银行家算法问题

1.

资源/进程	Max			Allocation			Need			available		
	A	B	C	A	B	C	A	B	C	A	B	C
P1	5	5	9	2	1	2	3	4	7	2	3	3
P2	5	3	6	4	0	2	1	3	4			
P3	4	0	11	4	0	5	0	0	6			
P4	4	2	5	2	0	4	2	2	1			
P5	4	2	4	3	1	4	1	1	0			

资源/进程	Work			Need			Allocation			Work+Allocation			Finish
	A	B	C	A	B	C	A	B	C	A	B	C	true
P4	2	3	2	2	2	1	2	0	4	4	3	7	true
P2	4	3	7	1	3	4	4	0	2	8	3	9	true
P3	8	3	9	0	0	6	4	0	5	12	3	14	true
P5	12	3	14	2	2	1	3	1	4	15	4	18	true
P1	15	4	18	3	4	7	2	1	2	17	5	20	true

是安全状态，P4->P2->P3->P5->P1

2.

request (0,3,4)<need (1,3,4) 可以

request (0,3,4)>available (2,3,3) 无法分配,让 P2 等待。

3.

Request(0,2,0) <=Need(3,4,7)

Request(0,2,0) <=Available(0,3,2)

系统先暂时假定为 P4 分配资源：

资源/进程	Max			Allocation			Need			available		
	A	B	C	A	B	C	A	B	C	A	B	C
P1	5	5	9	2	1	2	3	4	7	2	3	2
										0	3	2
P2	5	3	6	4	0	2	1	3	4			
P3	4	0	11	4	0	5	0	0	6			
P4	4	2	5	2	0	4	2	2	1			
				4	0	5	0	2	0			
P5	4	2	4	3	1	4	1	1	0			

检测其是否安全：

资源/进程	Work			Need			Allocation			Work+Allocation			Finish
	A	B	C	A	B	C	A	B	C	A	B	C	true
P4	0	3	2	0	2	0	4	0	5	4	3	7	true
P2	4	3	7	1	3	4	4	0	2	8	3	9	true
P3	8	3	9	0	0	6	4	0	5	12	3	14	true
P5	12	3	14	2	2	1	3	1	4	15	4	18	true
P1	15	4	18	3	4	7	2	1	2	17	5	20	true

是安全状态，P4->P2->P3->P5->P1

4.

P1:Request1(0,2,0)，利用银行家算法进行检查

Request1(0,2,0)<=Need1(3,4,7)

Request1(0,2,0)<=Available(0,3,2)

为 P1 申请分配资源：

资源/进程	Max			Allocation			Need			available		
	A	B	C	A	B	C	A	B	C	A	B	C
P1	5	5	9	2	1	2	3	4	7	2	3	2
				2	3	2	0	3	7	0	3	2
										0	1	2
P2	5	3	6	4	0	2	1	3	4			
P3	4	0	11	4	0	5	0	0	6			
P4	4	2	5	2	0	4	2	2	1			
				4	0	5	0	2	0			
P5	4	2	4	3	1	4	1	1	0			

Available(0,1,2)不满足任何一个进程的需求，无安全序列，无法分配。