**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）**

FSCS: 143-86-147-91-177-94-150-102-175-130

总量为57+61+56+86+83+59+48+73+45=565个移动

SSTF:143-147-150-130-102-94-91-86-175-177

总量为4+3+20+28+8+3+5+89+2=162个移动

SCAN:143-147-150-175-177-199-130-102-94-91-86

总量为4+3+25+2+22+69+28+8+3+5=169个移动

电梯：143-147-150-175-177-130-102-94-91-86

总量为4+3+25+2+47+28+8+3+5=125个移动

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）总共有10\*100\*16=16000个分配单位，用16000个bit，也就是2000个byte，2kB

（2）2000/5=400，所以有400个空白文件目录就大于位示图

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**：一个直接地址能存放512B，一个一级间接索引能使用512\*128=65536B=64kB，一个二级索引能存放64\*128=8MB的文件，一个三级索引能存放1GB的文件

1. 一个1MB的文件使用2048个数据盘块，需要1个二级间接盘块，一级盘块128个，二级用了1910个，共计2038个
2. 一个25MB的文件使用51200个数据盘块，用了51190个间接盘块，使用128个1级盘块，二级16384块，34678个三级盘块

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）初值：1，值域是[1-n, 1]

（2）初值：m，值域是[m-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）**

1. x=10, y=9, z=15
2. X=10, y=9, z=5
3. X=10, y=19, z=15

**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 letter0=3, letter1=letter2=letter3=0

SEMAPHORE space0=0, space1=3, space2=space3=2

SEMAPHORE mutex0=mutex1=mutex2=mutex3=1

Pi的工作算法：

while(True){

P(letter[i])

P(mutex[i])

// 取消息

V(mutex[i])

V(space[i])

// 加工消息

P(space[(i+1)%4])

P(mutex[(i+1)%4])

// 放消息

V(mutex[(i+1)%4])

V(letter[(i+1)%4])

}

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. SEMAPHORE mutex=1, seats=100

While(True){

P(seats)

P(mutex)

// 登记座号，姓名

V(mutex)

// 图书馆阅读

P(mutex)

// 注销登记信息

V(mutex)

V(seats)

}

1. Type library = monitor{

InterfaceModule IM

SEMAPHORE login\_mutex, empty\_mutex

Int login\_count, empty\_count

DEFINE comein, comeout

USE enter, leave, wait, signal

}

Procedure comein(){

Enter(IM)

Wait(empty\_mutex, empty\_count, IM)

Wait(login\_mutex, login\_count, IM)

// login

Signal(login\_mutex, login\_count, IM)

Leave(IM)

}

Procedure comeout(){

Enter(IM)

Wait(login\_mutex, login\_count, IM)

// log out

Signal(login\_mutex, login\_count, IM)

SIgnal(empty\_mutex, empty\_count, IM)

Leave(IM)

}

Process student(){

Library.comein()

// reading

Library.comeout()

}

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）**

(1) PV操作

SEMAPHORE pick\_white = 1, pick\_black = 0

P1(){

While(True){

P(pick\_white)

// pickup white chess

V(pick\_black)

}

}

P2(){

While(True){

P(pick\_black)

// pickup black chess

V(pick\_white)

}

}

1. 管程

Type pick\_chess = monitor{

SEMAPHORE pick\_white, pick\_black

Int white\_count, black\_count

interfaceModule IM

DEFINE white\_pick, black\_pick,white\_start, black\_start

USE enter, leave, wait, signal

}

Procedure white\_pick(){

Enter(IM)

Wait(white\_count, pick\_white, IM)

Leave(IM)

}

Procedure white\_start(){

Enter(IM)

Signal(white\_count, pick\_white, IM)

Leave(IM)

}

Procedure black\_pick(){

Enter(IM)

Wait(black\_count, pick\_black, IM)

Leave(IM)

}

Procedure black\_start(){

Enter(IM)

Signal(black\_count, pick\_black, IM)

Leave(IM)

}

Procedure white(){

While(True){

Pick\_chess.White\_pick();

// pick white chess

Pick\_chess.Black\_start();

}

}

Procedure black(){

While(True){

Pick\_chess.Black\_start()

// pick black chess

Pick\_chess.White\_start()

}

}

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)

SEMAPHORE mutex=1, put=3, get=0

Producer(){

While(True){

P(put)

P(mutex)

For i in range(3)

// select a buffer

// put data into the selected buffer

V(get)

V(get)

V(get)

V(mutex)

}

}

Consumer(){

While(True){

P(get)

P(mutex)

// get data from the buffer

If ( consumed 3 num) V(put)

V(mutex)

// consume

}

}

1. 管程

Type prod\_cons = monitor{

SEMAPHORE empty, full

Int full\_count, empty\_count

InterfaceModule IM

Define cons, prod

USE enter, leave, wait, signal

}

Procedure cons(){

Enter(IM)

If(full\_count == 0) wait(full\_count, full, IM)

Full\_count--

Signal(empty, empty\_count, IM)

Leave(IM)

}

Procedure prod(){

Enter(IM)

For i in range(3)

// select a specific buffer

If(full\_count == 9) wait(empty\_count, empty, IM)

Full\_count++

Signal(full\_count, full, IM)

Leave(IM)

}

Procedure producer(){

// produce

Prod\_cons.prod()

}

Procedure consumer(){

Prod\_cons.cons()

// consume

}

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）**

|  |  |
| --- | --- |
| Process | need |
| P0 | 0 0 1 2 |
| P1 | 1 7 5 0 |
| P2 | 2 3 5 6 |
| P3 | 0 6 5 2 |
| P4 | 0 6 5 6 |

发现存在如下的安全序列：

（P0， P3， P1， P2， P4）(P0，P3，P4，P2，P1)等，所以处于安全状态

**（2）**如果分配了，Available变为0 4 0 0, 没有进程可以分配，不存在安全序列，不能分配