Consider the system with 5 processes $\langle P_0 P_1 P_2 P_3 P_4 \rangle$ & 3 Resources (A B C / 11 7 8) Total

a) find Need Matrix. {Need = (Max − allocation.)}

b) If the system in a safe state, if yes find safe Sequence.

	Allocation			Max			Need			Available.		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	1	0	1	7	5	3	6	5	2	4	4	5.
P_1	2	1	0	3	2	2	1	1	2			
P_2	2	0	2	9	0	2	7	0	0			
P_3	1	2	0	2	2	2	1	0	2			
P_4	1	0	0	4	3	2	3	3	2			
	7	3	3									

Available = Total − allocation

= 11 7 8 − 7 3 3

= 4 4 5

Safety Algo.

True { if Need ≤ available. <P1 P3 P4 P0
execute P.
N. available. = Old available + allocation.

false { else.
forward.

P0 =>

Need.				available.		
6	5	2	≰	4	4	5

forward.

P1 =>

Need.				available.		
1	1	2	≤	4	4	5

execute P1

New available = Old. available + allocation

=				4	4	5	+	2 1 0
=				6	5	5		

P2 =>

Need.				available.		
7	0	0	≰	6	5	5

forward.

P3 =>

Need				available		
1	0	2	≤	6	5	5

execute P3

New available = 6 5 5 + 1 2 0 = 7 7 5

445
- 112
av 333

210
P1 +112
322 Return.

= 333
+322
655

P_4 =) Need. available

3	3	2	$\leq$	7	7	5

execute P_4

N. available = | 7 | 7 | 5 | + | 1 | 0 | 0 |

= | 8 | 7 | 5 |

P_0 =) Need. available

6	5	2	$\leq$	8	7	5

execute P_0

N. available = | 8 | 7 | 5 | + | 1 | 0 | 1 |

= | 9 | 7 | 6 |

P_2 =) Need. available

7	0	0	$\leq$	9	7	6

execute P_2.

N. available = | 9 | 7 | 6 | + | 2 | 0 | 2 |

= | 11 | 7 | 8 | = Total Resources

{ Safe. State }

Yes System is in Safe State.

Safe Sequence. $\langle P_1 P_3 P_4 P_0 P_2 \rangle$

Q. 5. <P_0 P_1 P_2 P_3 P_4> Processes

3 Resources.

A	B	C
10	5	7

	Allocation A B C	MAX A B C	Need A B C	Available A B C
P_0	0 1 0	7 5 3	7 4 3	3 3 2
P_1	2 0 0	3 2 2	1 2 2	
P_2	3 0 2	9 0 2	6 0 0	
P_3	2 1 1	2 2 2	0 1 1	
P_4	0 0 2	4 3 2	4 3 0	
	7 2 5			

(5)

(a) find Need Matrix.

(b) is the System in Safe state, if yes then find safe Sequence.

(c) P_1 request for |1|0|2|, is it will be. fullfill immidiately or not.
if then
then find safe Sequence.

Solution Part c

P_1 request |1|0|2|

Resource allocation algorithm.

Issue: if request $\leq$ Need. (T)
if request $\leq$ available (T)
else.
Not fullfill request

if. P_1 Request ≤ Need. } True.

1	0	2	≤	1	2	2

if Request ≤ available.

1	0	2	≤	3	3	2

Request will fullfil immediately.

available = available − Request.

P_1 allocation = allocation + Request.

Need = Need − Request

available = | 3 | 3 | 2 | − | 1 | 0 | 2 |

= | 2 | 3 | 0 |

P_1 allocation = | 2 | 0 | 0 | + | 1 | 0 | 2 |

= | 3 | 0 | 2 |

Need = | 1 | 2 | 2 | − | 1 | 0 | 2 |

= | 0 | 2 | 0 |

Ans →

updated Table.

	Allocation			MAX			Need			Available.		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	5	3	7	4	3	2	3	0
P_1	3	0	2	3	2	2	0	2	0			
P_2	3	0	2	9	0	2	6	0	0			
P_3	2	1	1	2	3	2	0	1	1			
P_4	0	0	2	4	3	2	4	3	0			

P_0 = Need. [7 4 3] $\not\leq$ available [2 3 0] $< P_1$

forward.

P_1 = Need [0 2 0] $\leq$ available [2 3 0]

execute P_1

M. available = Old available + Allocation

= [2 3 0] + [3 0 2]

= [5 3 2]

P_2 = Need [6 0 0] $\not\leq$ available. [2 3 0]

forward.

P_3 =>

Need		
0	1	1

$\not\leq$

Available		
2	3	0

~~Execute~~ P_3 forward

P_4 =>

Need		
4	3	0

$\not\leq$

available		
2	3	0

forward.

P_0 =>

Need		
7	4	3

$\not\leq$

available		
2	3	0

forward.

P_2 =>

Need		
6	0	0

$\not\leq$

available		
2	3	0

forward.

P_3 =>

Need		
0	1	

$\not\leq$

Availm		
2	3	0

Now, System is not in Safe state.
So their will be not safe Sequence.

⇒ Resource allocation Algorithm.

{ if Request ≤ Need.
  &
  if Request ≤ available.

Then. Request approved

else.

decline.

changes { Available = Available − Request
Need = Need − Request.
allocation = allocation + Request.