**操作系统2020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** | **8**  **满分16** | **8**  **满分8** | **总分** |
| **分值** |  |  |  |  |  |  |  |  |  |  |  |

1、假定磁盘有200个柱面，编号0~199，当前存取臂的位置在143号柱面上，并刚刚完成了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：为完成上述请求，下列算法（FCFS、SSTF、SCAN、电梯调度算法）存取臂移动的总量是多少？并算出存取臂移动的顺序。

**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。故位示意图占用16000 / 8 = 2000字节。
2. 已知空白文件目录的每个目录项占5个字节，而位示意图占用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意图。

3、在一个操作系统中，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**：

在这样的操作系统中，10个直接盘块存放的容量为：512B \* 10 / 1024 = 5KB

一次间接索引盘块存放的内容为：512B \* 128 / 1024 = 64KB

二次间接索引盘块存放的内容为：512B \* 128 \* 128 / 1024 = 8192KB

三次间接索引盘块存放的内容为：512B \* 128 \* 128 \* 128 / 1024 = 1048576KB

1. 则1MB为1024KB，1024KB - 64KB - 5KB = 955KB，955 \* 1024B / 512B = 1910，所以1MB文件分别占用128个一次间接盘块和1910个二次间接盘块。
2. 25MB为25 \* 1024KB - 64KB - 5KB - 8192KB = 17339KB，17339 \* 1024B / 512B = 34678，所以25MB文件分别占用128个一次间接盘块和128 ^ 2 = 16384个二次间接盘块，34678个三次间接盘块。

4、【基本概念】设有n个进程共享一个互斥段，如果：**（8分，每小题4分）**

(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：所采用的信号量初值是否相同？信号量值的变化范围如何？

**答：**

1. 互斥信号量初始值为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】有两个优先级相同的进程P1和P2，各自执行的操作如下，信号量S1和S2初值均为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**：**（6分）**

对进程语句进行编号，方便描述：

P1( ) { P2( ) {

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

} }

①②⑤⑥是不相交语句，可以任意交错次序执行，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，还有第三种状况，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z = y + 1 = 5，故y = z + y = 9，而x = 10。

第三种情况：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）**

大致思路是：先从自己的信箱里取出一个（P操作）然后加工处理完，通知上一个人可以投递了（V）。然后申请投递下一家的信箱（P操作），允许后信息加入该队列，并增加下家信箱中已有信息的个数（V操作）

Sput0:=0 sput1:=3 sput2:=2 sput3:=2 // Mi中目前可放的信息数

sget0:=3 sget1:=0 sget2:=0 sget3:=0//Mi中目前已有的信息数

M0,M1,M2,M3分别代表信箱

P0:

While(true){

P(sget0);

Message0:=M0.getMessage();

Produce message0;

V(sput0);

P(sput1);

M1.add(messgae0);

V(sget1);

}

P1:

While(true){

P(sget1);

Message1:=M1.getMessage();

Produce message1;

V(sput1);

P(sput2);

M2.add(message1);

V(sget2);

}

P2:

While(true){

P(sget2);

Message2:=M2.getMessage();

Produce message2;

V(sput2);

P(sput3);

M3.add(message2);

V(sget3)

}

P3:

While(true){

P(sget3);

Message3:=M3.getMessage();

Produce message3;

V(sput3);

P(sput0);

M0.add(message3);

V(sget0);

}

7、【PV、管程】有一阅览室，读者进入时必须先在一张登记表上登记，该表为每一座位列出一个表目，包括座号、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P、V操作；(2）管程，来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）

PV方式

Semaphore person = 1;

Semaphore seats =100;

Struct table{

string name;

int seat\_num;

}

table[100] t;

Int pos = 0;

Begin:

P(seats);

P(person);

//登记

t[i] =table(name,seat\_num);

i++

V(person);

//自习

P(person);

//消除座位、姓名

V(person);

V(seats);

End

（2）

管程方式

typedef lib = monitor{

string seats[100];

semaphore mutex

int mutex\_count;

int inside\_num;

mutex=0;mutex\_count = 0; inside\_num = 0;

define come(string name),out(string name)

use wait(),enter(),signal(),leave()

void come(string name){

enter(IM);

if(inside\_num>=100){wait(mutex,mutex\_count,IM)}

inside\_num ++;

for(int i = 0;i<100;i++){

if(seat.isEmpty()){

seat[i] = name;

