北京航空航天大学

2019－2020 学年 第二学期期末

《操作系统》

A卷

班 级\_\_\_\_\_\_\_\_\_\_\_\_\_\_学 号 \_\_\_18373205\_\_\_\_

姓 名\_\_\_\_\_\_\_\_陈鸿\_\_\_\_\_\_成 绩 \_\_\_\_\_\_\_\_\_

2020年6月4日

班号 学号 姓名 成绩

《 操作系统 》期末考试卷

注意事项：

1、为避免网络拥堵，请提前10分钟左右将答卷提交到课程中心；

2、如果由于网络等原因不能通过课程中心提交，应在交卷截止时间前直接把答卷通过邮件发给任课老师，并在小班微信群里通知老师相关情况；

3、如果同时通过课程中心和邮件提交了答卷，以课程中心为准；

4、题目整体答在word中题目后面，写清题号。最后可以提交word或者pdf。

**题目：**

1. **存储管理1（共15分）**

一个32位的虚拟存储系统有两级页表，其逻辑地址形式如下：

|  |  |  |
| --- | --- | --- |
| 第一级页表（10位） | 第二级页表（10位） | 页内偏移（12位） |

物理地址为32位，形式为：

|  |  |
| --- | --- |
| 物理页框号（20位） | 页内偏移（12位） |

页表项（PTE）格式为：

|  |  |
| --- | --- |
| 20位 | 12位（为标志位） |
| 物理页框号 | 第0位为有效位，如果为0，表示Invalid；如果为1表示Valid。  第1位为读写位，如果为0，表示 Read Only ；如果为1表示 Read/Write。 |

请问：

1. 进程地址空间共多少字节？（本小题3分）
2. 将答卷同学的学号最后两位做为32位逻辑地址的前两位，例如183763MN对应的逻辑地址为0xMN000000。如果从逻辑地址0xMN000000开始映射4MB页表，第一级页表的逻辑地址在什么位置？第一级页表中有一个表项指向第一级页表，该表项的逻辑地址是多少？说明理由（注意B代表字节，一个32位地址占4字节）（本小题6分）
3. 假设当前进程第一级页表的物理地址为0x00200000，利用后面物理内存的信息，请在下表中写出以下指令的执行结果。（本小题6分）

对于Load指令，如果成功执行，写出读入的数据（读取一个字节），否则写Error；对于Store指令，如果成功执行，写OK，否则写Error。

|  |  |
| --- | --- |
| 指令 | 结果 |
| Load [0x00001034] | 0x12 |
| Store [0x00C07665] | OK |
| Store[0x00C005FF] | ERROR |
| Load [0x00C03012] |  |
| Load [0xFF80078F] | ERROR |
| Load [0xFFFFF00B] |  |

