Compare between all 3 scheduling algorithms

Round Robin (RR):

- In Round Robin scheduling, each process is assigned a fixed time quantum or time slice. The processes are executed in a circular order for the duration of their time slice.
- If a process doesn't complete within its time quantum, it's moved to the back of the queue and the next process is given CPU time.
- It ensures fairness by giving each process an equal share of the CPU.
- Suitable for time-sharing systems, as it prevents any single process from monopolizing the CPU.
- May suffer from high context-switching overhead if the time quantum is too small.
- Works well when processes have similar resource requirements.

Priority Scheduling:

- In Priority scheduling, each process is assigned a priority value. The process with the highest priority is given CPU time. Preemption can occur when a higher-priority process arrives or becomes ready.
- Ensures that high-priority processes get executed promptly.
- Can suffer from "starvation," where lower-priority processes never get a chance to execute if high-priority processes keep arriving.
- Prioritization must be managed carefully to avoid certain processes being neglected.
- Suitable for real-time systems or situations where some processes require immediate attention.

First Come First Serve (FCFS):

- In FCFS scheduling, processes are executed in the order they arrive in the ready queue. The first process to arrive gets the CPU first.
- Simple to implement and understand.
- May lead to "convoy effect" where a long process holds up shorter processes in the queue behind it.
- Not suitable for time-sharing systems as it doesn't ensure fair sharing of CPU time.
- Typically not used in practice for general-purpose operating systems due to its lack of responsiveness and potential inefficiencies.