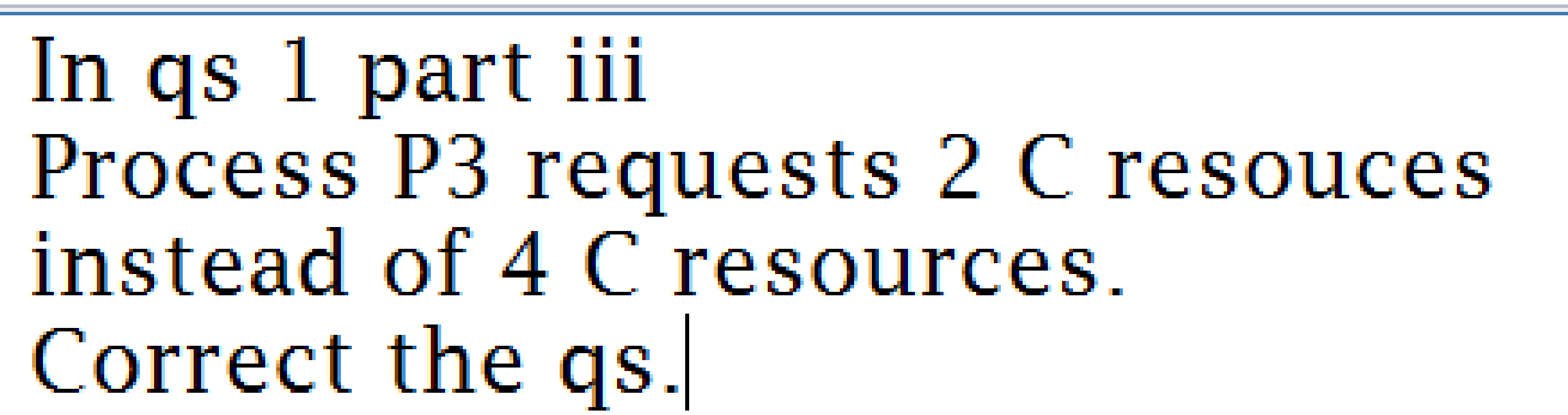
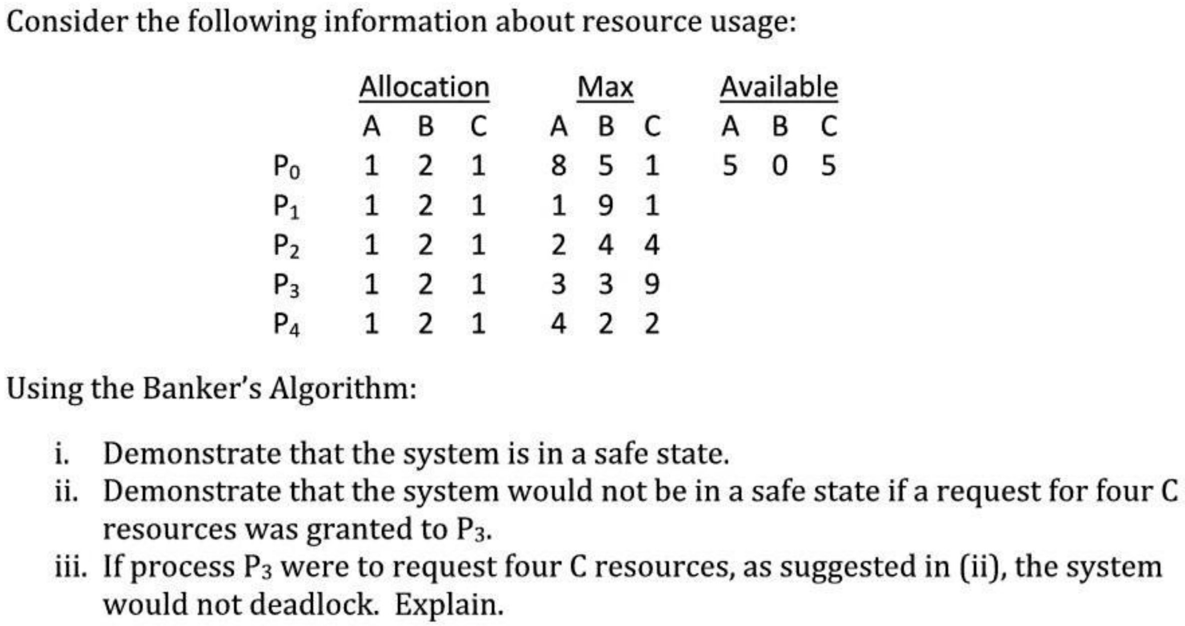


