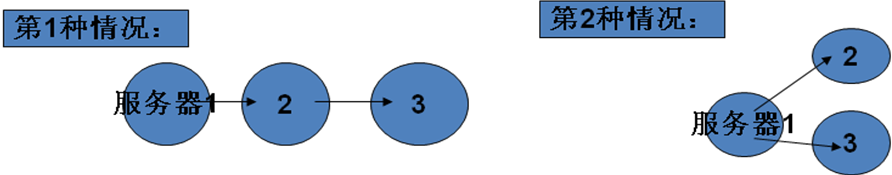
**第1章 作业**

（选学）某客户进程向一个微内核OS请求服务，该请求需要3个核外服务器提供服务。服务器调用关系如下：

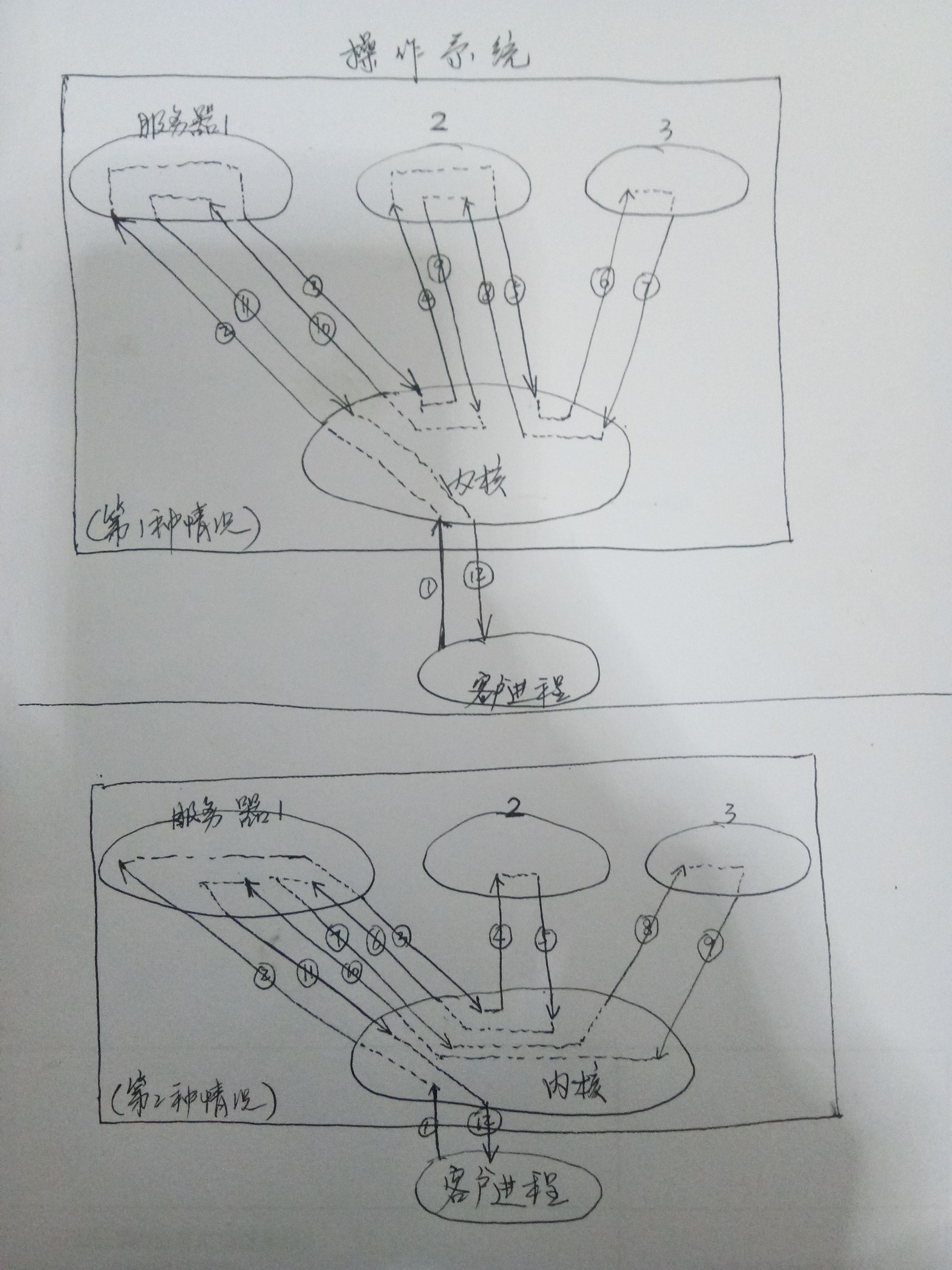


请问两种情况下OS完成该请求需进行多少次上下文切换？画图说明。

答：第1 种情况需进行\_12\_\_次上下文切换。

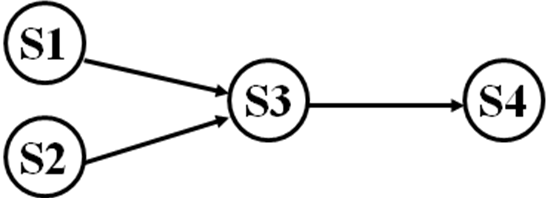
第2 种情况需进行\_12\_\_次上下文切换。

上下文切换示意图如下（可参考课本图1-12）：



**第2章**

习题2.前趋图如下：



习题5.引入进程：为了实现操作系统中多个程序的并发执行。

影响： （1）实现OS中多个程序的并发执行，大大提高资源利用率和系统吞吐量。（2）进程控制块及相关机构占用内存空间，对进程的切换、同步、通信等管理增加了系统开销。

**作业题：有三个进程p1、p2、p3协作解决文件打印问题：p1每次将1条记录从磁盘读入缓冲池1，缓冲池1有x个缓冲区；p2每次将1条记录从缓冲池1复制到缓冲池2，缓冲池2有y个缓冲区；p3每次从缓冲池2取出1条记录打印输出。请用信号量机制实现这三个进程的同步。**

