Banker's Algorithm Example Solutions

Exercise 1

Assume that there are 5 processes, P_0 through P_4, and 4 types of resources. At T_0 we have the following system state:

Max Instances of Resource Type A = 3 (2 allocated + 1 Available)
Max Instances of Resource Type B = 17 (12 allocated + 5 Available)
Max Instances of Resource Type C = 16 (14 allocated + 2 Available)
Max Instances of Resource Type D = 12 (12 allocated + 0 Available)

Given Matrices												
	Allocation Matrix (N0 of the allocated resources By a process)				**Max Matrix** Max resources that may be used by a process				**Available Matrix** Not Allocated Resources			
	A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	**A**	**B**	**C**	**D**
P_0	0	1	1	0	0	2	1	0	1	5	2	0
P_1	1	2	3	1	1	6	5	2				
P_2	1	3	6	5	2	3	6	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				
Total	**2**	**12**	**14**	**12**								

1. **Create the need matrix (max-allocation)**

Need(i) = Max(i) – Allocated(i)
(i=0) (0,2,1,0) - (0,1,1,0) = (0,1,0,0)
(i=1) (1,6,5,2) - (1,2,3,1) = (0,4,2,1)
(i=2) (2,3,6,6) - (1,3,6,5) = (1,0,0,1)
(i=3) (0,6,5,2) - (0,6,3,2) = (0,0,2,0)
(i=4) (0,6,5,6) - (0,0,1,4) = (0,6,4,2)
Ex. Process P1 has max of (1,6,5,2) and allocated by (1,2,3,1)
Need(p1) = max(p1)- allocated(p1) = (1,6,5,2) – (1,2,3,1) = (0,4,2,1)

Need Matrix = Max Matrix – Allocation Matrix				
	A	**B**	**C**	**D**
P_0	0	1	0	0
P_1	0	4	2	1
P_2	1	0	0	1
P_3	0	0	2	0
P_4	0	6	4	2

2. **Use the safety algorithm to test if the system is in a safe state or not?**
   a. **We will first define work and finish:**

Initially work = available = (1, 5 , 2, 0)
Finish = False for all processes

Finish matrix	
P_0	False
P_1	False
P_2	False
P_3	False
P_4	False

Work vector			
1	**5**	**2**	**0**

   b. **Check the needs of each process [needs(pi) <= Max(pi)], if this condition is true:**
      - **Execute the process , Change Finish[i] =True**
      - **Release the allocated Resources by this process**
      - **Change The Work Variable = Allocated (pi) + Work**

need$_0$ (0,1,0,0) <= work(1,5,2,0)

P0 will be executed because need(P0) <= Work P0 will be True

Finish matrix	
P_0 – 1	**True**
P_1	False
P_2	False
P_3	False
P_4	False

P0 will release the allocated resources(0,1,1,0)
Work = Work (1,5,2,0)+Allocated(P0) (0,1,1,0) = 1,6,3,0

Work vector			
1	**6**	**3**	**0**

Need$_1$ (0,4,2,1) <= work(1,6,3,0)   Condition Is False P1 will Not be executed

Need$_2$ (1,0,0,1) <= work(1,6,3,0)   Condition Is False P2 will Not be executed

Need$_3$ (0,0,2,0) <= work(1,6,3,0)   P3 will be executed

P3 will be executed because need(P3) <= Work P3 will be True

Finish matrix	
P_0 – 1	**True**
P1	False
P_2	False
P_3 -2	**True**
P_4	False

P3 will release the allocated resources (0,6,3,2)
Work = Work (1,6,3,0)+Allocated(P3) (0,6,3,2) = 1,12,6,2

Work vector			
1	12	6	2

Need$_4$ (0,6,4,2) <= work(1,12,6,2)   P4 will be executed

P4 will be executed because need(P4) <= Work P4 will be True

Finish matrix	
P_0 – 1	**True**
P1	False
P_2	False
P_3 -2	**True**
P_4 -3	**True**

P4 will release the allocated resources (0,0,1,4)
Work = Work (1,12,6,2) + Allocated(P4) (0,0,1,4) = 1,12,7,6

Work vector			
1	12	7	6

Need$_1$ (0,4,2,1) <= work(1,12,7,6)   P1 will be executed

P1 will be executed because need(P1) <= Work P1 will be True

Finish matrix	
P_0 – 1	**True**
P_1 -4	**True**
P_2	False
P_3 -2	**True**
P_4 -3	**True**

P1 will release the allocated resources (1,2,3,1)
Work = Work (1,12,7,6) + Allocated(P1) (1,2,3,1) = 2,14,10,7

