- The program opens the "Output.txt" file to redirect standard output to it.
- It records the start time for measuring the execution time of the entire program using **clock_gettime().**
- It **forks the first child process** (pid1) to execute the FIFO_program. The child process is given a new priority using the nice() function and then executes the program using execvp().
- The parent process (main) records the start time for measuring the execution time of the FIFO_program.
- The program forks the second child process (pid2) to execute the RR_program and measures its execution time similarly to the first child process.
- The program forks the third child process (pid3) to execute the OTHER_program and measures its execution time.
- The parent process waits for each child process to complete using waitpid() and records the end time for measuring the execution time of each program.
- Finally, the program calculates the execution times for each of the three programs (FIFO, RR, and Other) in seconds and writes to the output.txt file.
- The program is designed to compare the execution times of different scheduling algorithms or program types (FIFO, RR, and Other) and record the results in the "Output.txt" file.
- The outcomes of the tests/measurements are:
- Execution times of FIFO_program, RR_program, and OTHER_program are recorded in seconds.
- The results are printed to the standard output and redirected to the "Output.txt" file.
- The graph can be plotted by running python script in the code.

In summary, the performance ranking can be explained by the **scheduling algorithms and task characteristics.** FIFO, being straightforward, performed well under certain conditions. RR shared resources fairly but introduced some overhead. The "OTHER_program" might involve tasks with distinct resource demands, making it the slowest. The choice of scheduling algorithm should consider the specific requirements of the tasks being performed.