DOCUMENTATION

GROUP MEMBERS:
1. Mohd Masood - 2022299
2 Akshat Karnwal – 2022052
Group No-86

This C code is a simplified implementation of a job scheduler with round-robin scheduling, where processes are executed in a cyclic manner. It uses shared memory to communicate between a shell-like interface and a daemon process responsible for scheduling and managing processes.

Let's break down the code into key components:

1. **Header Files:**
   - Standard C libraries are included for file operations, process management, time handling, and inter-process communication.

2. **Shared Memory Setup:**
   - Shared memory is established using `shm_open` and `mmap` to create and map memory regions accessible by multiple processes.
   - Five shared memory segments (`SHM_NAME`, `SHM_NAME2`, `SHM_NAME3`, `SHM_NAME4`, `SHM_NAME5`) are used to store process information, queue pointers, and flags.

3. **Struct Definition:**
   - A structure `struct Process` is defined to represent a process with attributes such as process state, PID, termination status, and the command to be executed.

4. **Function Definitions:**
   - `sleep_ms`: Sleeps for a specified number of milliseconds using `nanosleep`.
   - `current_time_millis`: Retrieves the current time in milliseconds using `gettimeofday`.
   - Queue manipulation functions (`enqueue`, `dequeue`, `enqueue2`, `dequeue2`) are defined to manage the ready and running queues.

5. **Command Parsing:**
- The `read_fn` function reads a command, splits it, and sets flags accordingly. It supports commands like "history," "submit," and "scheduler."
- remove_new_line replaces the new line character by \0.

6. **Pipeline Implementation:**
   - The `pipel` function handles command pipelines using pipes, forks, and execvp.
Although this is not used in this assignment.
Rather it was used in the previous assignment.

7. **History Logging:**
   - The `updateHistory` function logs process information, including Executable name, PID, start time, and execution time, wait time, into a file (`data.txt`). Dumping this history on recieving Ctrl C signal by using signal handling.

8. **Round-Robin Scheduler (`start_sched2`):**
   - The scheduler function dequeues processes from the ready queue, forks them, stops them using SIGSTOP, and enqueues them back.
   - It then dequeues a specified number of processes and sends them SIGCONT, allowing them to run for a time slice (`TSLICE`).
   - After the time slice, it checks if processes have completed or need to be stopped, updating the ready queue accordingly.

9. **Signal Handling:**
   - The `handle_timer` function is a signal handler for SIGALRM. It invokes the scheduler function when a timer signal is received by passing the signal manually to the daemon process using kill command.

10. **Signal Sending Mechanism (`send_signal`):**
   - The `send_signal` function sends periodic signals to the daemon process to trigger scheduling. This is done to ensure automatic working and scheduling of the processes. Ideally it should same as time slice but for easiness and showing the round robin schedling and other functionalities, we have kept it 10 sec.
It can be manipulated accordingly.

11. **Main Function:**
   - The main function initializes shared memory, forks a daemon process for scheduling, and another process to send periodic signals.

- It enters a shell-like loop, accepting commands such as "submit," "run," and "print."
- The shell interacts with the daemon, enqueuing processes for scheduling.
- As the ready queue is using shared memory, it is easy for the parent to shared data to a seperate daemon scheduler process.

12. **Shared Memory Cleanup:**
   - At the end, shared memory is unmapped, and the shared memory segments are unlinked.

This code represents a simple job scheduler, but it's worth noting that handling processes in a real-world scenario involves more complexities, error checking, and robustness considerations.

CONTRIBUTIONS:

1. Akshat Karnwal - Update history, priority implementation, daemon, basic shell loop
2. Mohd Masood – Scheduler Round Robin function, shared memory, daemon

Git hub link-

****