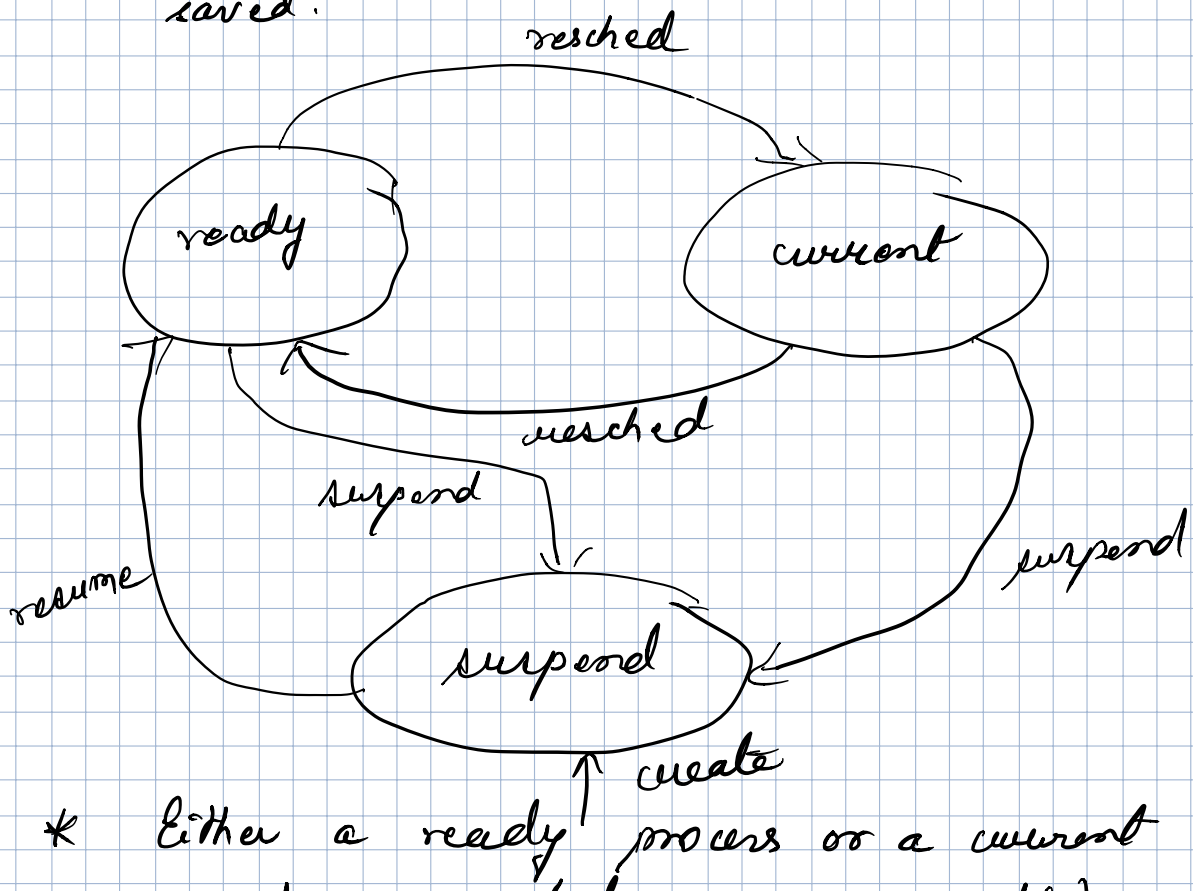
Suspension

* Temporarily stopping a process.
* Prohibit the process from using the CPU.
* We could resume it later.
* Complete state of suspended process is saved.



* Either a ready process or a current can be suspended using suspend().
* Only the suspended process can be resumed.
* A process can suspend itself.
* If somebody resumes a process it is

put on the ready list. it is not executed immediately. why? because this won't violate scheduling policy.

Suspend system call

* Takes process id as argument.
* Validation on process id
  - → whether it is 0 or above
  - → should not be above no. of processes allowed in system.
* Process we want to suspend i.e. the PID we passed to suspend system call must be either
  - → current process
  - → ready state process

Resume system call

* Resume the execution of previously suspended process
* Make process eligible for processor
* Re-establish scheduling event
* Resume → doesn't guarantee instantaneous execution
  - ↳ why? scheduling is required.
* Receives pid as argument
* Checks whether pid(state) == suspended

Yes / NO

record priority to return / Error

(insert proces in ready list)
(call resched)

Q) Complexity of insert function?

O(n) Because it has while loop.

Q) Complexity of getitem function?
It removes a proces from ready list.

O(1)

why? Because there is no loop here. Because of implementation of queuetab. We just track that place in queue and update it.

Ready

* It makes the proces eligible and puts it in ready state.

Proces Termination

* Permanent termination.
* Actual entry needs to be cleared and that entry becomes available for other processes.
* Useful in XINU because XINU allow limited number of processes.
* When a proces is executing, it

might be using resources from kernel. So before termination it must remove them.

* Done through kill() system call
* We should never kill null process.
* Takes process id as argument.
* Switching :- Depends on state of pid passed as argument.

(1) PR_CURR :- Suicide

↓

PR_FREE (clear entry in process table)

↓

resched();

(2) PR_SLEEP / PR_RECTIME (waiting for a timer or message whichever occurs first)

↓ ↓

PR_FREE wasleep (pid)

↓ ↓

semaphore ← PR_FREE

(3) PR_READY

↓

getitem(pid) (remove process from process table)

(4) default

↓

PR_FREE

Process Creation

* Find a free entry in the process table.
* Start populating that entry with the information about new process.
  - ↳ It should be in suspended state.
  - ↳ We need to resume it later
  - ↳ Need to select a new pid.
  - ↳ Typically OS try to avoid reusing process id but when they try to reuse