MLFQ - Multilevel **Feedback** Queue Scheduling (preemptive)

➡ Same as MLQ but change the priority of the process based on *observations*

Rule 1	If Priority(A) > Priority(B), A runs
Rule 2	If Priority(A) = Priority(B), A & B run in round-robin fashion using the time slice (quantum length) of the given queue
Rule 3	When a job enters the system, it is placed at the highest priority (the topmost queue)
Rule 4	Once a job uses up its time allotment at a given level (regardless of how many times it has given up the CPU), its priority is reduced (i.e., it moves down one queue)
Rule 5	After some time period S, move all the jobs in the system to the topmost queue

✓ **Good :** Turing-award winner algorithm

Operating System	Preemption	Algorithm
Amiga OS	Yes	Prioritized round-robin scheduling
FreeBSD	Yes	Multilevel feedback queue
Linux kernel before 2.6.0	Yes	Multilevel feedback queue
Linux kernel 2.6.0–2.6.23	Yes	O(1) scheduler
Linux kernel after 2.6.23	Yes	Completely Fair Scheduler
classic Mac OS pre-9	None	Cooperative scheduler
Mac OS 9	Some	Preemptive scheduler for MP tasks, and cooperative for processes and threads
macOS	Yes	Multilevel feedback queue
NetBSD	Yes	Multilevel feedback queue
Solaris	Yes	Multilevel feedback queue
Windows 3.1x	None	Cooperative scheduler
Windows 95, 98, Me	Half	Preemptive scheduler for 32-bit processes, and cooperative for 16-bit processes
Windows NT (including 2000, XP, Vista, 7, and Server)	Yes	Multilevel feedback queue

source: Wikipedia - Scheduling (Computing)
https://en.wikipedia.org/wiki/Scheduling_(computing)