物理内存（大尾端）

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | +0 | +1 | +2 | +3 | +4 | +5 | +6 | +7 | +8 | +9 | +A | +B | +C | +D | +E | +F |
| 00000000 | 0E | 0F | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 1A | 1B | 1C | 1D |
| 00000010 | 1E | 1F | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 2A | 2B | 2C | 2D |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00001010 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 4A | 4B | 4C | 4D | 4E | 4F |
| 00001020 | 40 | 03 | 41 | 01 | 30 | 01 | 31 | 03 | 00 | 03 | 00 | 00 | 00 | 00 | 00 | 00 |
| 00001030 | 00 | 11 | 22 | 33 | 44 | 55 | 66 | 77 | 88 | 99 | AA | BB | CC | DD | EE | FF |
| 00001040 | 10 | 01 | 11 | 03 | 31 | 03 | 13 | 00 | 14 | 01 | 05 | 03 | 16 | 01 | 17 | 00 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00002030 | 10 | 01 | 11 | 00 | 12 | 03 | 67 | 03 | 11 | 03 | 00 | 00 | 00 | 00 | 00 | 00 |
| 00002040 | 02 | 20 | 03 | 30 | 04 | 40 | 05 | 50 | 01 | 60 | 03 | 70 | 08 | 80 | 09 | 90 |
| 00002050 | 10 | 00 | 31 | 01 | 10 | 03 | 31 | 01 | 12 | 03 | 30 | 00 | 10 | 00 | 10 | 01 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00004000 | 30 | 00 | 31 | 01 | 11 | 01 | 33 | 03 | 34 | 01 | 35 | 00 | 43 | 38 | 32 | 79 |
| 00004010 | 50 | 28 | 84 | 19 | 71 | 69 | 39 | 93 | 75 | 10 | 58 | 20 | 97 | 49 | 44 | 59 |
| 00004020 | 23 | 03 | 20 | 03 | 00 | 01 | 62 | 08 | 99 | 86 | 28 | 03 | 48 | 25 | 34 | 21 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00100000 | 00 | 00 | 10 | 65 | 00 | 00 | 20 | 67 | 00 | 00 | 30 | 00 | 00 | 00 | 40 | 07 |
| 00100010 | 00 | 00 | 50 | 03 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00103000 | 11 | 22 | 00 | 05 | 55 | 66 | 77 | 88 | 99 | AA | BB | CC | 00 | 00 | 00 | 07 |
| 00103010 | 22 | 33 | 44 | 55 | 66 | 77 | 88 | 99 | AA | BB | CC | DD | EE | FF | 00 | 67 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001FE000 | 04 | 15 | 00 | 00 | 48 | 59 | 70 | 7B | 8C | 9D | AE | BF | D0 | E1 | F2 | 03 |
| 001FE010 | 10 | 15 | 00 | 67 | 10 | 15 | 10 | 67 | 10 | 15 | 20 | 67 | 10 | 15 | 30 | 67 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001FF000 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 65 | 00 | 00 | 10 | 67 | 00 | 00 | 00 | 00 |
| 001FF010 | 00 | 00 | 20 | 67 | 00 | 00 | 30 | 67 | 00 | 00 | 40 | 67 | 00 | 00 | 50 | 07 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 001FFFF0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 10 | 00 | 00 | 67 | 00 | 10 | 30 | 67 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00200000 | 00 | 10 | 00 | 07 | 00 | 10 | 10 | 07 | 00 | 10 | 20 | 07 | 00 | 10 | 30 | 07 |
| 00200010 | 00 | 10 | 40 | 07 | 00 | 10 | 50 | 07 | 00 | 10 | 60 | 07 | 00 | 10 | 70 | 07 |
| 00200020 | 00 | 10 | 00 | 07 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 00200FF0 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 1F | E0 | 07 | 00 | 1F | F0 | 07 |
| …. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

（1）32位，即2^32字节，4GB

（2）我的学号的18373205

0x05000000->页表起始

第一级页表的逻辑地址：0x05000000>>10+0x05000000=0x05014000

所求表项的逻辑地址是：0x05000000>>20+0x05000000>>10+0x05000000=0x05014050

（3）

|  |  |
| --- | --- |
| 指令 | 结果 |
| Load [0x00001034] | 0x12 |
| Store [0x00C07665] | OK |
| Store[0x00C005FF] | ERROR |
| Load [0x00C03012] |  |
| Load [0xFF80078F] | ERROR |
| Load [0xFFFFF00B] |  |

1. **存储管理2（共10分）**
2. 一个进程的页面走向为：5、4、3、2、4、5、4、1、5、2、5、4、5、2、1，系统中共有3个物理内存页，开始时物理页中没有调入任何页面。使用最优页面置换算法、FIFO页面置换算法和最近最久未使用算法（LRU）的缺页次数各为多少次？（给出计算过程）（本小题6分）
3. 假设只考虑页内碎片和页表引起的额外内存开销。如果进程的平均大小是1MB，每个页表项的大小是8B，为减小额外的内存开销，页面大小应设置为多少？（给出推导过程）（本小题4分）

