操作系统 2021 课后应用题作业 2

姓名：_____ 学号：_____

提醒：直接在本文档填写解题答案，

提交作业的文件名命名规范为【学号_姓名_作业 2.doc】

题序	1 满分 8 分	2 满分 6 分	3 满分 8 分	4 满分 8 分	5 满分 6 分	6 满分 8 分	7 满分 16 分	8 满分 16	9 满分 16	10 满分 8	总分
分值											

1、假定磁盘有 200 个柱面，编号 0～199，当前移动臂位于 143 号柱面上，并刚刚完成 125 号柱面的服务请求。如果请求队列的先后顺序是：86，147，91，177，94，150，102，175，130；试问：为了完成上述请求，下列算法移动臂所移动的总量分别是多少？并给出移动臂移动的顺序。①先来先服务算法；②最短查找时间优先算法；③双向扫描算法；④电梯调度算法。

答：（12 分，每小题 3 分）

(1)先来先服务算法 FCFS 为 565，依次为 143-86-147-91-177-94-150-102-175-130。
(2)最短查找时间优先算法 SSTF 为 162，依次为 143-147-150-130-102-94-91-86-175-177。
(3)扫描算法 SCAN 为 169，依次为 143-147-150-175-177-199-130-102-94-91-86。
(4)电梯调度为 125(先向地址大的方向)，依次为 143-147-150-175-177-130-102-94-91-86。

2、有一个磁盘组共有 10 个盘面，每个盘面有 100 个磁道，每个磁道有 16 个扇区。若以扇区为分配单位，现问：(1)用位示图管理磁盘空间，则位示图占用多少空间?(2)若空白文件目录的每个目录项占 5 个字节，则什么时候空白文件目录大于位示图?

答：（6 分，每小题 3 分）

(1) 磁盘扇区总数为：10×16×100=16000 个，故位示图占用 16000/8=2000 字节。**（3 分）**
(2)已知空白文件目录的每个目录项占 5 个字节，而位示图占用 2000 字节，也就是说 2000 字节可容纳 400 个文件目录项。当空白文件目录>400 时，空白文件目录大于位示图。**（3 分）**

3、假设在 Unix 文件系统中，inode 节点中分别含有 10 个直接地址的索引和一、二、三级间接索引。若设每个盘块有 512B 大小，每个盘块中可存放 128 个盘块地址，则(1)一个 1MB 的文件占用多少间接盘块？(2)一个 25MB 的文件占用多少间接盘块？ **（8 分，每小题 4 分）**

答：

这个结果中不包含间接盘块存储盘块地址的存储开销。

直接块容量=10×512B/1024=5KB
一次间接容量=128×512B/1024=64KB
二次间接容量=128×128×512B/1024=64KB×128=8192KB
三次间接容量=128×128×128×512B/1024=64KB×128=8192KB×128=1048576KB

1MB 为 1024KB，1024KB-69KB=955KB，955×1024B/512B=1910 块，1MB 的文件分别占用 **1910** 个二次间接盘块。

25×1024KB-69-8192=17339KB，17339×1024B/512=34678 块，25MB 的文件分别占用 34678 个三次间接盘块和 128*128=**16384** 个二次间接盘块。

4、【基本概念】**（8 分，每小题 4 分）**

设有 n 个进程共享一个互斥段，如果：①每次只允许一个进程进入互斥段；②每次最多允许 m 个进程（m≤n）同时进入互斥段。

试问：以上两种情况下所采用的信号量初值是否相同？试给出信号量值的变化范围。

答：

所采用的互斥信号量初值不同。

(1) 互斥信号量初值为 1，变化范围为[-n+1, 1]。

当没有进程进入互斥段时，信号量值为 1；当有 1 个进程进入互斥段但没有进程等待进入互斥段时，信号量值为 0；当有 1 个进程进入互斥段且有一个进程等待进入互斥段时，信号量值为-1；最多可能有 n-1 个进程等待进入互斥段，故此时信号量的值应为-(n-1)也就是-n+1。

(2) 互斥信号量初值为 m，变化范围为[-n+m, m]。

当没有进程进入互斥段时，信号量值为 m；当有 1 个进程进入互斥段但没有进程等待进入互斥段时，信号量值为 m-1；当有 m 个进程进入互斥段且没有一个进程等待进入互斥段时，信号量值为 0；当有 m 个进程进入互斥段且有一个进程等待进入互斥段时，信号量值为-1；最多可能有 n-m 个进程等待进入互斥段，故此时信号量的值应为-(n-m)也就是-n+m。