Work vector			
2	14	10	7

Need$_2$ (1,0,0,1) <= work(2,14,10,7)   P2 will be executed

P2 will be executed because need(P2) <= Work P2 will be True

Finish matrix	
P_0 – 1	**True**
P_1 -4	**True**
P_2 -5	**True**
P_3 -2	**True**
P_4 -3	**True**

P2 will release the allocated resources (1,3,6,5)
Work = Work (2,14,10,7) + Allocated(P1) (1,3,6,5) = 3,17,16,12

Work vector			
3	17	16	12

The system is in a safe state and the processes will be executed in the following order:
P0,P3,P4,P1,P2

Exercise 2:

If the system is in a safe state, can the following requests be granted, why or why not? Please also run the safety algorithm on each request as necessary.

a. P1 requests (2,1,1,0)
   We cannot grant this request, because we do not have enough available instances of resource A.
b. P1 requests (0,2,1,0)

   There are enough available instances of the requested resources, so first let's pretend to accommodate the request and see the system looks like:

	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	1	1	0	0	2	1	0	1	3	1	0
P_1	1	4	4	1	1	6	5	2				
P_2	1	3	6	5	2	3	6	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				

Need Matrix

	A	B	C	D
P_0	0	1	0	0
P_1	0	2	1	1
P_2	1	0	0	1
P_3	0	0	2	0
P_4	0	6	4	2

Now we need to run the safety algorithm:

Initially

Work vector	Finish matrix	
1	P_0	False
3	P_1	False
1	P_2	False
0	P_3	False
	P_4	False

Let's first look at P_0. $Need_0$ (0,1,0,0) is less than work, so we change the work vector and finish matrix as follows:

Work vector	Finish matrix	
1	P_0	True
4	P_1	False
2	P_2	False
0	P_3	False
	P_4	False

$Need_1$ (0,2,1,1) is not less than work, so we need to move on to P_2.

$Need_2$ (1,0,0,1) is not less than work, so we need to move on to P_3.

$Need_3$ (0,0,2,0) is less than or equal to work. Let's update work and finish:

Work vector	Finish matrix	
1	P_0	True
10	P_1	False
5	P_2	False
2	P_3	True
	P_4	False

Let's take a look at $Need_4$ (0,6,4,2). This is less than work, so we can update work and finish:

Work vector	Finish matrix	
1	P_0	True
10	P_1	False
6	P_2	False
6	P_3	True
	P_4	True

We can now go back to P_1. $Need_1$ (0,2,1,1) is less than work, so work and finish can be updated:

Work vector	Finish matrix	
1	P_0	True
14	P_1	True
10	P_2	False
7	P_3	True
	P_4	True

Finally, $Need_2$ (1,0,0,1) is less than work, so we can also accommodate this. Thus, the system is in a safe state when the processes are run in the following order:

P_0,P_3,P_4,P_1,P_2. We therefore can grant the resource request.

Exercise 3

Assume that there are three resources, A, B, and C. There are 4 processes P_0 to P_3. At T_0 we have the following snapshot of the system:

	Allocation			Max			Available		
	A	B	C	A	B	C	A	B	C
P_0	1	0	1	2	1	1	2	1	1
P_1	2	1	2	5	4	4			
P_2	3	0	0	3	1	1			
P_3	1	0	1	1	1	1			

1.Create the need matrix.

	Need		
	A	B	C
P_0	1	1	0
P_1	3	3	2
P_2	0	1	1
P_3	0	1	0

2. Is the system in a safe state? Why or why not?

In order to check this, we should run the safety algorithm. Let's create the work vector and finish matrix:

Work vector	Finish matrix	
2	P_0	False
1	P_1	False
1	P_2	False
	P_3	False

$Need_0$ (1,1,0) is less than work, so let's go ahead and update work and finish:

Work vector	Finish matrix	
3	P_0	True
1	P_1	False
2	P_2	False
	P_3	False

$Need_1$ (3,3,2) is not less than work, so we have to move on to P_2.

$Need_2$ (0,1,1) is less than work, let's update work and finish:

Work vector	Finish matrix	
6	P_0	True
1	P_1	False
2	P_2	True
	P_3	False

$Need_3$ (0,1,0) is less than work, we can update work and finish:

Work vector	Finish matrix	
7	P_0	True
1	P_1	False
3	P_2	True
	P_3	True

We now need to go back to P_1. $Need_1$ (3,3,2) is not less than work, so we cannot continue. Thus, the system is not in a safe state.