（1）

OPT：

5 缺 5

4 缺 54

3 缺 543

2 缺 542

4 中

5 中

4 中 542

1 缺 512

5 中

2 中

5 中 512

4 缺 542

5 中

2 中 542

1 缺

OPT总缺页次数 7

FIFO：

5 缺 5

4 缺 54

3 缺 543

2 缺 243

4 中 243

5 缺 253

4 缺 254

1 缺 154

5 中 154

2 缺 124

5 缺 125

4 缺 425

5 中 425

2 中 425

1 缺 415

FIFO 总缺页次数 11

LRU：

5 缺 5

4 缺 54

3 缺 543

2 缺 243

4 中 243

5 缺 245

4 中 245

1 缺 145

5 中 145

2 缺 125

5 中 125

4 缺 425

5 中 425

2 中 425

1 缺

LRU 总缺页次数 9

1. 设每页x字节

则最好能满足：

1MB/x=x/8B

x=2^12B

1. **存储管理3（共10分）**

在一个可变分区内存管理系统中，总内存大小为128KB，某一时刻内存占用情况如下面的内存分配表所示：

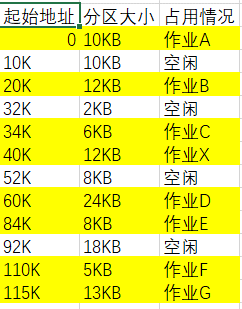
|  |  |  |  |
| --- | --- | --- | --- |
| 分区号 | 起始地址 | 分区大小 | 占用情况 |
| 1 | 0 | 10KB | 作业A |
| 2 | 10K | 10KB | 空闲 |
| 3 | 20K | 12KB | 作业B |
| 4 | 32K | 2KB | 空闲 |
| 5 | 34K | 6KB | 作业C |
| 6 | 40K | 20KB | 空闲 |
| 7 | 60K | 24KB | 作业D |
| 8 | 84K | 8KB | 作业E |
| 9 | 92K | 18KB | 空闲 |
| 10 | 110K | 5KB | 作业F |
| 11 | 115K | 13KB | 作业G |

作业X请求12KB内存，作业Y请求30KB内存，作业Z请求9KB内存。

如果作业到达、结束按照以下顺序进行：X到达、C结束、D结束、Y到达、E结束、Z到达。分别使用FirstFit、 BestFit算法进行内存分配，给出最终内存分配表。

FirstFit

x 到达：



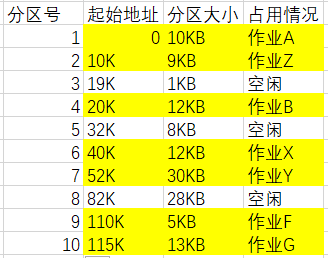
C、D结束：



Y到达：



E结束、Z到达：



BestFit

x起始位置92K

y起始位置32K

z 起始位置10K



1. **磁盘管理（共10分）**

在一个磁盘IO调度系统中，某一时刻t，队列中的磁盘访问请求序列（柱面号）为：10，22，20，2，40，6，38。假设磁头初始位置在第20号柱面，初始时磁头向柱面号大的方向移动，磁头每移动1个柱面需要4ms，磁盘共51个柱面（编号0-50）。

1. 使用电梯调度算法处理完所有上述请求，每次需要扫描到磁盘柱面边界才开始反向扫描。给出实际磁头访问上述请求柱面的次序，并求磁头寻道总时间。（本小题5分）
2. 如果在t+31ms时刻又请求访问50号柱面，t+70ms时刻又请求访问1号柱面，t+91ms时刻又请求访问10号柱面，仍然使用电梯调度算法，处理完所有上述请求，给出从t时刻开始，实际磁头访问请求柱面的次序，并求磁头寻道总时间。（本小题5分）

给出求解过程，包括访问每一请求柱面的时刻、磁头位置、请求队列等。只写出答案不给分。

(1)

顺序20，22，38，40，10，6，2

寻道总时间4ms\*(2+16+2+10+40+4+4)=312ms

(2) 31/4=7.75

已过27 未到28 不影响总的路径

当前请求队列10，2，40，6，38，50

70/4=17.5

已过37 未到38 最后访问1号

当前请求队列10，2，40，6，38，50，1

91/4=22.75

已过42 未到43 当前请求队列不变

当前请求队列10，2，40，6，38，50，1

总顺序：

顺序20，22，38，40，50，10，6，2，1

总时间 312+4=316ms

1. **进程同步与互斥1（共15分）**

1个仓库最多可以容纳50件产品（不分产品类型），每次只允许一个产品进出仓库。甲乙两个车间分别生产A、B两种产品并共用上述仓库。如果仓库满了则不能进行新的生产。有5个需要A产品的客户和5个需要B产品的客户，分别从仓库提取A、B产品。请用P、V操作来实现上述甲、乙车间以及A、B产品的客户之间的同步与互斥关系。给出伪代码描述，并添加尽量详细的注释，说明使用的信号量含义，以及主要代码含义。

semaphore mutex=1,full1=0,full2=0,empty=50;