答：**（1）信号量说明：**

**Int In1=0,out1=0,in2=0,out2=0;**

**Item Buffer1[n], Buffer2[m];**

**Semaphore mutex1=1, empty1=n, full1=0;**

**Semaphore mutex2=1, empty2=m, full2=0;**

**Int Nextp=0, nextc=0, nextq=0,nextd=0;**

**（2）进程P1：**

**Do{ ……**

**produce an item nextp;**

**……**

**Swait(empty1，mutex1);**

**Buffer1(in1):=nextp;**

**In1:=(in1+1) % x;**

**Ssignal(mutex1，full1);**

**}while(TRUE);**

**（3）进程P2：**

**Do{**

**Swait(full1，mutex1);**

**nextc:=buffer1(out1);**

**out1:=(out1+1) % x**

**Ssignal(mutex1，empty1);**

**……**

**Comsume the item in nextc;**

**……**

**produce an item nextq;**

**……**

**Swait(empty2，mutex2);**

**Buffer2(in2):=nextq;**

**In2:=(in2+1) % y;**

**Ssignal(mutex2，full2);**

**}while(TRUE);**

**（4）进程P3：**

**Do{**

**Swait(full2，mutex2);**

**nextd:=buffer2(out2);**

**out2:=(out2+1) % y**

**Ssignal(mutex2，empty2);**

**……**

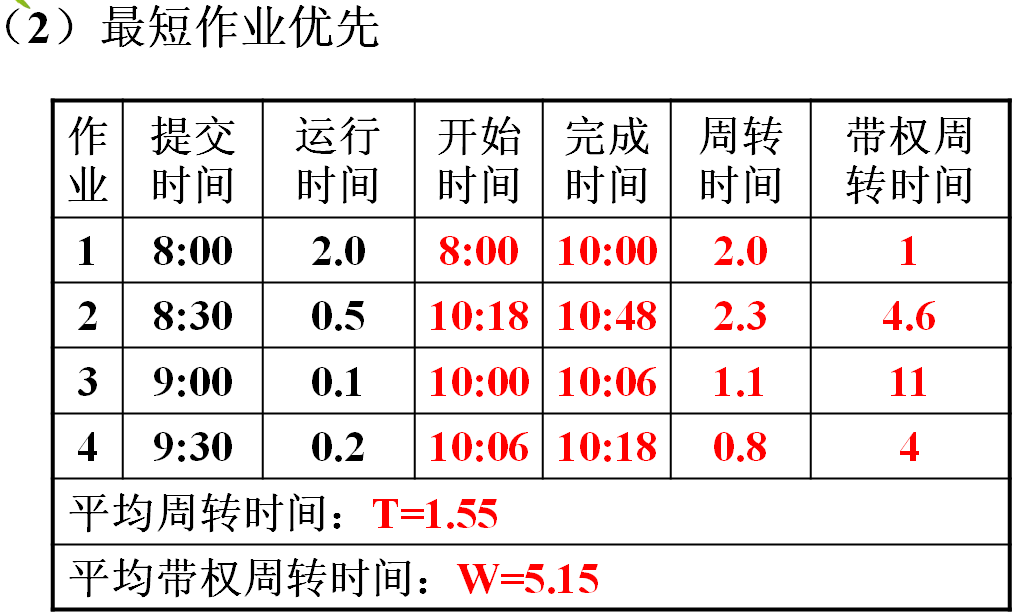
**Comsume the item in nextd;**

**……**

**}while(TRUE);**

**第3章**

补充习题：关于作业调度算法的举例。



作业题（习题31）答案：

（1）该状态是安全的。因为利用安全性算法对该状态进行分析，可得到一个安全序列{P0,P3,P1,P2,P4}使各进程都能正常推进直至全部进程结束。

（2）若进程P2提出请求Request2(1,2,2,2)，按银行家算法进行检查：

①Request2(1,2,2,2)<=Need2(2,3,5,6)，合法；

②request2(1,2,2,2)<=Available1(1,6,2,2),合法；

③系统假设为P2分配资源(1,2,2,2)，则资源变化情况如下图：

|  |  |  |  |
| --- | --- | --- | --- |
| Process | Allocation | Need | Available |
| A B C D | A B C D | A B C D |
| P0 | 0 0 3 2 | 0 0 1 2 | 0 4 0 0 |
| P1 | 1 0 0 0 | 1 7 5 0 |  |
| P2 | 2 5 7 6 | 1 1 3 4 |  |
| P3 | 0 3 3 2 | 0 6 5 2 |  |
| P4 | 0 0 1 4 | 0 6 5 6 |  |

④利用安全性算法，对上图资源状态进行分析，可用资源Available（0，4，0，0）已不能满足任何进程的需要，此刻不存在安全系列，故系统进入不安全状态。

经以上银行家算法分析，若P2提出请求Request2(1,2,2,2)，系统不能将资源分配给它。

**第4章**

略

**第5章**

作业题（习题13）：

