
semaphore rw=1;
semaphore mutex=1;
semaphore w=1;
int readcount=0;
cobegin
  Reader:
    begin;
    ┌ P(w)             ☆ 多一层W 最外 其余一样
    │ P(mutex)
    │ if(readcount==0)
    │   P(rw);
    │ readcount ++;
    │ V(mutex)
    └ V(w)
      读文件
      ⋮  ⋮ coend

  Writer:
    begin
    ┌ P(w)             也是多一层W 最外
    │  P(rw)
    │  写文件
    │  V(rw)
    └ V(w)
    end
```