//mutex 互斥访问缓冲区

//full1 产出A的数量

//empty 缓冲区位置的数量

main()

cobegin{

//p1甲车间

while(ture){

P(empty);

P(mutex);

creatA();

V(mutex);

V(full1);

}

//p2乙车间

while(ture){

P(empty);

P(mutex);

creatB();

V(mutex);

V(full2);

}

//p3需要A的客户

while(ture){

P(full1);

P(mutex);

getA();

V(mutex);

V(empty);

}

//p4需要B的客户

while(ture){

P(full2);

P(mutex);

getB();

V(mutex);

V(empty);

}

}coend

1. **进程同步与互斥2（共15分）**

校车问题：乘客来到校车的停车站等待校车。当巴士到达的时候，所有正在等待的乘客调用 boardBus()上车。一旦开始上车，任何新到来的乘客都必须等待下一辆巴士。校车的容量为50人，如果有超过50个人排队，50名之外的乘客需要等待下一辆巴士。当所有等待的乘客上车完毕，巴士可以离开（调用depart()）。如果巴士到达时，没有任何乘客，它就会立刻离开（调用depart()）。

请用PV操作编写巴士进程和乘客进程的同步互斥关系，满足上述约束条件。给出伪代码描述，并添加尽量详细的注释，说明使用的信号量含义，以及主要代码含义。

semaphore mutex1=1,mutex2=1,canOn=0,canGo=0;

//具体含义见后

canOn告诉后面的人可以上了，cango告诉车可以走了

int count=0;//上车时车上的人数，mutex1互斥访问

int waitingpeople=0;//等待的人，mutex2互斥访问

int i;

main()

cobegin{

//p1乘客

while(ture){

P(mutex2);

waitingpeople++;//等待的人++

V(mutex2);

P(mutex1);

P(canOn);

P(mutex2);

waitingpeople--;

count++;

if(count==50||waitingpeople==0){

V(canGo);

}

else V(canOn);

V(mutex2);

V(mutex1);

}

//p2bus

while(ture){

P(mutex2);

if(waitingpeople==0){

V(mutex2);

depart();

}else {

V(mutex2);

V(canOn);

P(canGo);

P(mutex1);

count=0;

V(mutex1);

depart();

