OS LAB EVALUATION  
DEADLOCK AND PREVENTIONS  
RITHIKA SRI J  
CB.EN.U4CSE19025  
08-05-2021

Assume that there are 5 processes, P0 through P4, and 4 types of resources. At T0, we have the following system state:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Allocation | | | | Max | | | | Available | | | |
| Process | A | B | C | D | A | B | C | D | A | B | C | D |
| P0 | 0 | 1 | 1 | 0 | 0 | 2 | 1 | 0 | 1 | 5 | 2 | 0 |
| P1 | 1 | 2 | 3 | 1 | 1 | 6 | 5 | 2 |
| P2 | 1 | 3 | 6 | 5 | 2 | 3 | 6 | 6 |
| P3 | 0 | 6 | 3 | 2 | 0 | 6 | 5 | 2 |
| P4 | 0 | 0 | 1 | 4 | 0 | 6 | 5 | 6 |

1. Create the need matrix (max-allocation)  
2. Use the safety algorithm to test if the system is in a safe state.

def calculate\_need(max,allocation):

#Need matrix

need = max

for i in range(n):

for j in range(m):

need[i][j] = need[i][j] - allocation[i][j]

return need

def issafe(available,need,allocation):

#to check safety

work=available

seq=[0]\*n

finish=["F"]\*n

c=0

while(c<n):

found=False

for i in range(n):

if (finish[i]=="F"):

for j in range(m):

if (need[i][j]>work[j]):

break

if(j==m-1):

for e in range(m):

work[e]=work[e]+allocation[i][e]

seq[c]=i

c+= 1

finish[i]="T"

found=True

if (found == False):

print("System is not in safe state")

return

for p in range(n):

seq.append("P"+str(p))

seq.pop(0)

print("System is in safe state.","\nSafe sequence is: ", end = " ")

print(seq)

return

if \_\_name\_\_=="\_\_main\_\_":

#number of process and resources

n=int(input("Number of process: "))

m=int(input("Number of resources: "))

#Available resources

print("Enter Available resources: ")

available=[]

for i in range(0,m):

available.append(int(input()))

#Allocation matrix

allocation=[]

print("Enter Allocation matrix: ")

for i in range(0,n):

alloc\_process=[]

for j in range(0,m):

alloc\_process.append(int(input()))

allocation.append(alloc\_process)

# Max matrix

max=[]

print("Enter Max matrix: ")

for i in range(0,n):

max\_process=[]

for j in range(0,m):

max\_process.append(int(input()))

max.append(max\_process)

need=calculate\_need(max,allocation)

print("Number of process: ",n)

print("Number of resources: ",m)

print("Allocation: ",allocation)

print("Max: ",max)

print("Need: ",need)

issafe(available,need,allocation)

OUTPUT:   
(for the above question)

Number of process: 5

Number of resources: 4

Enter Available resources:

1

5

2

0

Enter Allocation matrix:

0

1

1

0

1

2

3

1

1

3

6

5

0

6

3

2

0

0

1

4

Enter Max matrix:

0

2

1

0

1

6

5

2

2

3

6

6

0

6

5

2

0

6

5

6

Number of process: 5

Number of resources: 4

Allocation: [[0, 1, 1, 0], [1, 2, 3, 1], [1, 3, 6, 5], [0, 6, 3, 2], [0, 0, 1, 4]]

Max: [[0, 1, 0, 0], [0, 4, 2, 1], [1, 0, 0, 1], [0, 0, 2, 0], [0, 6, 4, 2]]

Need: [[0, 1, 0, 0], [0, 4, 2, 1], [1, 0, 0, 1], [0, 0, 2, 0], [0, 6, 4, 2]]

System is in safe state.

Safe sequence is: ['P0', 'P1', 'P2', 'P3', 'P4']

CLIPPED OUTPUT:

