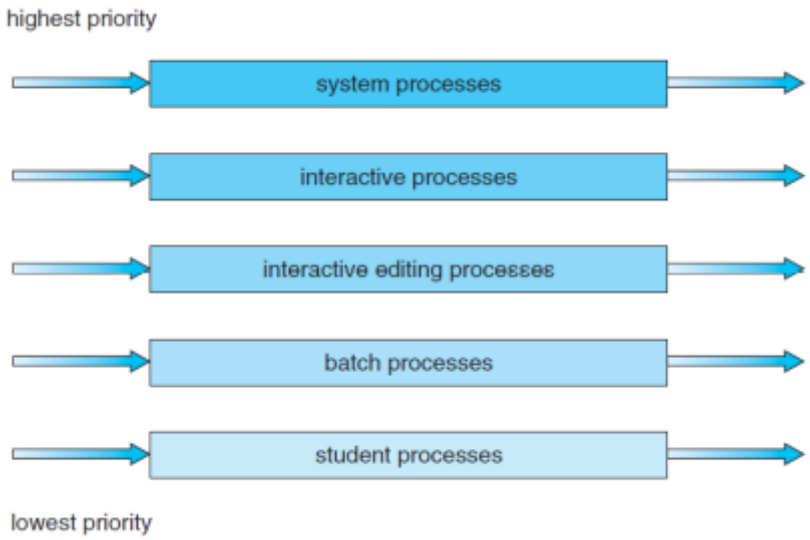
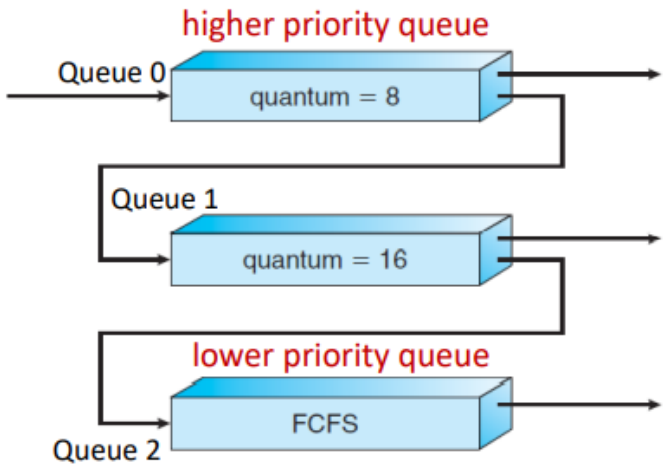
Scheduling lab – Benjamin DAVID, Robin VAN DESSEL

Exercice 2

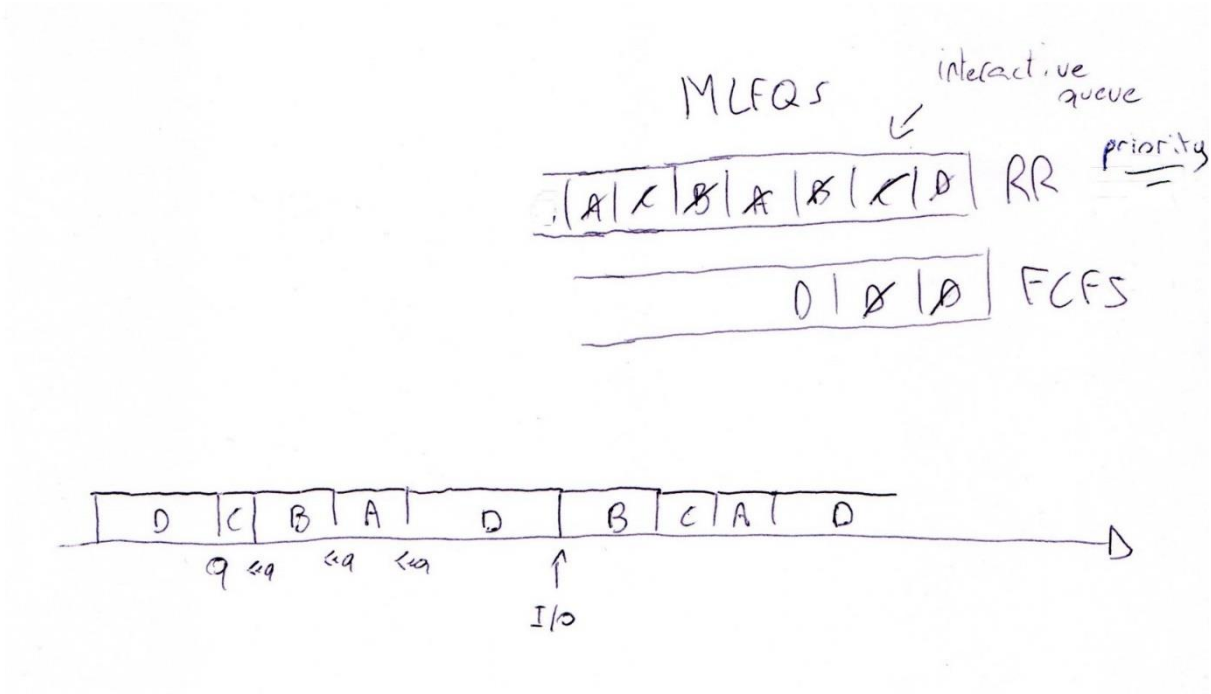
Multilevel Queue Scheduling is designed to handle scheduling of processes classified into different classes where each class has its own requirement and scheduling algorithm. In MLQ, processes are permanently assigned to one queue. No process in lower queues can run unless all above queues are empty. MLQS is commonly used because interactive and background process have different scheduling needs. Interactive process needs to execute fast whereas background processes (batch processes) execution can be delay. In order to reduce interactive process response time, the scheduler will put them in the highest queue.



In **Multilevel Feedback Queue Scheduling** processes can move between queues. Long processes that use too much cpu time are moved down to the lower level. Process in higher priority queue preempt process in lower priority queue. Each queue has its own algorithm. Aging can be applied to prevent starvation: a process that waits too long in a lower-priority queue may be moved to a higher-priority queue.



Example:



Let's take A, B, C three interactives process and D a non-interactive process. All 3 processes enter the RR queue with q=4. D isn't interactive so it runs for q units of time. It is then move to FCFS queue. C, B and A runs for less than the quantum time so they are put back in the RR queue at the end of their I/O. While all interactives process waits for their I/O completion, the interactive queue is empty so D runs. It gets preempted by interactive process when they complete their I/O. D is put back in the FCFS queue because it takes too long to process. If D stays for too long in the FCFS queue, the scheduler will use aging and move D to the RR queue. Also, if D becomes interactive and completes before quantum time, it will be move to the RR queue.