# 4.简答题

1、设内存中有3道程序A、B、C，它们按A、B、C的优先次序执行。它们的计算和I/O操作的时间见下表（单位：ms）。假设3道程序使用相同设备进行I/O操作，即程序以串行方式使用设备，试画出单道运行和多道运行的时间关系图（调度程序的执行时间忽略不计），并回答在这两种情况下，完成这3道程序分别需要多长时间（多道运行时采用抢占式调度策略）。

|  |  |  |  |
| --- | --- | --- | --- |
| 操作 | 程序 | | |
| A | B | C |
| 计算 | 30 | 60 | 20 |
| I/O操作 | 40 | 30 | 40 |
| 计算 | 10 | 10 | 20 |

答案：

（1）单道运行情况的时间关系图如下。



（评分标准：A、B、C三个程序的先后顺序画的对，得1分；A、B、C三个程序在横轴的时间标注正确，得1分）。

完成这3道程序的运行时间为260ms。（1分）

（2）多道运行情况的时间关系图如下。



（评分标准：A、B、C三个程序的先后顺序画的对，得1分；A、B、C三个程序在横轴的时间标注正确，得1分）。

完成这3道程序的运行时间为190ms。（1分）

2、假设内存中有三道程序A，B，C，它们按A-B-C的先后次序执行，它们进行“计算”和“I/O操作”的时间如下表所示，假设三道程序使用相同的I/O设备。

|  |  |  |  |
| --- | --- | --- | --- |
| 程序 | 操作 | | |
| 计算 | I/O操作 | 计算 |
| A | 20 | 30 | 10 |
| B | 30 | 50 | 20 |
| C | 10 | 20 | 10 |

(1)试画出单道运行时三道程序的时间关系图，并计算完成三道程序要花多少时间。

(2)试画出多道运行时三道程序的时间关系图，并计算完成三道程序要花多少时间。

答案：

（1）单道运行情况的时间关系图如下。



（评分标准：A、B、C三个程序的先后顺序画的对，得1分；A、B、C三个程序在横轴的时间标注正确，得1分）。

完成这3道程序的运行时间为200。（1分）

（2）多道运行情况的时间关系图如下。



（评分标准：A、B、C三个程序的先后顺序画的对，得1分；A、B、C三个程序在横轴的时间标注正确，得1分）。

完成这3道程序的运行时间为130。（1分）

3、设内存中有3道程序A、B、C，它们按A、B、C的优先次序执行。它们的计算和I/O操作的时间见下表。假设3道程序使用相同设备进行I/O操作，即程序以串行方式使用设备。

试画出单道运行和多道运行的时间关系图（调度程序的执行时间忽略不计），并回答在这两种情况下，完成这3道程序分别需要多长时间。

|  |  |  |  |
| --- | --- | --- | --- |
|  | 输入I | 计算C | 输出O |
| A | 20 | 10 | 10 |
| B | 20 | 30 | 10 |
| C | 20 | 30 | 20 |

答案：

（1）单道运行情况的时间关系图如下。



（评分标准：A、B、C三个程序的先后顺序画的对，得1分；A、B、C三个程序在横轴的时间标注正确，得1分）。

完成这3道程序的运行时间为170。（1分）

（2）多道运行情况的时间关系图如下。



（评分标准：A、B、C三个程序的先后顺序画的对，得1分；A、B、C三个程序在横轴的时间标注正确，得1分）。

完成这3道程序的运行时间为120。（1分）

4、设有如下3个程序段S1~S3，使用Beinstein条件判断哪些程序段可并发执行，哪些不可以并发执行。

S1:a=x+y；

S2:b=z+1；

S3:c=a-b；

参考答案：

3个程序段S1~S3的读集和写集分别如下：

R(S1)={x,y}，W(S1)={a}；

R(S2)={z}，W(S2)={b}；

R(S3)={a,b}，W(S3)={c}。

∵R(S1)∩W(S2)=Φ，W(S1)∩R(S2)=Φ，W(S1)∩W(S2)=Φ，

∴S1和S2可以并发执行；

∵R(S3)∩W(S1)={a}，

∴S1和S3不可以并发执行；

∵R(S3)∩W(S2)={b}，

∴S2和S3不可以并发执行。

5、设有如下3个程序段A、B、C，使用Beinstein条件判断哪些程序段可并发执行，哪些不可以并发执行。

A:a=x+y；

B:b=a^2；

C:c=4\*a+x/y

参考答案：

3个程序段A、B、C的读集和写集分别如下：

R(A)={x,y}，W(A)={a}；

R(B)= {a}，W(B)={b}；

R(C)={a,x,y}，W(C)={c}。

∵R(B)∩W(A)={a}，

∴A和B不可以并发执行；

∵R(C)∩W(A)= {a}，

∴A和C不可以并发执行。

∵R(B)∩W(C)=Φ，W(B)∩R(C)=Φ，W(B)∩W(C)=Φ，

∴B和C可以并发执行；

6、设有如下3个程序段S1~S3，使用Beinstein条件判断哪些程序段可并发执行，哪些不可以并发执行。

S1:x=a+2；

S2:y=b+3；

S3:z=x+y；

参考答案：

3个程序段S1~S3的读集和写集分别如下：

R(S1)={a}，W(S1)={x}；

R(S2)={b}，W(S2)={y}；

R(S3)={x,y}，W(S3)={z}。

∵R(S1)∩W(S2)=Φ，W(S1)∩R(S2)=Φ，W(S1)∩W(S2)=Φ，

∴S1和S2可以并发执行；

∵R(S3)∩W(S1)={x}，

∴S1和S3不可以并发执行；

∵R(S3)∩W(S2)={y}，

∴S2和S3不可以并发执行。

7、假定某页式管理系统中，主存为128KB，分成32块，块号为0,1,2,3,…，31；某程序逻辑地址空间共有5页，其页号为0,1,2,3,4，被分别装入主存的3,8,4,6,9物理块中。回答以下问题：

（1）该系统物理块的大小是多少？写出该程序每一页在内存的起始地址。

（2）写出分页式存储管理的地址变换过程

（3）逻辑地址为[3,70]（其中方括号中的第一个元素为页号，第二个元素为页内地址，按十进制计算）的物理地址是多少？

（4）逻辑地址9000的物理地址是多少？

参考答案：

（1）该系统物理块大小为4KB，第0页的起始地址为：12288；第1页起始地址：32768；第2页的起始地址：16384；第3页起始地址：24576；第4页起始地址：36864.

（2）根据逻辑地址结构，计算页号和页内地址，根据页号查页表，得到块号，依据：物理地址=块号\*块长+页内地址，完成物理地址。

（3）24576+70=24646

（4）16384+808=17192 （9000=8192+808）

8、设有一页式存储管理系统，向用户提供的逻辑地址空间最大为16页，每页2048B，内存总共有8个存储块。回答以下问题：

（1）该系统的逻辑地址至少为多少位？该系统物理内存空间有多大？说明原因

（2）写出分页式存储管理的地址变换过程。

（3）某程序逻辑地址空间共有3页，其页号为0、1、2，被分别装入主存的3、0、1号物理块中，请计算给出逻辑地址为3000、5000、7000的物理地址分别是多少？

参考答案：

（1）逻辑地址至少为15位，内存空间有16KB

（2）根据逻辑地址结构，计算页号和页内地址，根据页号查页表，得到块号，依据：物理地址=块号\*块长+页内地址，完成物理地址。

（3）3000： 0\*2048+（3000-2048）=952

5000： 1\*2048+（5000-4096）=2952

7000：越界

9、(2016）某进程的段表内容见下表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 段号 | 段长 | 内存起始地址 | 权限 | 状态 |
| 0 | 100 | 6000 | 只读 | 在内存 |
| 1 | 200 | 8000 | 读写 | 在内存 |
| 2 | 300 | 4000 | 读写 | 在内存 |

请回答以下问题：

（1）写出段式存储管理地址变换过程。

（2）当访问：段号1，段内地址120的逻辑地址时，地址转换的结果。

（3）当访问：段号2、段内地址为400的逻辑地址时，进行地址转换的结果。

参考答案：

（1）依据逻辑地址结构，计算出段号和段内地址，用段号查段表得到该段的起始地址，依据：物理地址=段起始地址+段内地址，得到物理地址。

（2）8000+120=8120

（3）越界

10、(2017）某计算机按字节编址，其动态分区内存管理采用最佳适应算法，每次分配和回收内存后都对空闲分区链重新排序。当前空闲分区信息见下表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分区起始地址 | 20K | 500K | 1000K | 200K |
| 分区大小 | 40KB | 80KB | 100KB | 200KB |

请回答以下问题：

（1）回收起始地址为60K、大小为140KB的分区后，系统中空闲分区的数量、空闲分区链第一个分区的起始地址和大小分别是多少。画出空闲分区表。

（2）依据初始空闲分区表信息，有进程A，大小为150K，为其分配内存后，该进程在内存的起始地址是多少？按上表样式，画出为进程A分配内存后的空闲分区表。

参考答案：

（1）3、500K、80KB

|  |  |  |  |
| --- | --- | --- | --- |
| 分区起始地址 | 500K | 1000K | 20K |
| 分区大小 | 80KB | 100KB | 380KB |

（2）该进程在内存的起始地址为150K

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 分区起始地址 | 20K | 350K | 500K | 1000K |
| 分区大小 | 40KB | 50KB | 80KB | 100KB |

11、.(2015年）某计算机系统按字节编址，采用二级页表的分页存储管理方式,虚拟地址格式如下所示：

|  |  |  |
| --- | --- | --- |
| 10位 | 10位 | 12位 |
| 页目录号 | 页表索引 | 页内偏移量 |

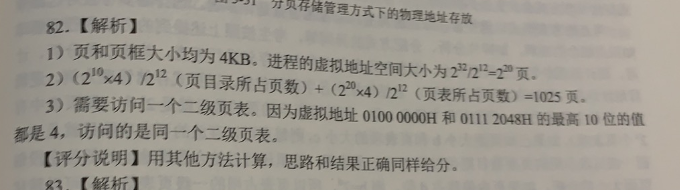
请回答下列问题。

1)页和块的大小各为多少字节？进程的虚拟地址空间大小为多少页？

2)假定页目录项和页表项均占4B,则进程的页目录和页表各占多少页？要求写出计算过程。

3)若某指令周期内访问的虚拟地址为0100 0000H和0111 2048H，则进行地址转换时共访问多少个二级页表？要求说明理由。

答案及解析：



12、某基于动态分区存储管理的计算机，其内存容量为55MB(初始为空)，分配和释放的顺序为：进程A分配15MB，进程B分配30MB，进程A释放15MB，进程C分配8MB，进程B分配6MB，请回答以下问题：

（1）若采用最坏适应算法此时内存中最大空闲分区的大小是多少？说明原因。

（2）采用最佳适应算法，则内存中最大空闲分区是多少？说明原因。

（3）采用最先适应，则内存中最大空闲分区大小是多少？说明原因。

参考答案：

（1）若采用最坏适应算法此时内存中最大空闲分区的大小是 7 M。

（2）采用最佳适应算法，则内存中最大空闲分区是 9 M；

（3）采用最先适应，则内存中最大空闲分区大小是 10 M。

13、（2013）某磁盘的转速为10000转/分，平均寻道时间是6ms，磁盘传输速率是20MB/s，磁盘控制器延迟为0.2ms。如果要读取一个4KB的扇区，请问

（1）查找扇区的平均时间是多少？

（2）读取该扇区的磁盘传输时间是多少？

（3）对该扇区进行磁盘访问的总时间由哪几部分构成？具体总时间是多少？

答案：

（1）10000转/分钟=6ms/周。（1分）。则平均查找扇区时间为3ms。（1分）。

（2）读取4KB扇区的时间为4KB/(20MB/s)=0.2ms。（计算过程1分，结果正确1分）

（3）磁盘访问总时间由寻道时间、旋转时间和传输时间三部分构成（1分），具体来说是6ms+3ms+0.2ms+0.2ms=9.4ms（1分）。

答案解析：

10000转/分钟=6ms/周，则平均查找扇区时间为3ms；而平均寻道时间为6ms；4KB读取时间：4KB/(20MB/s)=0.2ms；磁盘控制器延迟为0.2ms；总的时间为：9.4ms

