Banker’s Algorithm

- Banker’s Algorithm is used to determine whether a process’s request for allocation of resources be safely granted immediately.

or

- The grant of request be deferred to a later stage.

- For the banker’s algorithm to operate, each process has to a priori specify its maximum requirement of resources.

- A process is admitted for execution only if its maximum requirement of resources is within the system capacity of resources.

- The Banker’s algorithm is an example of resource allocation policy that avoids deadlock.

Example:- Consider the following table of a system:

Process	Allocated				Max				Available			
	R1	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**
P1	0	0	1	2	0	0	1	2	2	1	0	0
P2	2	0	0	0	2	7	5	0				
P3	0	0	3	4	6	6	5	0				
P4	2	3	5	4	4	3	5	6				
P5	0	3	3	2	0	6	5	2				

1. Compute NEED Matrix.
2. Is the system in safe state? Justify.

Solution:- Consider the following table of the system:

Process	Allocated				Max				Available			
	R1	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**
P1	0	0	1	2	0	0	1	2	2	1	0	0
P2	2	0	0	0	2	7	5	0				
P3	0	0	3	4	6	6	5	0				
P4	2	3	5	4	4	3	5	6				
P5	0	3	3	2	0	6	5	2				

1. **Compute NEED Matrix = ?**

Need [i] = Max[i] - Allocated[i],

Therefore,

Need Matrix

NEED MATRIX	R1	R2	R3	R4
P1	0	0	0	0
P2	0	7	5	0
P3	6	6	2	2
P4	2	0	0	0
P5	0	3	2	0

2. **Is the system is Safe State?**

By applying the Banker’s Algorithm:

Let **Avail** = Available; i.e . Avail = {2,1,0,0}

Iteration 1. Check all processes from P1 to P5.

For P1:⍰

if (**P1 Need < Avail**)⍰TRUE

then calculate

Avail= Avail + Allocated [P1]

= {2,1,0,0} + = {0,0,1,2}

Avail = **{2,1,1,2}**

2. **Is the system is Safe State?**

By applying the Banker's Algorithm:

Iteration 1.

For P2:⍰

if (**P2 Need < Avail**)⍰FALSE

//then Check for next process.

For P3:⍰

if (**P3 Need < Avail**)⍰ FALSE

//then Check for next process.

2. **Is the system is Safe State?**

By applying the Banker's Algorithm:

Iteration 1.

For P4:⍰

if (**P4 Need < Avail**)⍰TRUE

then calculate

Avail= Avail + Allocated [P4]

= {2,1,1,2} + = {2,3,5,4}

Avail = **{4,4,6,6}**

2. **Is the system is Safe State?**

By applying the Banker's Algorithm:

Iteration 1.

For P5:⍰

if (**P5 Need < Avail**)⍰TRUE

then calculate

Avail= Avail + Allocated [P5]

= {4,4,6,6} + = {0,3,3,2}

Avail = **{4,7,9,8}**

2. **Is the system is Safe State?**

By applying the Banker’s Algorithm:

Iteration 2. Check only process P2 to P3.

For P2:⍰

if (**P2 Need < Avail**)⍰TRUE

then calculate

Avail= Avail + Allocated [P2]

= {4,7,9,8} + = {2,0,0,0}

Avail = **{6,7,9,8}**

2. **Is the system is Safe State?**

By applying the Banker’s Algorithm:

Iteration 2. Check only process P2 to P3.

For P3:

if (**P3 Need < Avail**)TRUE

then calculate

Avail= Avail + Allocated [P3]

= {6,7,9,8} + = {0,0,3,4}

Avail = **{6,7,12,12} =System Capacity**

Since, all the processes got TRUE marked, no further iterations are required.

Therefore, **Safe Sequence = P1, P4, P5, P2 , P3**

Therefore, the System is in the Safe State.

THANKYOU!