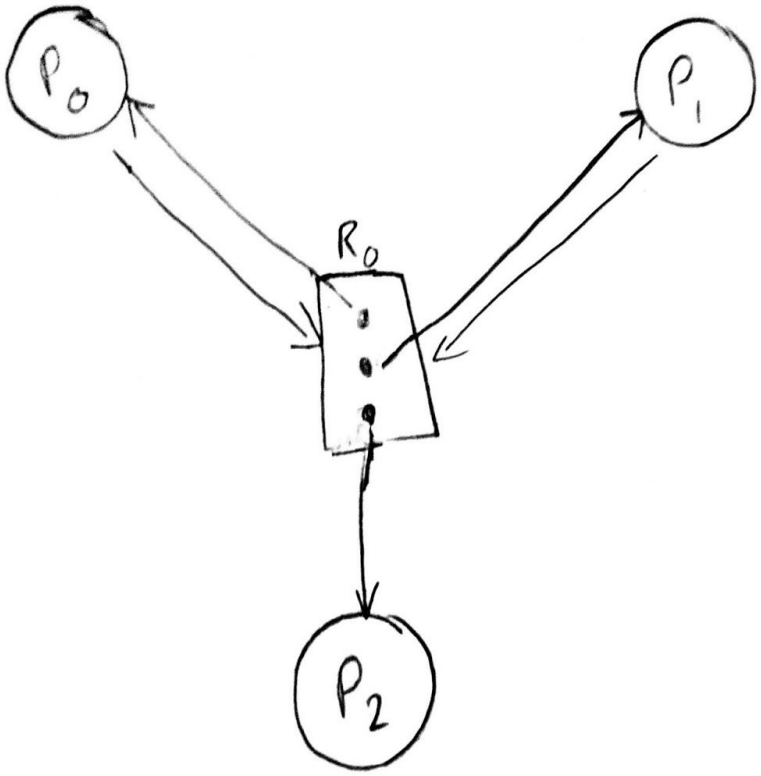
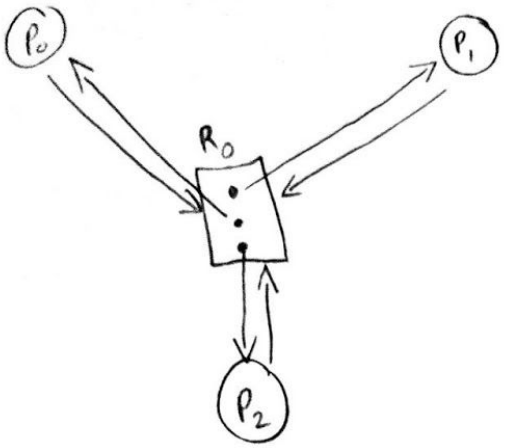
Q1

Unsafe state that finishes.



Order of execution: P_2, P_1, P_0

Unsafe state that leads to deadlock.



Order of execution: There is a deadlock immediately since all three processes are waiting on each other to release a resource that they won't be able to get since R_0 has been already fully utilized.

Sequence:

P_0 requests R_0 - blocked

P_1 requests R_0 - blocked

P_2 requests R_0 - blocked

Q2

Available:

00112 P3 releases
+ 11110
11222 P0 releases
+ 10211
21433 P_2 releases
+ 11010
32443

Need

✓ P_0 01002
✓ P_1 02100
✓ P_2 10300
✓ P_3 00111

P_3, P_0, P_2, P_1

The lowest value x can be is 1 such that the system is in a safe state. What I did was apply the banker's algorithm by first calculating the need matrix as shown above. Then every time a process finished I updated the available vector by adding all the resources a process releases. By doing the algorithm with x = 1, the safe execution is P3, P0, P2, and finally P1.