14、某文件占10个磁盘块，现要把将文件磁盘块逐个读入内存缓冲区，并送用户区进行分析。假设一个缓冲区与一个磁盘块大小相同，将一个磁盘块读入缓冲区的时间为100μs，将缓冲区的数据传送到用户区的时间是50μs，CPU对一块数据进行分析的时间为50μs。

（1）单缓冲和双缓冲的根本区别是什么？哪种方式处理文件的速度更快？

（2）在单缓冲区结构下，除了第1个磁盘块以外的其他磁盘块，其实际运行时间是多少？读入并分析完该文件的时间是多少？

（3）在双缓冲区结构下，除了第1个磁盘块以外的其他磁盘块，其实际运行时间是多少？读入并分析完该文件的时间是多少？  
答案：

（1）单缓冲和双缓冲的根本区别是缓冲区的数量不同，单缓冲只有一个缓冲区，双缓冲有两个缓冲区。（1分）。因此，双缓冲处理文件的速度更快。（1分）。

（2）除了第1个磁盘块以外的其他磁盘块，其实际运行时间是（100+50）=150μs。（1分）。整个文件处理完的总时间是50+150\*10=1550μs。（1分）。

（3）除了第1个磁盘块以外的其他磁盘块，其实际运行时间是100μs。（1分）。整个文件处理完的总时间是100+100\*10=1100μs。（1分）。

15、若磁盘转速为7200转/分，平均寻道时间为8ms，每个磁道包含1000个扇区。

（1）磁盘访问时间由哪三部分构成？

（2）平均查找扇区时间是多久？（单位ms，小数点后保留2位有效数字）

（3）访问一个扇区的平均存取时间是多少？（单位ms，小数点后保留2位有效数字）

答案：

（1）磁盘访问时间由三部分构成，寻道时间，旋转延迟时间，传输时间。

（2）“磁盘转速为7200转/分”可以求得“每转一圈需要时间：8.33ms”，可以得到平均的查找扇区时间是4.17ms。

（3）12.18ms

答案解析：

本题考查磁盘访问时间的知识。

磁盘访问时间由三部分构成，寻道时间，旋转延迟时间，传输时间。

寻道时间已经给出，为8ms。

旋转延迟时间，是指定扇区移动到磁头下面所经历的时间，与磁盘的转速有关。并且，由于我们无法确定磁臂头到指定扇区的具体距离，通常需要计算的是平均时间。

通过，“磁盘转速为7200转/分”可以求得“每转一圈需要时间：8.33ms”，可以得到平均的查找扇区时间是4.17ms。 （60\*1000/7200）=8.33

传输时间可以计算得出，访问一个扇区的时间是转一圈时间的一千分之一，即8.33/1000=0.01ms。

则总时间是以上三项的和，12.18ms。

16、(2020 年统考真题）某文件系统的目录项由文件名和索引结点号构成。若每个目录项长度为64字节，其中4个字节存放索引结点号，6个字节存放文件名，文件名由小写英文字母构成。

（1）索引节点对应的位数是多少？

（2）在题目中，能够创建的文件数量与哪个因素有关？

（3）该文件系统能创建的文件数量的上限是多少？

答案：

（1）32位

（2）索引结点个数

（3）232个

答案解析：

最多创建文件个数=最多索引结点个数。由题目可知，索引结点占4个字节，对应32位，最多可以表示232个文件。

17、某文件系统物理结构采用三级索引分配方法，如果每个磁盘块的大小为 1024B，每个盘块索引号占用4B。

（1）在什么情况下，会使用二级甚至三级索引分配方法？

（2）每个磁盘块可以存放多少索引号？

（3）在该文件系统中，最大文件的大小是多少GB？（结果保留整数即可）

答案：

（1）当文件特别大，其索引块太多时，单级索引分配方式效率较低，因此要采用二级甚至三级索引分配。

（2）256个

（3）16GB

答案解析：

根据已知条件，每个盘块为1024B，每个索引号为4B，因此，每个索引块可以存放256个索引号，三级索引块可以管理文件的大小为：256×256×256×1024B≈16GB。

18、（2017 年统考真题）某文件系统中，针对每个文件，用户类别分为 4类：安全管理员、文件主、文件主的伙伴、其他用户；访问权限分为5类：完全控制、执行、修改、读取、写入。若文件控制块中用二进制位串表示文件权限，为表示不同类别用户对一个文件的访问权限。

（1）什么是用户权限表？

（2）用户权限表能否被用户直接访问？为什么？

（3）描述文件权限的位数至少应为多少位？

答案：

（1）用户权限表是由一个域对每个用户可以执行的一组操作所构成的，表中的每一项即为该域对某一对象的访问权限。

（2）不能。因为要保证用户权限表的安全性。

（3）20

答案解析：

可以把用户访问权限抽象成一个矩阵，行代表用户，列代表访问权限。这个矩阵有4行5列，1代表true，0代表false，所以需要20位，选D。

# 5.同步分析题

1、设有5个并发进程P1~P5中分别有语句A~E，它们的前趋关系如下图所示。采用信号量机制，分别设置信号量a、b、c、d、e，请写出实现前趋关系的并发程序。

参考答案：

a

b

c

d

e

A

C

E

B

D

main(){

semaphore a,b,c,d,e;

a.value=b.value=c.value=d.value=e.value=0; //信号量赋初值

cobegin

P1();P2();P3();P4();P5();

coend

}

P1(){...A;V(a);... }

P2(){...B;V(b);V(d);...}

P3(){...P(a);P(b);C;V(c);...}

P4(){...P(d);D;V(e);...}

P5(){...P(c);P(e);E;...}

2、设有5个并发进程P1~P5中分别有语句S1~S5，它们的前趋关系如下图所示。采用信号量机制，分别设置信号量a、b、c、d、e，请写出实现前趋关系的并发程序。

a

b

c

d

e

S1

S2

S5

S3

S4

参考答案：

main(){

semaphore a,b,c,d,e;

a.value=b.value=c.value=d.value=e.value=0; //信号量赋初值

cobegin

P1();P2();P3();P4();P5();

coend

}

P1(){...S1;V(a);V(b);... }

P2(){...P(a);S2;V(d);V(c)...}

P3(){...P(b);P(d);S3;V(e);...}

P4(){...P(c);S4;...}

P5(){...P(e);S5;...}

3、设有5个并发进程P1~P5中分别有语句S1~S5，它们的前趋关系如下图所示。采用信号量机制，分别设置信号量a、b、c、d、e，请写出实现前趋关系的并发程序。

a

b

c

d

e

S1

S2

S5

S3

S4

d

参考答案：

main(){

semaphore a,b,c,d,e;

a.value=b.value=c.value=d.value=e.value=0; //信号量赋初值 cobegin

P1();P2();P3();P4();P5();

coend

}

P1(){...S1;V(a);V(b);... }

P2(){...P(a);S2;V(c);...}

P3(){...P(b);S3;V(d);...}

P4(){...S4;V(e);...}

P5(){...P(c);P(d);P(e);S5;...}

4、哲学家用餐问题：五个哲学家共用一张圆桌，分别坐在周围的五张椅子上，在圆桌上有五个碗和五只筷子，他们的生活方式是交替地进行思考和进餐。平时，一个哲学家进行思考，饥饿时便试图取用其左右最靠近他的筷子，只有在他拿到两只筷子时才能进餐。进餐完毕继续思考。以下是用信号量机制解决该哲学家用餐问题的一种方法。

struct semaphore chopstick[5]=(1,1,1,1,1);

cobegin

void philosopher\_i(void) //i=0,1,2,3,4

{

while(true)

{

思考；

P(chopstick[i]);

P(chopstick[(i+1)%5];

进餐；

V(chopstick[i]);

V(chopstick[(i+1)%5];

}

}

coend

请回答以下问题：

（1）试分析以上算法在什么情况下会产生死锁？

（2）采用抽屉原理（鸽巢原理）改写以上算法，使其不发生死锁。

参考答案：

（1）以上算法中所有哲学家总是先拿左边的筷子，再拿右边的筷子，那么就有可能出现这样的情况，就是5个哲学家都拿起了左边的筷子，当他们想拿右边的筷子时，却因为筷子已经被别的哲学家拿去，而无法拿到。此时所有的哲学家都不能进餐，这就出现了死锁现象。

（2）采用抽屉原理（鸽巢原理）改写算法如下：

struct semaphore chopstick[5]=(1,1,1,1,1);

struct semaphore count=4;

cobegin

void philosopher\_i(void) /\*i=0,1,2,3,4

{

while(true)

{

思考；

P(count);

P(chopstick[i]);

P(chopstick[(i+1)%5];

进餐；

V(chopstick[i]);

V(chopstick[(i+1)%5];

V(count);

}

}

coend

5、读者-写者问题：有两种进程reader和writer，允许多个reader同时读一个对象，但不允许一个writer进程和其它writer或reader同时访问共享对象。使用信号量的P、V操作解决以上问题，定义信号量Wmutex，控制对数据库的访问，全局变量Rcount记录正在读（>0）和想要读(<0)的进程数，信号量Rmutex控制对Rcount的访问。部分代码如下：

type int semaphore;

semaphore Rmutex=【初值1】;

semaphore Wmutex=【初值2】;

int Rcount=【初值3】;

void reader(){

while(True){

P(Rmutex);

if(Rcount==0) P(Wmutex);

Rcount=Rcount+1;

V(Rmutex);

read\_datebase()

P(Rmutex);

Rcount=Rcount-1;

if(Rcount==0) V(Wmutex);

V(Rmutex);

... ;

}

}

void writer(){

while (True){

think-up-data();

【程序段】

}

}

请回答以下问题：

（1）代码中【初值1】、【初值2】和【初值3】应分别设置成什么值？

（2）编写进程writer()中【程序段】处缺少的代码。

（3）分析在该算法中，读者进程和写者进程哪个享有更高优先权，为什么？

参考答案：

（1）代码中【初值1】、【初值2】和【初值3】应分别设置成1、1、0。

（2）进程writer()中【程序段】处缺少的代码：

P(Wmutex);

writer-database();写数据对象

V(Wmutex);

（3）该算法中，读者进程享有更高优先权。因为如果第一个读者进程执行申请到信号量Wmutex，只能由最后一个读者释放信号量Wmutex，期间写者进程必须等待，由最后一个读者释放信号量Wmutex之后唤醒它。

6、生产者-消费者问题：多个生产者和多个消费者共用一个容量为N仓库，生产者向仓库中存放产品，而消费者从仓库中取出产品，如果仓库满了，说明产品过剩，生产者不继续生产，反之，如果仓库空了，消费者无法取货，只能等待。现给出信号量和全局变量定义及生产者进程代码。

semaphore mutex=1;

semaphore empty=n;

semaphore full=0;

int i,j;

ITEM buffer[n];

ITEM data\_p,data\_c;

void proceducer()

{

while(true)

{

produce an item in data\_p;

P(mutex);

