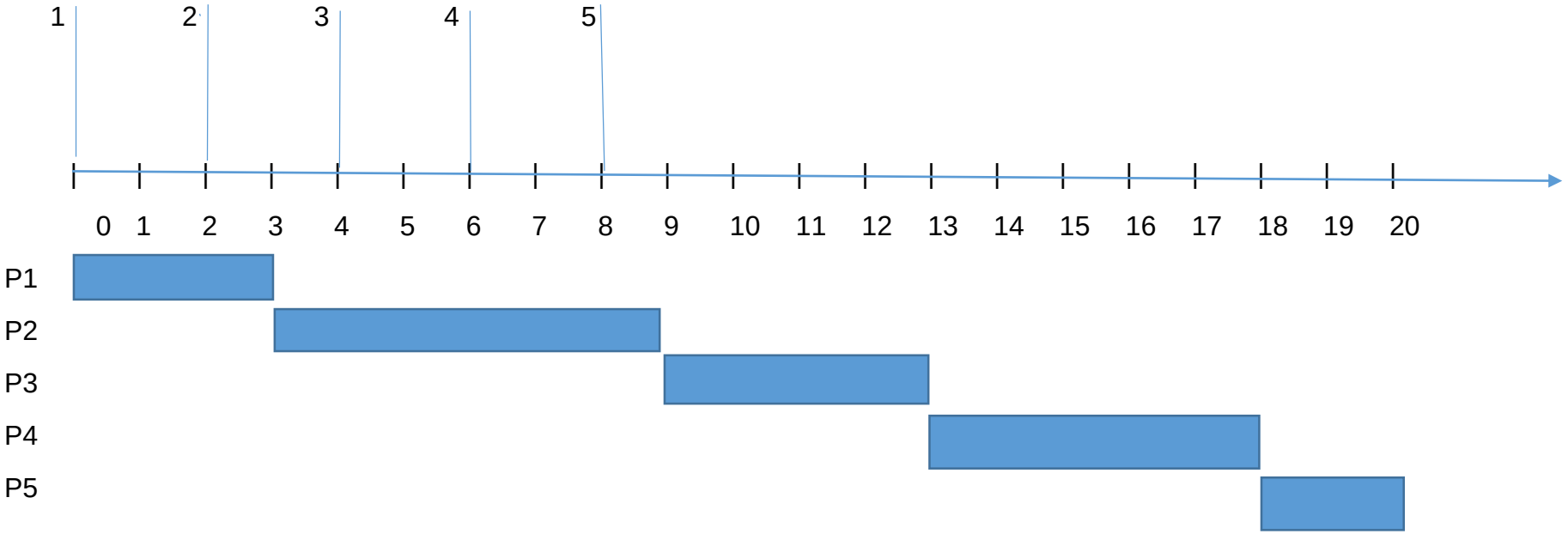
PROCESS	ARRIVAL TIME	CPU BURST
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2

NON-PREEMPTIVE SCHEDULING ALGORITMS

FCFS **first come first serve : the first process in the ready queue is scheduled**

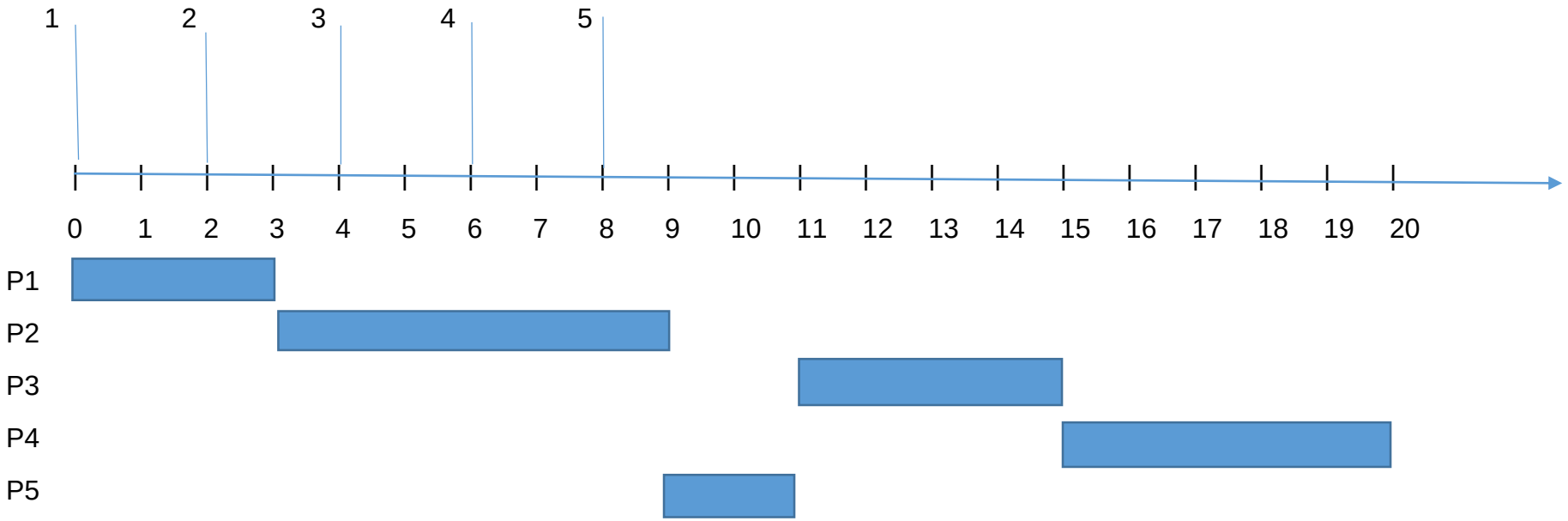
ADVANTAGES: easy to implement, a queue implementation

DISADVANTAGES: if large processes in front of the queue we end up with poor average waiting time and poor average turnaround time



PROCESS	ARRIVAL TIME	CPU BURST
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2

Shortest Job First: the shortest process in the queue is scheduled first. If there is a tie, pick the process with the earliest arrival time.



Advantages:

Disadvantages:

Priority scheduling algorithms: can be preemptive or non-preemptive.

The relation between priority number and the priority of a process depends on the environment, language.

For example: for Java Threads: low priority number means low priority

In Unix the lower the priority number the higher the priority (they is an adjustment value and a nice value in order to determine the priority of a process)

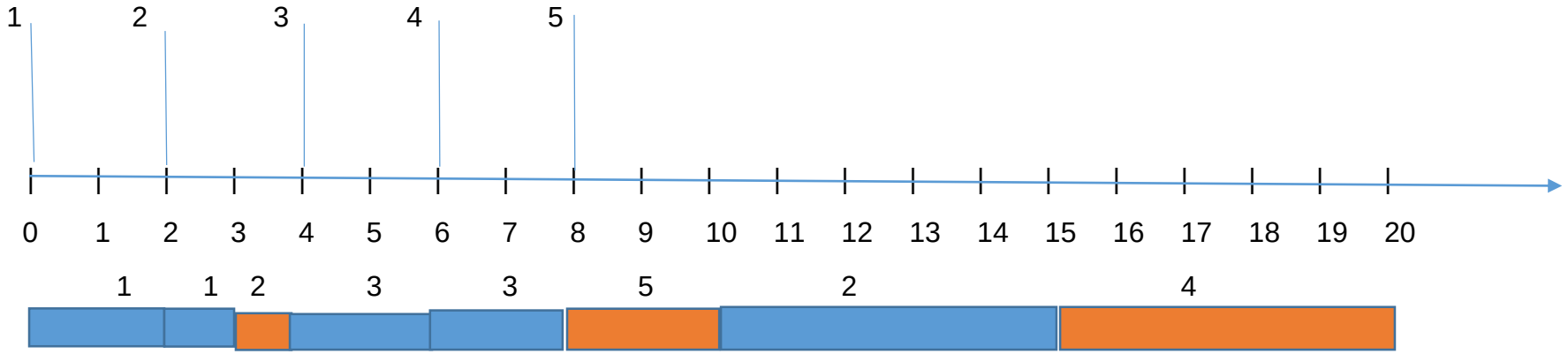
PREEMPTIVE SCHEDULING ALGORITHMS: the process can be preempted while executing its CPU burst because a new process with better credentials got in the ready queue, or for other reasons like the fact that the quantum expired. In that case the process gets back at the end of the ready queue and waits for being rescheduled.

PROCESS	ARRIVAL TIME	CPU BURST
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2

SHORTEST REMAINING TIME

In the beginning, the process with the shortest CPU burst is scheduled. If there is a tie than pick the one with the earliest arrival time.

If, while the process executes another process arrives in the ready queue, there is an interrupt; The scheduler compares the remaining time of the interrupted process with the CPU burst of the new process. If the remaining time is <= than the CPU burst of the new process, the already scheduled process continues its execution. Otherwise the process is preempted and the new arrived process is scheduled.



P1	0			1	0															
P2	-			6			5									0				
P3	-						4			2			0							
P4	-									5										0
P5	-												2		0					

Advantages:

Disadvantages:

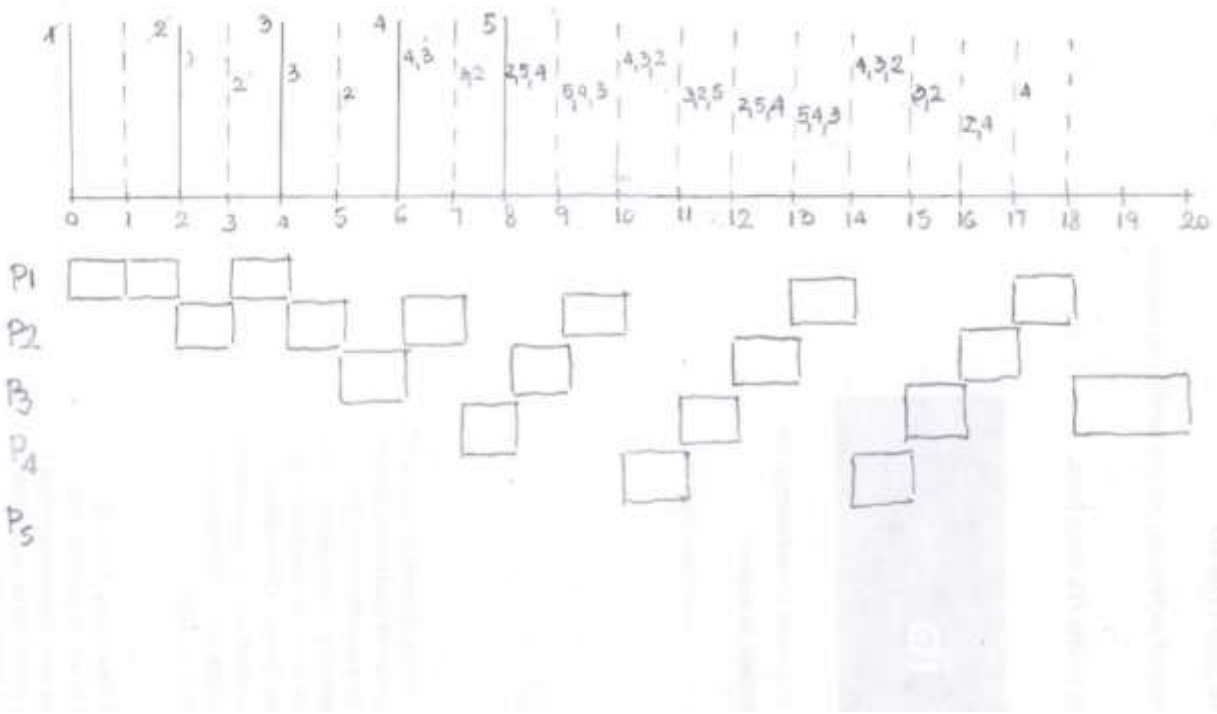
Round Robin Scheduling Algorithm: a process cannot execute more than a fixed quantum of time

Advantages:

Disadvantages:

Picking the size of the quantum:

In this example: Q = 1 time unit



Multilevel Scheduling Algorithms: We have multiple queues.

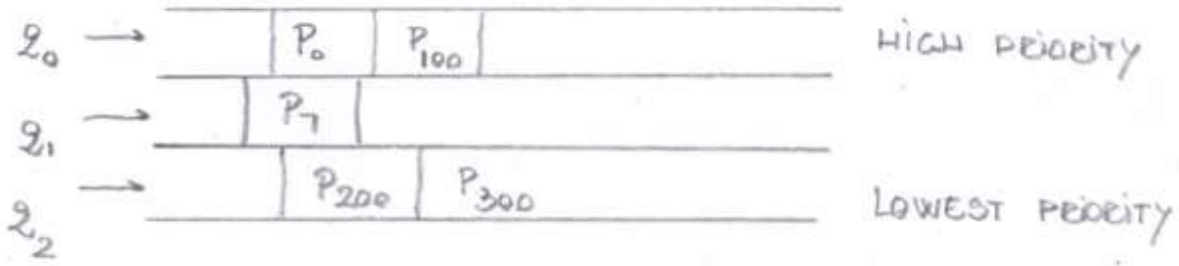
Between queues we have non-preemptive or preemptive priority scheduling algorithms.

Inside the queue, each queue has its own scheduling algorithms.

Design questions: What process goes in what queue and when. If the queue uses a preemptive algorithm once the process is preemptive, where the process goes? Into the same queue, in an above queue, in a below queue.

How it works: Let's consider we have 3 queues and processes inside these queues.

q0, q1, q2. q0 has the highest priority, q1 next highest priority, q2 the lowest priority.



Multilevel Feedback Scheduling Algorithms

Multilevel Feedback Queue Scheduling allows a process to move between queues. It can be moved to a lower-priority queue (if the process uses too much of the CPU) or to a higher-priority (if the process has a large waiting time).

The following multilevel scheduling algorithm establishes a preference for shorter jobs. Scheduling is done on a preemptive basis, and a dynamic priority mechanism is used.

When a process first enters the system, it is placed in q0. When it returns to the Ready list, it is placed in ql. After each subsequent execution it is demoted to the next lower priority queue. A short process will complete without migrating very far downward. Newer, shorter processes are favored over older, longer processes.

Within each queue, except the lowest priority one, a Round Robin policy is used. The lowest-priority queue is treated in a FCFS fashion.

There are many flavors that can be used in implementing a Multilevel Feedback Queue Scheduling algorithm. Let's first define our policy: We will consider a version that uses preemption in the same fashion as the Round Robin, at periodic intervals of time, and the preemption times vary according to the queue.

If a new process arrives in q0 while a process is running in ql, the process in ql will be preempted ONLY AFTER it finishes its time quantum.

PROCESS	ARRIVAL TIME	CPU BURST
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2

q0: round robin with quantum =1

q1: round robin with quantum of 2

q2: FCFS