CS 162 Section 3

Short Answer

1. List the four requirements for deadlock and explain each.

2. Explain two approaches to avoiding deadlock.

Resource Allocation Graphs

Consider a system with four processes P1, P2, P3, and P4, and two resources, R1, and R2, respectively.

Each resource has two instances. Furthermore:

- P1 allocates an instance of R2, and requests an instance of R1
- P2 allocates an instance of R1, and doesn't need any other resource
- P3 allocates an instance of R1 and requires an instance of R2
- P4 allocates an instance of R2, and doesn't need any other resource

Draw the resource allocation graph.

Is there a cycle in the graph? If yes name it.

Is the system in deadlock? If yes, explain why. If not, give a possible sequence of executions after which every process completes.

Resource Allocation Tables

Consider the following snapshot of a system with five processes (P1, P2, P3, P4, P5) and four resources (R1, R2, R3, R4). There are no current outstanding queued unsatisfied requests.

Currently Available Resources

R1	R2	R3	R4
2	1	2	0

	Current Allocation				Max Need				Still Needs			
Process	**R1**	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**	**R1**	**R2**	**R3**	**R4**
P1	0	0	1	2	0	0	3	2	0	0	2	0
P2	2	0	0	0	2	7	5	0	0	7	5	0
P3	0	0	3	4	6	6	5	6	6	6	2	2
P4	2	3	5	4	4	3	5	6	2	0	0	2
P5	0	3	3	2	0	6	5	2	0	3	2	0

Is this system currently deadlocked, or can any process become deadlocked? Why or why not? If not deadlocked, give an execution order.

If a request from a process P1 arrives for (0, 4, 2, 0), can the request be immediately granted? Why or why not? If yes, show an execution order.

If a request from a process P2 arrives for (0, 1, 2, 0), should the request be immediately granted? Why or why not? If yes, show an execution order.