}

}

leave(IM);

}

void out (string name){

enter(IM);

inside\_num --;

for(int i = 0;i<100;i++){

if(seat[i]==name){

seat[i] = null;

}

}

signal(mutex,mutex\_count,IM);

leave(IM);

}

};

begin

reader i(){

lib.come(name);

//读书

lib.out(name);

}

end

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方法

int white\_num = 0;//捡子个数

int black\_num = 0;//

Semaphore white = 1;

Semaphore black = 0;

Semaphore s = 1;

黑

P(black)

P(s)

black\_num++;

V(s)

V(white)

白

P(white)

P(s)

white\_num++;

V(s)

V(black)

（2）

管程方法

typedef chess = monitor{

semaphore w,b

int w\_count,b\_count;

w=0,b=0;w\_count = 0;

black\_count = 0;

define white(),black()

use wait(),enter(),signal(),leave()

void white(){

enter(IM);

//捡

signal(b,black\_count,IM);

wait(w,w\_count,IM);

leave(IM);

}

void black(){

enter(IM);

wait(b,b\_count,IM);

//捡

signal(w,wlack\_count,IM);

leave(IM);

}

}

白：

while(true){

chess.white();

}

黑：

while(true){

chess.black();

}

9、【PV、管程】一组生产者进程和一组消费者进程共享九个缓冲区，每个缓冲区可以存放一个整数。生产者进程每次一次性向3个缓冲区写入整数，消费者进程每次从缓冲区取出一个整数。请用：(1)信号量和P、V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）

PV方法

int full\_num = 0;//用于识别还有几个位置是空的

mutex.value = 0;

mutex mutex\_list[9];

semaphore send.s = 1;

semaphore read.s = 1;

semaphore ex.s = 1;

semaphore S.s =1;

Sender:

P(send);

if(full\_num<=6){

P(ex);

int count = 0;

for(int i = 0;i<9;i++){

if( count==3){break;}

else if(mutex\_list[i].value == 0){

//写入消息

count++;

full\_num++;

}

}

V(ex);

}

V(send);

Recevier:

P(read);

if(full\_num>0){

for(int i = 0;i<9;i++){

if(mutex\_list[i].value == 1){

//读出消息

full\_num--;

break;

}

}

}

V(read);

（2）

管程

typedef prod= monitor{

int pos[9]

semaphore p,s

int free\_num;

int p\_count,s\_count;

p = 0;s = 0;p\_count=0;s\_count=0;free\_num =0;

define produce(),consume();

use wait(),leave(),enter(),signal()

void produce(int a,int b,int c){

enter(IM);

if(free\_num<3){wait(p,p\_count,IM);}

int count = 0;

for(int i = 0;i<9;i++){

if(pos[i] == null){

switch(count){

case0:pos[i]=a;

break;

case1:pos[i]=b;

break;

case2:pos[i]=c;

break;

}

count++;

}

if(count==3){count=0;

break;}

}

signal(s,s\_count,IM)

leave(IM);

}

void consume(i){

enter(IM);

if(free\_num==9){wait(s,s\_count,IM);}

for(int i = 0;i<9;i++){

if(pos[i]!=null){message=pos[i];break;}

}

if(free\_num>=3){signal(p,p\_count,IM)}

else if(free\_num>0){signal(s,s\_count,IM)}

leave(IM);

}

}

begin:

生产者:

while(true){

prod.produce(1,2,3);

}

消费者:

while(true){

prod.consume();

}

end

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 |
| P0 | 0 0 3 2 | 0 0 4 4 | 1 6 2 2 |
| P1 | 1 0 0 0 | 2 7 5 0 |  |
| P2 | 1 3 5 4 | 3 6 10 10 |  |
| P3 | 0 3 3 2 | 0 9 8 4 |  |
| P4 | 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. 首先满足P0，归还后是1 6 5 4;然后满足P3，归还后1 9 8 6，然后满足P4，归还后1 9 9 10，然后满足P1，满足2 9 9 10，然后满足P2.能顺序执行完，故是安全的
2. 当P2申请1 2 2 2后，P2的占有资源达到2 5 7 6，但不足以执行，剩余资源为0 4 0 0并不足以支持任意进程执行，导致死锁。故不安全，不能分配