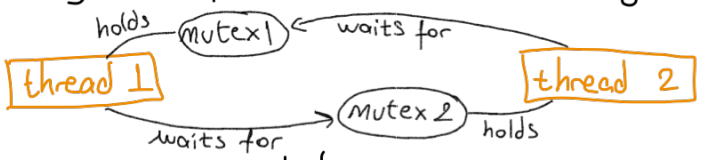
deadlocks

preemption = forcefully taking resource from a thread / process
- ↳ **preemptible resources** = CPU (can stop any process at any time)
- ↳ **non-preemptible resources** = mutex, lock, virtual memory region

deadlock = a set of blocked threads / processes each holding a resource and waiting to acquire a resource held by another in the set
- ↳ when two are more threads wait for another
- ↳ none of the deadlocked threads ever make process



starvation = a thread / process not making any progress since others are using the needed resources
- deadlock ⇒ starvation (when there is a deadlock, there is always starvation) not vice versa (starvation ⇏ deadlock)

deadlock conditions = deadlock happens if all four happen simultaneously

① **mutual exclusion** = only one process can use a resource at a time

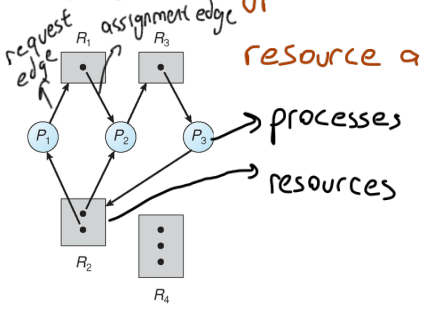
② **hold and wait** = a process holding at least one resource is waiting to acquire additional resources held by other processes

③ **no preemption** = a resource can be released only voluntiarily by the process holding it

④ **circular wait** = there exists a set of waiting processes $P_0 \xrightarrow{waits} P_1 \xrightarrow{waits} P_2 \dots P_n \xrightarrow{waits} P_0$

system model =

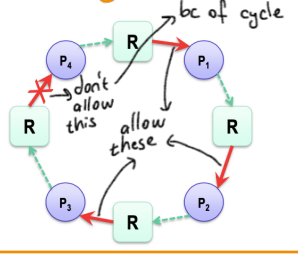
↳ **resource types** = CPU, memory, I/O devices (disk, network..) → each can have multiple instances



resource allocation graph =
- processes
- resources
- if there is deadlock ⇒ there is cycle (not vice versa)
- ↳ no cycle ⇒ no deadlock
- if there is cycle ⇒ if only one instance per resource type ⇒ deadlock
  ⇒ if several instances per resource type ⇒ possibility of deadlock

safe state = when there is no possibility of deadlock, all process completes

dining philosopher's example



finite resource problem - bankers algorithm

processes	resources: Allocated A	B	C	Max A	B	C	current: Needs A	B	C
P1	0	1	0	7	5	3	7	4	3
~~P2~~	2	0	0	3	2	2	~~1~~	~~2~~	~~2~~
P3	3	0	2	9	0	2	6	0	0
P4	2	1	1	2	2	2	0	1	1
P5	0	0	2	4	3	3	4	3	1

worst case

Available A	B	C
~~3~~	~~3~~	~~2~~
2	1	0

↳ P2 completes
3 2 2

↳ P4 completes
5 3 3

P2, P4, P5, P3, P1 → guarantee the completion of all

others are also guarantee, not gurenteed

one also can be completed without deadlock