In qs 1 part iii
Process P3 requests 2 C resouces instead of 4 C resources.
Correct the qs.

Assignment 29-08-2022

1)



Consider the following information about resource usage:

	Allocation			Max			Available		
	A	B	C	A	B	C	A	B	C
P_0	1	2	1	8	5	1	5	0	5
P_1	1	2	1	1	9	1			
P_2	1	2	1	2	4	4			
P_3	1	2	1	3	3	9			
P_4	1	2	1	4	2	2			

Using the Banker's Algorithm:

i. Demonstrate that the system is in a safe state.
ii. Demonstrate that the system would not be in a safe state if a request for four C resources was granted to P_3.
iii. If process P_3 were to request four C resources, as suggested in (ii), the system would not deadlock. Explain.

— x —

	Allocation			Max			Available			Need		
P_0	1	2	1	8	5	1	5	0	5	7	3	0 ✓
P_1	1	2	1	1	9	1	6	2	6	0	7	0
P_2	1	2	1	2	4	4	7	4	7	1	2	3 ✓
P_3	1	2	1	3	3	9	8	6	8	2	1	8 ✓
P_4	1	2	1	4	2	2	9	8	9	3	0	1. ✓
							10	10	10.			

Safe sequence:- ~~$P_2 \rightarrow P_4 \rightarrow P_0 \rightarrow P_3 \rightarrow P_1$~~.

$P_4 \rightarrow P_2 \rightarrow P_0 \rightarrow P_3 \rightarrow P_1$

When P_3 requests 4 C resources,

request (0, 0, 4) < Available (5, 0, 5).

So the request is granted to P_3, and we arrive at the following new resource allocation table.

	Allocation			Max			Available			Need		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	1	2	1	8	5	1	5	0	1	7	3	0 ✓
P_1	1	2	1	1	9	1	6	2	2	0	7	0 ✓
P_2	1	2	1	2	4	4	~~7~~	~~4~~	~~3~~	1	2	3 ✓
P_3	1	2	5	3	3	9	~~8~~	~~6~~	~~4~~	2	1	4 ✓
P_4	1	2	1	4	2	2	~~9~~	~~8~~	~~9~~	3	0	1 ✓
							~~10~~	~~10~~	~~10~~.			

Finding the safe sequence,

$P_4 \rightarrow$ [as the need for every process > available resources, so no safe state].

~~As we see that there can be a safe sequence when P_3 requests for an additional 4C resources, so the system would not deadlock, even after the request is granted.~~

As we see that there is no safe sequence when P_3 is allocated an additional 4 C resources, so the resources shall not be granted.

—x—

When a request for 2C resources is granted to P_3, its requirement becomes <1,2,3>. and as request <0,0,2> is less than available <5,0,5> so the request is accepted.

Process	Allocation			Max			Available			Need		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	1	2	1	8	5	1	5	0	3	7	3	0
P_1	1	2	1	1	9	1	6	2	4	0	7	0
P_2	1	2	1	2	4	4	7	4	5	1	2	3
P_3	1	2	3	3	3	9	8	6	6	2	1	6
P_4	1	2	1	4	2	2	9	8	9	3	0	1
							10	10	10			

Safe sequence:

$$P_4 \rightarrow P_2 \rightarrow P_0 \rightarrow P_3 \rightarrow P_1$$

Need for P_4 is <3,0,1> which is less than the available resources <5,0,3>. So the need for P_4 is completed and its resources are deallocated. Similarly we see that P_2, P_0, P_3 and P_1 can be executed sequentially without deadlock.