P(empty);

buffer[i]= data\_p;

i=(i+1) % n;

V(mutex);

V(full);

}

}

请回答以下问题：

（1）分析生产者进程中资源信号量和互斥信号量的P操作顺序是否合理。

（2）编写消费者进程的代码。

参考答案：

（1）不合理，应该先对资源信号量执行P操作，再对互斥信号量执行P操作，因为一旦一个生产者申请到互斥信号量成功，再去申请资源信号量失败，就会阻塞到资源信号量的等待队列中，期间不会释放互斥信号量；而消费者进程因无法申请到互斥信号量而进入到互斥信号量的等待队列中，无法消费产品，就不会释放生产者所需的资源信号量，系统进入死锁状态。

（2）

void consumer()

{

while(True)

{

P(&full);

P(&mutex);

remove\_item(&item);

V(&mutex);

V(&empty);

consume\_item(item);

}

}

7、有一个报箱为A、B两人共同使用，每次只能装一份报纸。A订阅生活报，B订阅新晚报，投递员C、D分属生活报和新晚报报社，可以将A、B、C和D看成4个进程，用信号量及PV操作写出他们的同步执行程序。

参考答案：

semaphore s=1,life=0,night=0;

void main(){

cobegin

A();B();C();D();

coend

}

void A(void){

wait(life);

取生活报;

signal(s);

}

void B(void){

wait(night);

取新晚报;

signal(s);

}

void C(void){

wait(s);

投递生活报；

signal(life);

}

void D(void){

wait(s);

投递新晚报；

signal(night);

}

8、3个进程PA，PB、PC合作解决文件打印问题：PA将文件记录从磁盘读入主存的缓冲区1，每执行一次读一个记录；PB将缓冲区1的内容复制到缓冲区2，每执行一次复制一个记录；PC将缓冲区2的内容打印出来，每执行一次打印一个记录。缓冲区的大小等于一个记录大小。采用信号量机制（自行设计信号量），设计并发进程保证文件正确打印。

参考答案：

semaphore empty1=1,empty2=1,full1=0,full2=0;

void main(){

cobegin

PA();PB();PC();

coend

}

void PA(void){

while(TRUE){

从磁盘读一个记录;

wait(empty1);

将记录存入缓冲区1;

signal(full1);

}

}

void PB(void){

while(TRUE){

wait(full1);

从缓冲区1读取一个记录;

signal(empty1);

wait(empty2);

将记录存入缓冲区2;

signal(full2);

}

}

void PC(void){

while(TRUE){

wait(full2);

从缓冲区2中取出记录;

signal(empty2);

打印记录;

}

}

9、桌上有一个盘子，每次只能向其中放入一个水果。爸爸专向盘子中放苹果，妈妈专向盘子中放橘子，儿子专吃盘子中的橘子，女儿专吃盘子中的苹果。只有盘子为空时，爸爸或妈妈就可向盘子中放一个水果；仅当盘子中有自己需要的水果时，儿子或女儿可以从盘子中取出。可以将爸爸、妈妈、儿子和女儿看成4个不同的进程，用信号量及PV操作写出他们的同步执行程序。

参考答案：

semaphore plate=1,apple=0,orange=0;

void main(){

cobegin

dad();mum();son();daughter();

coend

}

void dad(void){

while(true){

prepare an apple;

wait(plate);

put the apple on the plate;

signal(apple);

}

}

void mum(void){

while(true){

prepare an orange;

wait(plate);

put the orange on the plate;

signal(orange);

}

}

void son(void){

while(true){

wait(orange);

take an orange from the plate；

signal(plate);

eat the orange;

}

}

void daughter(void){

while(true){

wait(apple);

take an apple from the plate；

signal(plate);

eat the apple;

}

}

# 6.算法分析题

1、设系统中有4个进程P0、P1、P2和P3，有4种类型的资源A、B、C和D，其中A资源的数量为3，B资源的数量为8，C资源的数量为9，D资源的数量为10，T0时刻系统状态如表所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 进程 | 已分配资源数 | | | | 最大资源需求量 | | | | 仍然需求资源数 | | | |
| A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 |  |  |  |  |
| P1 | 1 | 0 | 0 | 0 | 1 | 7 | 5 | 0 |  |  |  |  |
| P2 | 1 | 3 | 5 | 4 | 2 | 3 | 5 | 6 |  |  |  |  |
| P3 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 |  |  |  |  |

使用银行家算法回答下面的问题。

（1）计算每个进程还可能需要的资源，并填入表的“仍然需求资源数”栏目中。

（2）T0时刻系统是否处于安全状态？为什么？

参考答案：

（1）

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 进程 | 已分配资源数 | | | | 最大资源需求量 | | | | 仍然需求资源数 | | | |
| A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 | 0 | 0 | 0 | 0 |
| P1 | 1 | 0 | 0 | 0 | 1 | 7 | 5 | 0 | 0 | 7 | 5 | 0 |
| P2 | 1 | 3 | 5 | 4 | 2 | 3 | 5 | 6 | 1 | 0 | 0 | 2 |
| P3 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 | 0 | 6 | 4 | 2 |

（2）由已知条件，系统中A、B、C和D四类资源的总数为（3,8,9,10），剩余可用资源的数量是(1,5,2,0)，P0已经获得所需的全部资源，可以先执行，当进程P0执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(1,5,3,2)，可以满足进程P2的仍然需求资源数，让进程P2执行，当进程P2执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(2,8,8,6)，可以满足进程P1的仍然需求资源数，让进程P1执行，当进程P1执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(3,8,8,6)，可以满足进程P3的仍然需求资源数，让进程P1执行。所以T0时刻系统中存在一个安全序列（P0,P2,P1,P3)，系统处于安全状态。