采用先进先出（FIFO）算法，当M=3时，页面转换情况如下：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| t | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| p | 4 | 3 | 2 | 1 | 4 | 3 | 5 | 4 | 3 | 2 | 1 | 5 |
| M=3 | 4 | 4 | 4 | 3 | 2 | 1 | 4 | 4 | 4 | 3 | 5 | 5 |
|  | 3 | 3 | 2 | 1 | 4 | 3 | 3 | 3 | 5 | 2 | 2 |
|  |  | 2 | 1 | 4 | 3 | 5 | 5 | 5 | 2 | 1 | 1 |
| F | \* | \* | \* | \* | \* | \* | \* |  |  | \* | \* |  |

从第一页调入算起，缺页次数为9次，缺页率，缺页率=9/12=75% 。

当M=4时，页面转换情况如下：

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| t | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| p | 4 | 3 | 2 | 1 | 4 | 3 | 5 | 4 | 3 | 2 | 1 | 5 |
| M=4 | 4 | 4 | 4 | 4 | 4 | 4 | 3 | 2 | 1 | 5 | 4 | 3 |
|  | 3 | 3 | 3 | 3 | 3 | 2 | 1 | 5 | 4 | 3 | 2 |
|  |  | 2 | 2 | 2 | 2 | 1 | 5 | 4 | 3 | 2 | 1 |
|  |  |  | 1 | 1 | 1 | 5 | 4 | 3 | 2 | 1 | 5 |
| F | \* | \* | \* | \* |  |  | \* | \* | \* | \* | \* | \* |

从第一页调入算起，缺页次数为10次，缺页率，缺页率=10/12=83.3% 。

**（请同学们自己考虑一下，如果用最佳算法或最近最久未使用算法呢？）**

**第6章**

关注补充习题：关于磁盘调度算法

**第7章**

略

**第8章**

**习题7.答：（课堂已讲解）**

**FCB A FAT**

|  |
| --- |
|  |
| **11** |
|  |

|  |  |
| --- | --- |
| **…** |  |
| **11** |  |
| **12** |  |
| **13** |  |
| **14** |  |
| **15** |  |
| **16** |  |
| **17** |  |
| **18** |  |
| **19** |  |
| **20** |  |
| **…** |  |

**FCB B**

|  |
| --- |
|  |
| **13** |
|  |

**9.答：不能。因为 盘块数＝512MB/512B=1M=1024K**

**FAT指针数＝64K**

**一个指针（表项）只能指向一个盘块，64K<1024K**

**14.答：**

**分配过程如下：**

1. **顺序扫描位示图，找到第1个值为0的位，其行列号为：i1=3,j1=3**
2. **计算出其对应的块号：b1=16\*(i1-1)+j1=35**
3. **修改位示图，令map[3,3]=1，然后把第35块分配给文件。**
4. **类似地，找到第2个值为0的位，其行列号为i2=4,j2=7，计算出其对应块号b2=16\*(i2-1)+j2=55，令map[4,7]=1，分配55号盘块。**

**15.答：**

**（1）500/32=15.625，应取16字，即位示图为16行32列。**

**（2）对应块号为：32\*(i-1)+j**

**（3）申请一块的过程：参考上题**

**归还一块的过程：I. 将回收盘块的块号b转换成位示图中的行列号，**

**行号i=(b-1) DIV n+1，列号j=(b-1) MOD n +1 。**

**II. 令map[ i ,j]=0 。**