Sol^{n}:

Consider the below table:

Process	Allocation				Max				[work] Available				(1) [max − Alloc] Need			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0	0	0	0	0
P_1	1	0	0	0	1	7	5	0	1	5	3	2	0	7	5	0
P_2	1	3	5	4	2	3	5	6	2	8	8	6	1	0	0	2
P_3	0	6	3	2	0	6	5	2	3	8	8	6	0	0	2	0
P_4	0	0	1	4	0	6	5	6	3	14	11	8	0	6	4	2
									3	14	12	12				

Safety Algorithm:

1) a) Work = Available ⇒ Work = 1 5 2 0
   b) Finish [i] = false. ⇒ Finish [i] = F.

	P_0	P_1	P_2	P_3	P_4
Finish	F	F	F	F	F
	0	1	2	3	4

2) a) Finish [i] = false
   b) $Need_i \leq Work$. [If no such 'i' exist go to step 4]

(P_0) Finish [0] == false ✓
$Need_0 \leq work$.
0 0 0 0 ≤ 1 5 2 0 ✓

3) Work = work + Alloc.
   Finish [i] = true. [goto step 2]

Work = 1 5 2 0 + 0 0 1 2
work = 1 5 3 2

	P_0	P_1	P_2	P_3	P_4
Finish	T	F	F	F	F
	0	1	2	3	4

Safe Sequence

P_0				

2) Find an index i such that both
Finish[i] = false
Need$_i$ ≤ Work.

P1 (*) Finish[1] = = false ✓
Need$_1$ ≤ Work.
0750 ≤ 1532 ✗

P2 (*) Finish[2] = = false ✓
Need$_2$ ≤ Work
1002 ≤ 1532 ✓

3) Work = Work + alloc.
= 1532 + 1354
Work = 2886.

Finish[2] = True.

	P0	P1	P2	P3	P4
Finish	T	F	T	F	F
	0	1	2	3	4

Safe Sequence | P0 | P2 | | | |

Goto step 2.

2) Finish[1] = = false. ✓
Need ≤ Work.
P1 0750 ≤ 2886 ✓

3) Work = Work + Alloc.
= 2886 + 1000
= 3886

Finish [1] = True.

	P0	P1	P2	P3	P4
Finish	T	T	T	F	F
	0	1	2	3	4

Safe Sequence | P0 | P2 | P1 | | |

Goto Step 2.

2) Finish [3] == false ✓
P3 Need3 ≤ Work.
0 0 2 0 ≤ 3 8 8 6 ✓

3) Work = Work + Alloc.
= 3 8 8 6 + 0 6 3 2
= 3 14 11 8

Finish [3] = True

	P0	P1	P2	P3	P4
Finish	T	T	T	T	F
	0	1	2	3	4

Safe Sequence | P0 | P2 | P1 | P3 | |

Goto Step 2

2) Finish [4] == false ✓
P4 Need4 ≤ Work.
0 6 4 2 ≤ 3 14 11 8 ✓

3) Work = Work + Alloc.
= 3 14 11 8 + 0 0 1 4
= 3 14 12 12

2) Finish[4] = True

	P0	P1	P2	P3	P4
Finish	T	T	T	T	T
	0	1	2	3	4

Safe Sequence | P0 | P2 | P1 | P3 | P4 |

Goto Step 2.

2) Find an index i such that
finish[i] = false
Needi ≤ work
If no such 'i' exist goto step 4.

4) If finish[P] == true for all P, Then sys. is in a safe state.

∴ The system | P0 | P2 | P1 | P3 | P4 | (2)

is in safe state.

(3) New request from process P1 (0 4 2 0) can be granted?

Soln:

Resource Request Algorithm.

i) If Requesti ≤ Needi → Y → Goto step 2
→ Error.

0 4 2 0 ≤ 0 7 5 0 ✓

2) If $Request_i \leq Available$ ✓ → ~~Goto step 3~~
P_i must wait.

0 4 20 ≤ 1 5 20 ✓

3) Then system pretend to have allocated the requested resources to process P_1 by modifying the state as follows:

3) Available = Available − $Request_i$
Available = 1 5 2 0 − 0 4 2 0
Available = 1 1 0 0

$Allocation_i$ = $Allocation_i$ + $Request_i$
~~Allocation~~
$Allocation_1$ = $Allocation_1$ + $Request_1$
= 1 0 0 0 + 0 4 2 0
$Allocation_1$ = 1 4 2 0

$Need_i$ = $Need_i$ − $Request_i$
$Need_1$ = 0 7 5 0 − 0 4 2 0
$Need_1$ = 0 3 3 0

If the resulting resource allocation the transaction is completed & process P_1 is allocated with its resources.
If the new state is unsafe then P_1 must wait for $Request_1$, & the old resource-allocation state is restored.

Apply safety algorithm now:

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

1) a) Work = Available
Finish[i] = false.

Work = 1 1 0 0

	P_0	P_1	P_2	P_3	P_4
Finish	F	F	F	F	F
	0	1	2	3	4

2) Finish [0] == False ✓
$Need_0 \leq Work$.
(P0) 0 0 0 0 $\leq$ 1 1 0 0 ✓

3) Work = work + Alloc.
= 1 1 0 0 + 0 0 1 2
Work = 1 1 1 2
Finish [0] = True.

	P_0	P_1	P_2	P_3	P_4
Finish	T	F	F	F	F
	0	1	2	3	4

Safe Sequence | P0 | | | | |

2) Finish [1] = false ✓
$Need_1 \leq Work$.
(P1) 0 3 3 0 $\leq$ 1 1 1 2 ✗

Finish [2] = = false ✓

(P2) Need2 ≤ Work.

1 0 0 2 ≤ 1 1 1 2 ✓

3) Work = Work + Alloc.

= 1 1 1 2 + 1 3 5 4

= 2 4 6 6

Finish [2] = True

	P0	P1	P2	P3	P4
Finish	T	F	T	F	F
	0	1	2	3	4

Safe Sequence | P0 | P2 | | | |

2) Finish [1] = False ✓

(P1) Need1 ≤ Work.

0 3 3 0 ≤ 2 4 6 6 ✓

3) Work = Work + Alloc.

= 2 4 6 6 + 1 4 2 0

= 3 8 8 6

Finish [1] = True

Finish	T	T	T	F	F

Safe Sequence | P0 | P2 | P1 | | |

2) Finish [3] = = false ✓

(P3) Need3 ≤ Work.

0 0 2 0 ≤ 3 8 8 6 ✓

3) Work = Work + Alloc.
= 3 8 8 6 + 0 6 3 2
= 3 14 11 8

Finish [3] = True.

Finish	T	T	T	T	F

Safe Sequence | P0 | P2 | P1 | P3 |

2) ~~4~~ Finish [4] = = false. ✓
Need$_4$ ≤ Work
0 6 4 2 ≤ 3 14 11 8. ✓

PH

3) Work = Work + Alloc.
= 3 14 11 8 + 0 0 1 4
= 3 14 12 12

Finish [4] = True.

T	T	T	T	T

Safe Sequence | P0 | P2 | P1 | P3 | P4 |

The system is safe, hence a request for P1 (0 4 2 0) can be granted.