5、【基本概念】**（6 分）**

有两个优先级相同的进程 P1 和 P2，其各自程序如下，信号量 S1 和 S2 的初值均 0。试问 P1、P2 并发执行后，x、y、z 的值各为多少？

P1() {	P2() {
y=1;	x=1;
y=y+3;	x=x+5;
V(S1);	P(S1);
z=y+1;	x=x+y;
P(S2);	V(S2);
y=z+y;	z=z+x;
}	}

答：

现对进程语句进行编号，以方便描述。

y=1;	①	x=1;	⑤
y=y+3;	②	x=x+5;	⑥
V(S1);		P(S1);	
z=y+1;	③	x=x+y;	⑦
P(S2);		V(S2);	
y=z+y;	④	z=z+x;	⑧

①、②、⑤和⑥是不相交语句，可以任何次序交错执行，而结果是唯一的。接着无论系统如何调度进程并发执行，当执行到语句⑦时，可以得到 x=10，y=4。按 Bernstein 条件，语句③的执行结果不受语句⑦的影响，故语句③执行后得到 z=5。最后，语句④和⑧并发执行，这时得到了两种结果为：

语句④先执行：x=10，y=9，z=15。

语句⑧先执行：x=10，y=19，z=15。

此外，还有第三种情况，语句③被推迟，直至语句⑧后再执行，于是依次执行以下三个语句：　z=z+x;

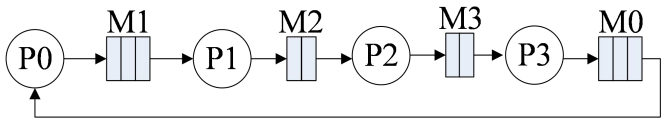
z=y+1;

y=z+y;

这时 z 的值只可能是 y+1=5，故 y=z+y=5+4=9，而 x=10。

第三种情况为：x=10，y=9，z=5。

6、【PV】四个进程 Pi（i=0…3）和四个信箱 Mj（j=0…3），进程间借助相邻信箱传递消息，即 Pi 每次从 Mi 中取一条消息，经加工后送入 M(i+1)mod4，其中 M0、M1、M2、M3 分别可存放 3、3、2、2 个消息。初始状态下，M0 装了三条消息，其余为空。试以 P、V 操作为工具，写出 Pi（i=0…3）的同步工作算法。



答：（8 分）

```
semaphore mutex1,mutex2,mutex3,mutex0;
    mutex1=mutex2=mutex3=mutex0=1;
    semaphore empty0,empty1,empty2,empty3;
    empty0=0;empty1=3;empty2=2;empty3=2;
    semaphore full0,full1,full2,full3;
    full0=3;full1=full2=full3=0; // M0 装了三条消息，其余为空。此处赋初值 full0=3，其余为 0。
    int in0,in1,in2,in3,out0,out1,out2,out3;
    in0=in1=in2=in3=out0=out1=out2=out3=0;
cobegin
```

process P0( ) {
while(true) {
P(full0);
P(mutex0);
{从 M0[out0]取一条消息};
out0=(out0+1) % 3;
V(mutex0);
V(empty0);
{加工消息};
P(empty1);
P(mutex1);
{消息存 M1[in1]};
in1=(in1+1) % 3;
V(mutex1);
V(full1);
}
}
	process P1( ) {
	while(true) {
	P(full1);
	P(mutex1);
	{从 M1[out1]取一条消息};
	out1=(out1+1) % 3;
	V(mutex1);
	V(empty1);
	{加工消息};
	P(empty2);
	P(mutex2);
	{消息存 M2[in2]};
	in2=(in2+1) % 2;
	V(mutex2);
	V(full2);
	}
	}

process P2( ) {
while(true) {
P(full2);
P(mutex2);
{从 M2[out2]取一条消息};
out2=(out2+1) % 2;
V(mutex2);
V(empty2);
加工消息;
P(empty3);
P(mutex3);
{消息存 M3[in3]};
in3=(in3+1) % 2;
V(mutex3);
V(full3);
}
}
	process P3( ) {
	while(true) {
	P(full3);
	P(mutex3);
	{从 M3[out3]取一条消息};
	out3=(out3+1) % 2;
	V(mutex3);
	V(empty3);
	{加工消息};
	P(empty0);
	P(mutex0);
	{消息存 M0[in0]};
	in0=(in0+1) % 3;
	V(mutex0);
	V(full0);
	}
	}

```
coend
```