2、设系统中有4个进程P0、P1、P2和P3，有4种类型的资源A、B、C和D，其中A资源的数量为3，B资源的数量为8，C资源的数量为9，D资源的数量为10，已知T0时刻系统处于安全状态，系统状态如表所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 进程 | 已分配资源数 | | | | 最大资源需求量 | | | | 仍然需求资源数 | | | |
| A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 | 0 | 0 | 0 | 0 |
| P1 | 1 | 0 | 0 | 0 | 1 | 7 | 5 | 0 | 0 | 7 | 5 | 0 |
| P2 | 1 | 3 | 5 | 4 | 2 | 3 | 5 | 6 | 1 | 0 | 0 | 2 |
| P3 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 | 0 | 6 | 4 | 2 |

使用银行家算法回答下面的问题。

（1）如果T0时刻从进程P3发来一个请求(0,6,2,0)，是否实施资源分配，为什么？

（2）如果T0时刻从进程P1发来一个请求(0,4,2,0)，是否实施资源分配，为什么？

参考答案：

（1）由已知条件，系统中A、B、C和D四类资源的总数为（3,8,9,10），剩余可用资源的数量是(1,5,2,0)，无法满足从进程P3发来的请求(0,6,2,0)，所以不能分配。

（2）如果T时刻从进程P1发来一个请求(0,4,2,0)，按照银行家算法进行检查，进程P1请求资源数(0,4,2,0)+已分配资源数量(1,0,0,0)小于进程P1最大资源量(1,7,5,0)；另外进程P1请求资源数(0,4,2,0)小于剩余可用资源的数量是(1,5,2,0)。如果满足进程P1的资源请求，进程P1仍然需求资源数变为(0,3,3,0)。

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 进程 | 已分配资源数 | | | | 最大资源需求量 | | | | 仍然需求资源数 | | | |
| A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 0 | 1 | 2 | 0 | 0 | 1 | 2 | 0 | 0 | 0 | 0 |
| P1 | 1 | 4 | 2 | 0 | 1 | 7 | 5 | 0 | 0 | 3 | 3 | 0 |
| P2 | 1 | 3 | 5 | 4 | 2 | 3 | 5 | 6 | 1 | 0 | 0 | 2 |
| P3 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 | 0 | 6 | 4 | 2 |

由已知条件，系统中A、B、C和D四类资源的总数为（3,8,9,10），剩余可用资源的数量是(1,1,0,0)，P0已经获得所需的全部资源，可以先执行，当进程P0执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(1,1,1,2)，可以满足进程P2的仍然需求资源数，让进程P2执行，当进程P2执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(2,4,6,6)，可以满足进程P1的仍然需求资源数，让进程P1执行，当进程P1执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(3,8,8,6)，可以满足进程P3的仍然需求资源数，让进程P3执行。所以T0时刻系统中存在一个安全序列（P0,P2,P1,P3)，系统处于安全状态，可以对进程P1发来的请求(0,4,2,0)，实施资源分配。

3、设系统中有4个进程P1、P2、P3和P4，有3种类型的资源R1、R2和R3，剩余可用数量 (0,1,1)，已知T0时刻系统状态如表所示。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 进程 | 已分配资源数 | | | 最大资源需求量 | | | 仍然需求资源数 | | |
| R1 | R2 | R3 | R1 | R2 | R3 | R1 | R2 | R3 |
| P1 | 1 | 0 | 0 | 3 | 2 | 2 |  |  |  |
| P2 | 6 | 1 | 2 | 6 | 1 | 3 |  |  |  |
| P3 | 2 | 1 | 1 | 3 | 1 | 4 |  |  |  |
| P4 | 0 | 0 | 2 | 7 | 2 | 2 |  |  |  |

使用银行家算法回答下面的问题。

（1）计算每个进程还可能需要的资源，并填入表的“仍然需求资源数”栏目中。

（2）T0时刻系统是否处于安全状态？为什么？

（3）如果T0时刻从进程P3发来一个请求(1,0,1)，是否实施资源分配，为什么？

参考答案：

（1）

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 进程 | 已分配资源数 | | | 最大资源需求量 | | | 仍然需求资源数 | | |
| R1 | R2 | R3 | R1 | R2 | R3 | R1 | R2 | R3 |
| P1 | 1 | 0 | 0 | 3 | 2 | 2 | 2 | 2 | 2 |
| P2 | 6 | 1 | 2 | 6 | 1 | 3 | 0 | 0 | 1 |
| P3 | 2 | 1 | 1 | 3 | 1 | 4 | 1 | 0 | 3 |
| P4 | 0 | 0 | 2 | 7 | 2 | 2 | 7 | 2 | 0 |

（2）由已知条件，T0时刻系统中R1、R2和R3三类资源剩余可用数量 (0,1,1)，可以满足进程P2的仍然需求资源数，让进程P2执行，当进程P2执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(6,2,3)，可以满足进程P1或P3的仍然需求资源数，假如让进程P1执行，当进程P1执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(7,2,3)，可以满足进程P3或P4的仍然需求资源数，让进程P3执行，当进程P3执行完毕，释放它占有的资源，系统中的可用资源的数量增加到(9,3,4)，可以满足进程P4的仍然需求资源数，最后P4执行。所以T0时刻系统中存在一个安全序列（P2,P1,P3,P4)，系统处于安全状态。

（3）由已知条件，T0时刻系统中R1、R2和R3三类资源剩余可用数量(0,1,1)，无法满足进程P3发来一个请求(1,0,1)，所以不能分配。

4、假设4个作业J1~ J4到达系统的时刻和估计运行时间如表所示，系统在t=2时开始作业调度。

4个作业的到达时刻和估计运行时间表

|  |  |  |
| --- | --- | --- |
| 作业 | 到达时刻t | 运行时间 |
| J1 | 0 | 3 |
| J2 | 1 | 4 |
| J3 | 1 | 2 |
| J4 | 3 | 1 |

