1. Process 1:

Scheduling Policy: SCHED_OTHER
Priority: Standard priority (nice: 0)
Task: Count from 1 to 2^32
2. Process 2:

Scheduling Policy: SCHED_RR
Priority: Default priority
Task: Count from 1 to 2^32
3. Process 3:

Scheduling Policy: SCHED_FIFO
Priority: Default priority
Task: Count from 1 to 2^32
To benchmark these scheduling policies, you'll need to perform the following steps:

Measure Execution Time: The parent process should record the start time (using clock_gettime()) before creating the child processes and the end time after each child process terminates (detected using the waitpid() system call).

Generate Histograms: You'll need to create histograms to visualize and compare the performance of these scheduling policies. The histograms should depict:

Time taken to complete the counting task on one axis.
Scheduling priorities on the other axis.
Use different colors to represent each scheduling policy.
Plot with Python: You can use Python to plot the histograms, allowing you to visualize and compare the completion times of the three different scheduling policies.

In essence, this exercise involves creating, timing, and benchmarking three processes with varying scheduling policies to determine which one completes the counting task most efficiently. The resulting histograms will provide a visual representation of the performance differences among the scheduling policies.