Process Control Block (*)

- Process Control Block (PCB)
  - OS data structure which contains the state information for each process
    - one PCB per process
  - State information is needed to suspend and correctly resume process execution when another process is scheduled to run
    - Process Identifier (PID), which identifies process
    - User identifier (UID), which identifies the user owning the process
      - User identification is passed from parent to child
    - CPU state
      - Data registers, Program Counter(PC), Stack Pointer(SP), PSW, MAR, MBR, etc.
    - Process Scheduling Control
      - Priority, events pending, process state
    - Process Accounting Information
      - PID, UID, amount of memory used, CPU time elapsed, etc.
      - Example: the UNIX `ps` command information
  - Memory Management
    - Location and access state of all user data
  - I/O Management
    - Files and devices currently opened
    - Device buffer status

Exercise:

1. Check-out UNIX PCB in files `/usr/include/sys/{proc.h, user.h}`
2. Read `man ps` and run the `ps` command to see what accounting state information is kept by the system
Short-Term Scheduling Policies(*)

- Priority Scheduling (Feedback)
  - Multi-level ready queues: RQ0, RQ1, RQ2, …
  - Different priority scheduling for each queue
    - If i < j then RQi is a higher priority to RQj
    - RQ0 is highest priority queue
  - Scheduler selects processes from highest queue which is non-empty
    - Select next process from RQ0.
      If RQ0 is empty then select process from RQ1
      If RQ1 is empty then select process from RQ2, and so on
    - Processes in a higher priority queue will always be scheduled in favour of processes in lower priority queues
  - Process priority aging using priority-based RR scheduling
    - Operation
      - Each new process is admitted to RQ0
        If process runs and times-out it is moved to RQ1
        else process blocks before expiry of quantum and is returned to RQ0 upon wakeup
      - As process keeps timing out it is moved down each priority queue
        If process begins to block before time-out it can be moved up each priority queue
    - CPU bound process will have lower priority to I/O bound processes

Practical use of multi-level priority queues for University machines

- system processes (highest priority)
- interactive processes
- other I/O processes
- staff processes
- batch processes
- student processes (lowest priority)