请回答以下问题：

（1）在先来先服务（FCFS）、短进程优先（SPN）和高响应比优先（HRRN）等几种调度算法中，要使系统用于进程调度的开销最小且无进程“饿死”现象，应选择哪种调度算法？

（2）给出该调度算法下，各个作业的开始时刻、完成时间、周转时间、带权周转时间，以及所有作业的平均周转时间和平均带权周转时间。（如不整除，小数点后保留两位有效数字。）

参考答案：

（1）短作业优先（SJF）调度算法。

（2）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 作业 | 开始时刻 | 完成时间 | 周转时间 | 带权周转时间 |
| J1 | 5 | 8 | 8 | 2.67 |
| J2 | 8 | 12 | 11 | 2.75 |
| J3 | 2 | 4 | 3 | 1.5 |
| J4 | 4 | 5 | 2 | 2 |

所有作业的平均周转时间：6

所有作业的平均带权周转时间：2.23

5、假设系统有4个进程P1~P4，它们的到达时间和服务时间如表所示。

4个进程到达时间和服务时间表

|  |  |  |
| --- | --- | --- |
| 进程 | 到达时间 | 服务时间 |
| P1 | 0 | 4 |
| P2 | 2 | 5 |
| P3 | 4 | 2 |
| P4 | 6 | 3 |

请回答以下问题：

（1）要使系统用于进程调度的开销最小且无进程“饿死”现象，可采用哪种调度算法？

（2）给出该调度算法下，各个进程的开始时间、完成时间、周转时间、带权周转时间，以及所有进程的平均周转时间和平均带权周转时间。（如不整除，小数点后保留两位有效数字。）

参考答案：

（1）先来先服务（FCFS）调度算法。

（2）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 进程 | 开始时间 | 完成时间 | 周转时间 | 带权周转时间 |
| P1 | 0 | 4 | 4 | 1 |
| P2 | 4 | 9 | 7 | 1.4 |
| P3 | 9 | 11 | 7 | 3.5 |
| P4 | 11 | 14 | 8 | 2.67 |

所有进程的平均周转时间：6.5

所有进程的平均带权周转时间：2.14

6、假设系统有4个进程A~D，它们的到达时间和服务时间如表所示。

4个进程到达时间和服务时间表

|  |  |  |
| --- | --- | --- |
| 进程 | 到达时间 | 服务时间 |
| A | 0 | 4 |
| B | 1 | 5 |
| C | 2 | 1 |
| D | 3 | 2 |

请回答以下问题：

（1）采用短进程优先调度算法，会有“饿死”现象。既考虑进程的长短，又考虑进程的等待时间，使等待时间长的长进程不会被“饿死”，可采用哪种调度算法？

（2）给出该调度算法下，各个作业的开始时刻、完成时间、周转时间、带权周转时间，以及所有作业的平均周转时间和平均带权周转时间。（如不整除，小数点后保留两位有效数字。）

参考答案：

（1）高相应比优先（HRRN）调度算法。

（2）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 进程 | 开始时间 | 完成时间 | 周转时间 | 带权周转时间 |
| A | 0 | 4 | 4 | 1 |
| B | 7 | 12 | 11 | 2.2 |
| C | 4 | 5 | 3 | 3 |
| D | 5 | 7 | 4 | 2 |

所有进程的平均周转时间：5.5

所有进程的平均带权周转时间：2.05

4时刻：B：（3+5）/5 C：（2+1）/1 D：（1+2）/2

5时刻：B：（4+5）/5 D：（2+2）/2

7、系统为某进程分配了4个物理块，该进程已访问的页号序列为

2,0,2,9,3,4,2,8,2,4,8,4,5。若进程要访问的下一页的页号为7，回答一下问题：

（1）若采用LRU调度算法，则应淘汰的页号是哪个？画出页面调度过程说明。

（2）若采用FIFO调度算法，则应淘汰的页号是哪个？画出页面调度过程说明。

（3）试比较LRU和FIFO调度算法的优缺点。

参考答案：

（1）LRU

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 0 | 2 | 9 | 3 | 4 | 2 | 8 | 2 | 4 | 8 | 4 | 5 |
| 2 | 0 | 2 | 9 | 3 | 4 | 2 | 8 | 2 | 4 | 8 | 4 | 5 |
|  | 2 | 0 | 2 | 9 | 3 | 4 | 2 | 8 | 2 | 4 | 8 | 4 |
|  |  |  | 0 | 2 | 9 | 3 | 4 | 4 | 8 | 2 | 2 | 8 |
|  |  |  |  | 0 | 2 | 9 | 3 | 3 | 3 | 3 | 3 | 2 |

2号页被淘汰

（2）FIFO

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | 0 | 2 | 9 | 3 | 4 | 2 | 8 | 2 | 4 | 8 | 4 | 5 |
| 2 | 0 | 0 | 9 | 3 | 4 | 2 | 8 | 8 | 8 | 8 | 8 | 5 |
|  | 2 | 2 | 0 | 9 | 3 | 4 | 2 | 2 | 2 | 2 | 2 | 8 |
|  |  |  | 2 | 0 | 9 | 3 | 4 | 4 | 4 | 4 | 4 | 2 |
|  |  |  |  | 2 | 0 | 9 | 3 | 3 | 3 | 3 | 3 | 4 |

4号页被淘汰

8、某请求页式存储管理系统中，页的大小为100字，一个进程的大小为1200字，可能的页面访问的地址序列为10、205、110、40、314、432、320、225、80、130、272、420、128。请回答以下问题：