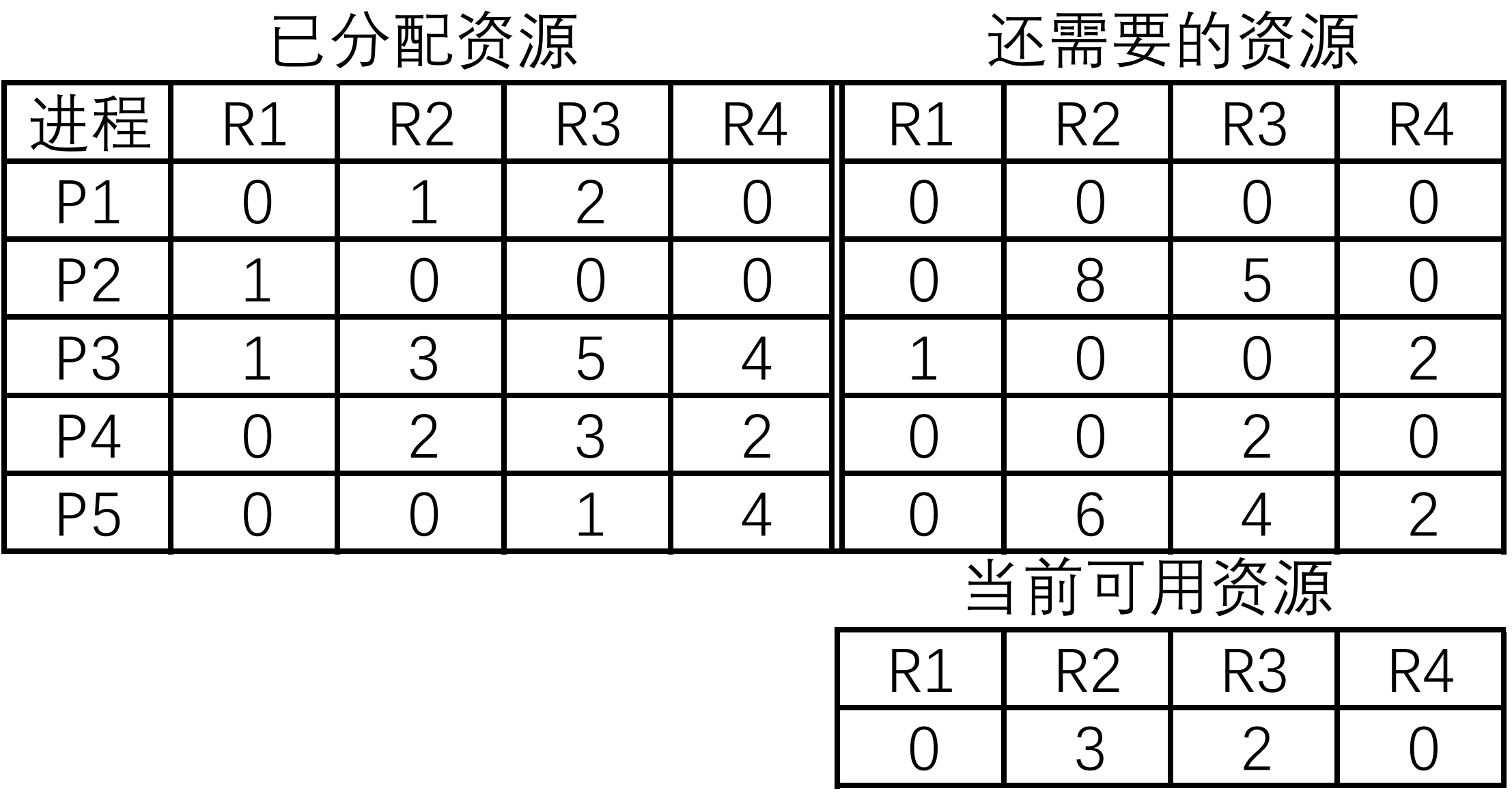
}

}

}coend

1. **死锁（共10分）**

假设系统中有5个进程（P1，P2，P3，P4，P5），运行这些进程需要4类资源（R1，R2，R3，R4）。在某时刻，各进程已经获得分配的资源、还需要的资源以及当前可用资源情况如图所示。



1. 该系统当前是否处于安全状态？是否处于死锁状态？要求给出回答的依据。（本小题6分）
2. 假设4类资源的单位代价相同，如何以最小的代价通过增加资源来避免系统发生死锁？要求给出解答过程。（本小题4分）
3. **P1不再需要资源 结束之后 当前可用变为 0 4 4 0**

**P4可以满足 释放后当前可用变为 0 6 7 2**

**P5可以满足 释放后当前可用变为 0 6 8 6**

**P2（需求0850）和P3（需求1002） 都无法满足 死锁**

**该系统当前处于死锁状态**

**（2）注意到P3与P2发生死锁时，P3仅需一个R1资源即可满足，故最小代价为一个R1资源 P3满足 当前可用变为 2 9 13 10 ,P2可用满足。**

1. **文件系统（共15分）**

一个Unix文件系统采用2KB大小的数据块和4字节的数据块地址。每个i节点包括10个直接索引，1个一级间接索引和1个二级间接索引。

1. 该文件系统中文件最大为多少KB？（本小题5分）
2. 假设磁盘上一半的文件大小恰好为2KB，另一半文件的大小恰好为1.5KB，那么磁盘空间的利用率是多少？如果数据块大小改为1KB，磁盘空间利用率是多少？（只考虑存储文件数据的磁盘块）（本小题6分）
3. 小明编写了一个读取文件的程序，可以从不同大小的文件中随机读取一定量数据，结果发现读取大文件时的平均性能有明显下降。请分析主要原因，并尝试给出解决该问题的思路。（本小题4分）
4. 一个数据块可以存2kB/4B个数据块地址，即512个

文件最大：10\*2KB+512\*2KB+512\*512\*2KB=525332KB

2）a.(2+1.5)/(2+2)=0.875=87.5%

b.(2+1.5)/(2+2)=0.875=87.5%

3）大文件所用的磁盘块数更多，并且大文件往往需要利用间接索引，增加了读取磁盘的次数，若要读大文件，更好的方式是不用索引的方式,，用链表的方式。