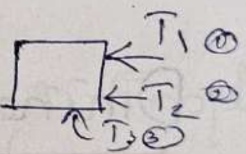
Monitors

* Monitor is one of the ways to achieve Process Synchronization.
* Monitor is supported by pgmming languages to achieve mutual exclusion between processes.
* Eg :- Java Synchronized methods.

* Java provides wait() and notify() constructs.
* A monitor is characterized by set of programmer defined operators.
* Busy Waiting is removed here.
  ↳ doesn't loops.



T_1	T_2	T_3

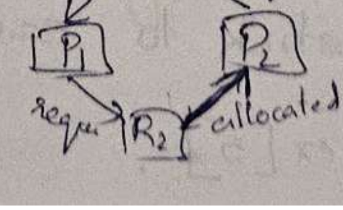
Blocked State

notify() inform T_1 as it comes first & moved to ready state.

notifyall() inform all proces in Blocked state,

Deadlock

* Situation where a set of processes are blocked because each process is holding a resource & waiting for another resource acquired by some other process.



* Necessary Condition for deadlock
  1. Mutual Exclusion
  2. Hold & Wait
  3. No preemption
  4. Circular Wait

1. Mutual Exclusion :-

→ Atleast one resource must be held in a non-sharable mode; ie, only one process at a time can use the resource.

→ If another process requests that resource, the requesting process must be delayed until the resource has been released.

2. Hold & Wait :-

→ A process must be holding atleast one resource & waiting to acquire additional resources that are currently being held by other processes.

Here P_1 holds R_1 & requests for R_2

IIly P_2 holds R_2 & request for R_1.

3. No pre-emption :-

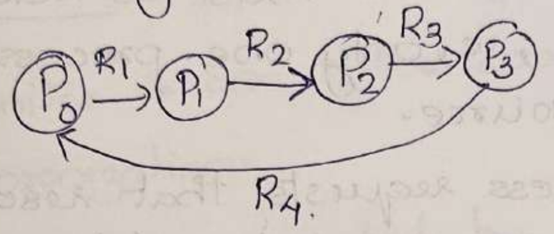
→ Resources cannot be preempted; ie, a resource can be released only voluntarily by the process holding it, after that process has completed its task.

Here once P_1 completes its operⁿ only the resources shared by P_1 can be released.

In blocking state & release resource, then premption.

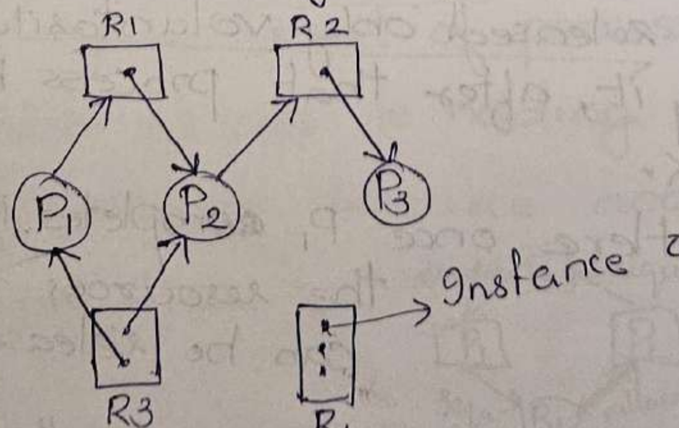
4. Circular Wait :-

→ A set $\{P_0, P_1, \ldots P_n\}$ of waiting processes must exist such that P_0 is waiting for a resource held by P_1, P_1 is waiting for a resource held by P_2, ..., P_{n-1} is waiting for a resource held by P_n and P_n is waiting for a resource held by P_0.

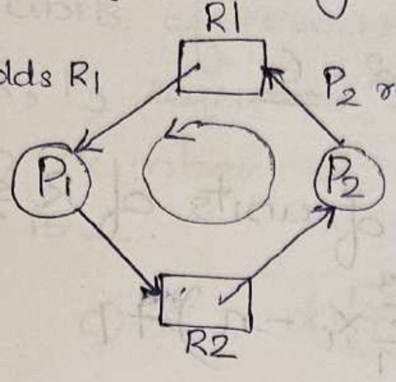


Resource Allocation Graph (RAG)

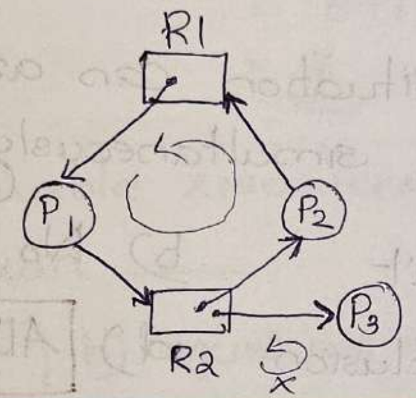
* Deadlocks can be described more precisely in terms of directed graph called a system resource-allocation graph.
* Vertices ⟶ Set of process $P = \{P_1, P_2, \ldots P_n\}$ & Set resources $R = \{R_1, R_2, \ldots R_m\}$
* Edges ⟶
  - ✓ A directed edge $P_i \rightarrow R_j$ (Request edge)
  - ✓ A directed edge $R_j \rightarrow P_i$ (Assignment edge)



* RAG with a cycle may represent a deadlock.



* RAG with cycle may not always represent a deadlock.



Q. Consider there are n processes in the s/m $P_1, P_2, P_3, \ldots, P_n$ where -

- Process P_1 requires x_1 units of resource R.
- Process P_2 requires x_2 units of resource R.
- Process P_3 requires x_3 units of resource R and so on.

In Worst Case :-

The no: of units that each process holds

= One less than its maximum demand.

$= \left(\sum_{i=1}^{n} x_i - n\right) + 1$ where n is the no: of process.

Q. A system is having 3 user processes each requiring 2 units of resource R. The minimum no: of R s.t. no deadlock will occur :-

$n = 3$

$P_1 \rightarrow 2$

$P_2 \rightarrow 2$

$P_3 \rightarrow 2.$

$$\sum_{i=1}^{3} = 2+2+2 = \underline{\underline{6}}$$

The minimum no. of units of R $\therefore$ no deadlock will occur $= \left(\sum_{i=1}^{3} x_i - n\right) + 1$

$$= 6 - 3 + 1 = 4 //$$

Q. A system is having 3 user processes P_1, P_2, P_3 where P_1 require 2 units of resource R, P2 requires 3 units of resource R, P_3 requires 4 units of resource R. The minimum number of units of R that ensures no deadlock is ___ .

a) 3 b) 4 c) 6 d) $\boxed{7.}$

$P_1 \rightarrow 2$
$P_2 \rightarrow 3$
$P_3 \rightarrow 4$

$\therefore n = 3$

$P_1 + P_2 + P_3 = 2+3+4 = \underline{9}$

$$\text{Minimum no: of units} = (9-3)+1 = 6+1 = \underline{\underline{7}}$$

Q. for non-sharable resources like a printer, mutual exclusion ___ .

a) $\boxed{\text{must exist}}$ b) must not exist c) may exist d) None of the mentioned.

Q. Consider a s/m with 3 processes that share 4 instances of the same resource type. Each process can request a maximum of K instances. Resource instances can be requested & released only one at a time. The largest value of K that will always avoid deadlock is ___ .

a) 1 b) $\boxed{2}$ c) 3 d) 4.

P1 P2 P3

$n = 3$

$P_1 + P_2 + P_3 = K+K+K = 3K$

$$4 = (3K-3)+1$$
$$3K = 4+3-1$$
$$K = \frac{6}{3} = 2$$