7、【PV、管程】有一个阅览室，读者进入时必须先在一张登记表上登记，此表为每个座位列出一个表目，包括座位号、姓名，读者离开时要注销登记信息；假如阅览室共有 100 个座位。试用：①信号量和 PV 操作；②管程，实现用户进程的同步算法。

（满分 16 分，每小题 8 分，即 PV 题 8 分，管程 8 分）

答：(1) 使用信号量和 P、V 操作：**（8 分）**

```
struct {char name[10] ;
        int number;
    } A[100];
    semaphore mutex,seatcount;
    int i; mutex=1; seatcount=100;
  for(int i=0;i<100;i++ )
    {A[i].number=i;A[i].name=null;}
cobegin
process readeri(char readername[ ]) {   //(i=1,2,…)
        P(seatcount);
        P(mutex);
          for (int i=1; i< 100 ;i++)
            if (A[i].name==null ) A[i].name=readername;
            reader get the seat number =i;   /*A[i].number*/
        V(mutex)
        {进入阅览室，座位号 i，座下读书};
        P(mutex);
          A[i].name=null;
        V(mutex);
        V(seatcount);
        离开阅览室;
      }
coend.
```

(2) 使用管程实现：**（8 分）**

```
type readbook=MONTOR {
  semaphore R;
   int R_count, i, seatcount;
   char name[100] ;
   seatcount=0;
  InterfaceModule IM;
  DEFINE readbook( ),readerleave( );
  USE enter(), leave(),wait( ), signal( );
```

```
void readercome(char readername[ ]) {
      enter (IM);
        if (seatcount>=100) wait(R,R_count,IM);
        seatcount=seatcount+1;
        for (int i=0;i<100;i++) {
         if (name[i]==null ) name[i]=readername;
        }
        get the seat number=i;
      leave(IM );
  }
```

```
void readerleave(char readername) {
    enter(IM);
      seatcount--;
      for(int i=0;i<100;i++)
          if (name[i]==readername) name[i]=null;
      signal(R,R_count,IM);
    leave(IM);
}
```

```
cobegin
    process reader i ( ) {          //i=1,2….
        readbook.readercome(readername);
        read the book;
        readbook.readerleave(readername);
        leave the readroom;
      }
coend
```

8、【PV、管程】在一个盒子里，混装了数量相等的黑白围棋子。现在用自动分拣系统把黑子、白子分开，设分拣系统有二个进程 P1 和 P2，其中 P1 拣白子；P2 拣黑子。规定每个进程每次拣一子；当一个进程在拣时，不允许另一个进程去拣；当一个进程拣了一子时，必须让另一个进程去拣。试分别**使用 PV 操作和管程方法**写出两进程 P1 和 P2 能并发正确执行的程序。**（满分 16 分，每小题 8 分）**

答：（1）PV 操作：实质上是两个进程的同步问题，设信号量 S1 和 S2 分别表示可拣白子和黑子，不失一般性，若令先拣白子。**（8 分）**

```
semaphore S1,S2;
S1=1;S2=0;
cobegin
```

process P1( ) {	process P2( ) {
while(true) {	while(true) {
P(S1);	P(S2);
拣白子	拣黑子
V(S2);	V(S1);
}	}
}	}

```
coend
```

（2）管程方法：（8 分）

```
type pickup_chess= MONITOR    {
    bool flag; flag=true;
    semaphore S_black,S_white;
    int S_black_count,S_white_count;
    InterfaceModule IM;
   DEFINE pickup_black ,pickup_white
   USE enter,leave,wait,signal;
```

void pickup_black ( ) {	void pickup_white( ) {
enter(IM);	enter(IM);
if (flag) wait(S_black,S_black_count,IM);	if(!flag) wait(S_white,S_white_count,IM);
flag=true;	flag=false;
pickup a black;	pickup a white;
signal(S_white,S_white_count,IM);	signal(S_black,S_black_count,IM);
leave(IM);	leave(IM);
}	}

```
cobegin
    process_B( ); process_W( );
coend
   process_B( ) {
            pickup_chess.pickup_black( );
            other;
        }
   process_W( ) {
            pickup_chess.pickup_white( );
            other;
    }
```

9、【PV、管程】一组生产者进程和一组消费者进程共享 9 个缓冲区，每个缓冲区可以存放一个整数。生产者进程每次一次性地向 3 个缓冲区中写入整数，消费者进程每次从缓冲区取出一个整数。请用：①信号量和 PV 操作；②管程，写出能够正确执行的程序。**（满分 16 分，每小题 8 分）**

