**OPERATING SYSTEM**

ASSIGNMENT-3 : SimpleScheduler - A Process Scheduler in C

This assignment is the continuation of SimpleShell i.e. Assignment-2. In assignment-2 we already implemented the shell in C from scratch. We mainly used fork, pipes, exec\_family commands.

The OS as a juggler is able to schedule limited number of CPUs across a large number of processes by using some CPU scheduling policy, where each process in the ready state gets to execute on the limited number of CPU resources for some quantum. Once the quantum of running process expires, it is moved to the ready queue from the running queue, and another process from the ready queue is brought into the running queue.

So, In Assignment-3 we have to implement a simpleScheduler.  
We have to implement mainly two functionalities :

1. Basic :

In, basic functionality we basically have to make a basic round-robin scheduling algorithm.

To this we run our shell that takes the user input infinitely, and we just checked for a input that contains *“submit”* using strncmp command and we separate the executable using “char\* executable\_path = command + 7;” and we forked a process in the submit part and in the child part we use execvp(executable\_path, argv) command to execute the executable\_path and in parent process we just STOP the execution of that process using kill(child, SIGSTOP) == 0 and we shared Queue to pass the process information like PID, execution time and waiting time to scheduler we use semaphores also to avoid some issues. After enqueuing all the details into the queue we just used the queue in the scheduler and we dequeued the element form the queue one by one and use their details to CONTINUE the process using kill(p->data[0], SIGCONT)here we can take also input in the queue after a successful execution of the processes in the queue. Here we also calculated the execution time and waiting time for each processes.

Scheduler will run as a daemon in the shell but trigger its execution we used a execute command   
Input format:  
submit ./h  
submit ./fib

submit ./h

submit ./h

submit ./fib

submit ./h

submit ./h

execute // here we trigger the scheduler to

Output format :

Hello world

Hello world

Hello world

Hello world

102563255..

102563255..

By pressing exit in the shell we can actually see the execution time and waiting time of each process.



1. Advance :

In advance we have to make priority queue i.e in this we have to use basic multi-level feedback or priority level queue to schedule our process.  
In this we have to use shared Queue with one more item as priority.  
  
  
  
  
  
  
Contribution :

Prince : input part, queue implementation, half scheduling algorithm, calculate waiting time,etc.

Prajil : shell part, enqueue part, half scheduling algorithm, calculate execution time, etc.

GITHUB : <https://github.com/Prince22378/OS-Assignments-2023/tree/main/Assignment-3>