（1）根据页面访问序列，给出逻辑页号的访问序列。

（2）若初始时没有页在主存，系统采用LRU置换算法，分配给该进程的物理块数为3，画表给出进程页面调度情况，并计算缺页中断率。

（3）若初始时没有页在主存，系统采用FIFO置换算法，分配给该进程的物理块数为3，画表给出进程页面调度情况，并计算缺页中断率。

（4）如何理解FIFO算法是公平算法，而LRU算法是非公平算法。

参考答案：

（1）（2）LRU

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 地址流 | 10 | 205 | 110 | 40 | 314 | 432 | 320 | 225 | 80 | 130 | 272 | 420 | 128 |
| 页号序列 | 0 | 2 | 1 | 0 | 3 | 4 | 3 | 2 | 0 | 1 | 2 | 4 | 1 |
| 页面调  度过程 | 0 | 2 | 1 | 0 | 3 | 4 | 3 | 2 | 0 | 1 | 2 | 4 | 1 |
|  | 0 | 2 | 1 | 0 | 3 | 4 | 3 | 2 | 0 | 1 | 2 | 4 |
|  |  | 0 | 2 | 1 | 0 | 0 | 4 | 3 | 2 | 0 | 1 | 2 |

缺页中断率：9/13

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 地址流 | 10 | 205 | 110 | 40 | 314 | 432 | 320 | 225 | 80 | 130 | 272 | 420 | 128 |
| 页号序列 | 0 | 2 | 1 | 0 | 3 | 4 | 3 | 2 | 0 | 1 | 2 | 4 | 1 |
| 页面调  度过程 | 0 | 2 | 1 | 1 | 3 | 4 | 4 | 2 | 0 | 1 | 1 | 4 | 4 |
|  | 0 | 2 | 2 | 1 | 3 | 3 | 4 | 2 | 0 | 0 | 1 | 1 |
|  |  | 0 | 0 | 2 | 1 | 1 | 3 | 4 | 2 | 2 | 0 | 0 |

（3）FIFO

缺页中断率：9/13

（4）LRU算法依据程序局部性原理确定页面重要性，并依据重要性进行调度，属于非公平算法。FIFO，先访问的页面先置换掉，所以是公平算法。

9、进程P有5页，页面访问次序为3、2、1、0、3、2、4、3、2、1、0、4，采用FIFO调度算法，请写出主存分配该进程3个物理块和4个物理块时的页面调度过程和缺页中断率，结合缺页中断率的变化，说明什么是belady现象。

参考答案：

3个物理块：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 1 | 0 | 4 |
| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 4 | 4 | 1 | 0 | 0 |
|  | 3 | 2 | 1 | 0 | 3 | 2 | 2 | 2 | 4 | 1 | 1 |
|  |  | 3 | 2 | 1 | 0 | 3 | 3 | 3 | 2 | 4 | 4 |

缺页中断率：9/12

4个物理块：

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 1 | 0 | 4 |
| 3 | 2 | 1 | 0 | 0 | 0 | 4 | 3 | 2 | 1 | 0 | 4 |
|  | 3 | 2 | 1 | 1 | 1 | 0 | 4 | 3 | 2 | 1 | 0 |
|  |  | 3 | 2 | 2 | 2 | 1 | 0 | 4 | 3 | 2 | 1 |
|  |  |  | 3 | 3 | 3 | 2 | 1 | 0 | 4 | 3 | 2 |

缺页中断率：10/12

Belady现象是指采用FIFO调度算法，有时会出现分配的物理块数增加，而缺页中断率反而上升的现象

10、如果当前读写磁头正在53号柱面上执行操作，依次有4个等待访问的请求，柱面号依次为 98、37、124、65。

（1）按照最短寻道时间优先调度算法，写出调度顺序（1分）和平均寻道长度（1分）。

（2）按照循环扫描算法（方向从小到大），写出调度顺序（1分）和平均寻道长度（1分）。

（3）同循环扫描算法相比，最短寻道时间优先调度算法可能会引发什么现象？（1分）请简要描述这种现象。（1分）

答案：

（1）调度顺序为65、37、98、124，平均寻道长度为(12+28+61+26)/4=34.25；

（2）调度顺序为65、98、124、37，平均寻道长度为(12+33+26+87)/4=39.5。

（3）最短寻道时间优先调度算法可能会导致饥饿现象。这种现象是指某些进程的请求总是被其他进程的请求抢占而长期得不到服务。

答案解析：

11、某硬盘有200个磁道（最外侧磁道号为0），磁道访问请求序列为：130、42、180、15、199，当前磁头位于第58号磁道并从外侧向内侧移动。

（1）按照SCAN算法，写出调度顺序（1分）和平均寻道长度（1分）。

（2）按照CSCAN算法，写出调度顺序（1分）和平均寻道长度（1分）。

（3）CSCAN算法是SCAN算法的改进，从效果上讲，哪里得到了改进？（2分）

答案：

（1）调度顺序为130、180、199、42、15，平均寻道长度为(72+50+19+157+27)/5=65；

（2）调度顺序为130、180、199、15、42，平均寻道长度为(72+50+19+184+27)/5=70.4。

（3）SCAN算法中对两端磁道请求比较不公平，因为通常两端的请求都是最后得到服务的。相比之下，CSCAN算法消除了对两端磁道请求的不公平。

答案解析：

12、当前磁盘读写位于柱面号20，此时有多个磁盘请求以下列柱面号顺序送到磁盘驱动器：10、22、2、40、6、38。在寻道时，移动一个柱面需要6ms。

（1）按照先来先服务算法，写出算法的调度顺序（1分），和总寻道时间（1分）。

（2）按照电梯算法（方向从0到40），写出算法的调度顺序（1分），和总寻道时间（1分）。

（3）同电梯算法相比，先来先服务算法的优点和缺点分别是什么？（2分）

答案：

（1）先来先服务算法，寻道的次序是20、10、22、2、40、6、38。总的寻道时间为(10+12+20+38+34+32)×6ms=876ms。

（2）电梯算法（方向从0到40），寻道的次序为20、22、38、40、10、6、2。总的寻道时间为(2+16+2+30+4+4)×6ms=348ms。

（3）先来先服务算法的优点是简单、公平；缺点是未对寻道进行优化，寻道时间较长。（这里说“总寻道时间”或“平均寻道时间”较长，都算对）