答：(1)信号量和 P、V 操作：**（8 分）**

```
var   int buf[9];
       int count,getptr,putptr;
       count=0;getpt=0;putpt=0;
       semaphoreS1,S2,SPUT,SGET;
       S1=1;S2=1;SPUT=3;SGET=0;
  main( ) {
    cobegin
      producer_i();consumer_j();
    coend
 }
```

```
process producer_i( ) {
        while(true) {
          {生产 3 个整数};
           P(SPUT);
           P(S1);
              buf[putptr]=整数 1;
              putptr=(putptr+1) % 9;
              buf[putptr]=整数 2;
              putptr=(putptr+1) % 9;
              buf[putptr]=整数 3;
              putptr=(putptr+1) % 9;
           V(S1);
           V(SGET);
           V(SGET);
           V(SGET):
       }
   }
```

```
process consumer_j( ) {
        int y;
        while(true) {
          P(SGET);
          P(S2);
            y=buf[getptr];
            getptr=(getptr+1) % 9;
            count++;
            if (count==3)
                {count=0;V(SPUT);}
        V(S2);
          {consume the  整数 y};
          }
    }
```

(2) 使用管程实现：生产者消费者。**（8 分）**

```
TYPE    get_put = MONITOR
          int buf [9];
          int count,getptr,putptr;
          semaphore SP,SG;
          int SP_count,SG_count;
        count:=0;getptr:=0;putptr:=0;
InterfaceModule IM;
DEFINE put, get;
USE wait, signal, enter, leave;
```

```
procedure put(int a1,int a2,int a3) {
   enter(IM);
    if (count>6) wait(SP,SP_count,IM);
   count=count+3;
   buf[putptr]=a1;
   putptr=(putptr+1) % 9;
   buf[putptr]=a2;
   putptr=(putptr+1) % 9;
   buf[putptr]=a3;
   putptr=(putptr+1) % 9;
      signal(SG,SG_count,IM);
      signal(SG,SG_count,IM);
      signal(SG,SG_count,IM);
  leave(IM);
}
```

```
procedure get(int b) {
     enter(IM);
      if (count= =0) wait(SG,SG_count,IM);
      b=buf[getptr];
      getptr=(getptr+1) % 9;
      count--;
      if (count < 7) signal(SP,SP_count, IM);
      else if (count > 0) signal(SG,SG_count,IM);
     leave(IM);
  }
```

```
cobegin
   process producer_i( ) {
```

```
        while(true) {
        {生产 3 个整数};
       get-put.put(a1,a2,a3);
        }
    }
    process consumer_j( ) {
        while(true) {
        get-put.get(b)
        {consume the  整数 b};
        }
    }
coend
```

10、【银行家算法】系统有 A、B、C、D 共 4 种资源，在某时刻进程 P0、P1、P2、P3 和 P4 对资源的占有和需求情况如表，试解答下列问题：**（满分 8 分，4+4）**

Process	Allocation				Claim				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	3	2	0	0	4	4	1	6	2	2
P_1	1	0	0	0	2	7	5	0				
P_2	1	3	5	4	3	6	10	10				
P_3	0	3	3	2	0	9	8	4				
P_4	0	0	1	4	0	6	6	10				

(1)系统此时处于安全状态吗？试给出一个可能的安全序列。**（4 分）**

(2)若此时进程 P2 发出 request1(1, 2, 2, 2)，系统能分配资源给它吗？为什么？ **（4 分）**

答：

(1)系统处于安全状态，存在安全序列：P0，P3，P4，P1，P2。

	CurrentAvail				C_{ki}-A_{ki}				Allocation				CurrentAvail+allocation				Possible
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D	
P0	1	6	2	2	0	0	1	2	0	0	3	2	1	6	5	4	True
P3	1	6	5	4	0	6	5	2	0	3	3	2	1	9	8	6	True
P4	1	9	8	6	0	6	5	6	0	0	1	4	1	9	9	10	True
P1	1	9	9	10	1	7	5	0	1	0	0	0	2	9	9	10	True
P2	2	9	9	10	2	3	5	6	1	3	5	4	3	12	14	14	True

(2)不能分配，否则系统会处于不安全状态。

若执行 P2 发出的 request1(1, 2, 2, 2)，此时 Available=(0, 4, 0, 0)，不能满足后续分配。