MINI-PROJECT-3
REPORT

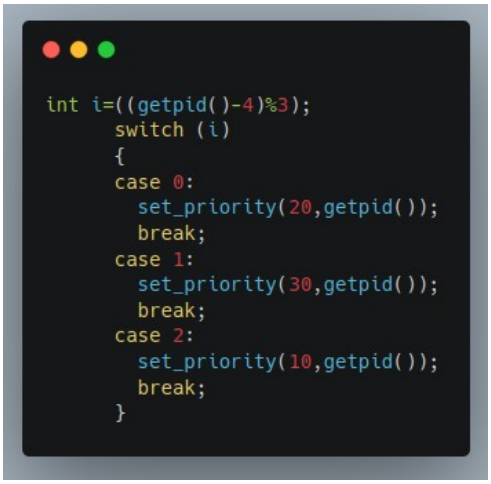
1) PRIORITY BASED SCHEDULER:

Implementation steps:

- Modify `proc` Structure: Update the `proc` structure to include variables for RTime, STime, WTime, SP, RBI, and DP.
- Initialize Process Values: Initialize these values appropriately in the `allocproc()` function and reset them when a process is scheduled.
- Modify the scheduler to use DP for process selection.
- Implement the `set_priority` system call. This should change the SP of the process, reset RBI to 25, and trigger rescheduling if the priority increases.
- Implement a user program called `setpriority` that uses the `set_priority` system call. This program should take process ID (pid) and priority as command-line arguments.

Modification in schedulertest function:

- I have set priorities of all nine processes scheduled in scheduler test as shown.
- This changes the static priority of processes.



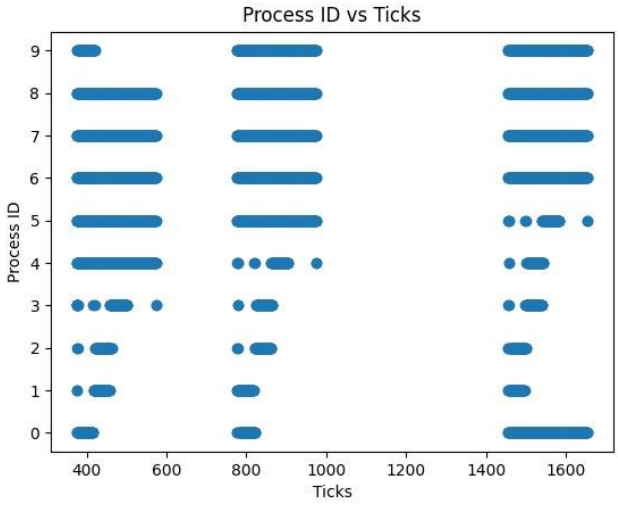
```
int i=((getpid()-4)%3);
      switch (i)
      {
      case 0:
        set_priority(20,getpid());
        break;
      case 1:
        set_priority(30,getpid());
        break;
      case 2:
        set_priority(10,getpid());
        break;
      }
```

Analysis:

- **Static Priority (SP):** An inherent importance of a process, SP is a user-defined priority.
  - Reduced SP levels signify more importance.
  - By utilising the set_priority system call, the user can configure SP.
  - Using SP, users can indicate which process they would like to prioritise.
- **Recent Behaviour Index(RBI):**
  - The Running Time (RTime), Sleeping Time (STime), and Waiting Time (WTime) components of the Recent Behaviour Index (RBI) are added together for weight.
  - In order to reflect a process's recent behaviour, RBI is utilised to modify DP.
  - Higher values of RBI indicate that a process has been waiting
  - or sleeping more than it has been running, potentially leading to a higher DP.
  - Lower values of RBI suggest that a process has been actively using the CPU, potentially leading to a lower DP**.**
- **Summary:**
  - SP and RBI integration forms a flexible priority system.
  - Users shape initial priority (SP) based on application needs.
  - RBI prevents process starvation by considering recent behavior.

- Scheduler balances SP and adjusts priorities dynamically via RBI. Considerations:
- Priority system effectiveness depends on workload and application diversity.
- Fine-tuning RBI weightings or SP range may be needed based on real-world usage and performance.

- Below is the graph plotted between pid of process and ticks.



- This shows that CPU bound processes are scheduled more frequently than input bound processes.

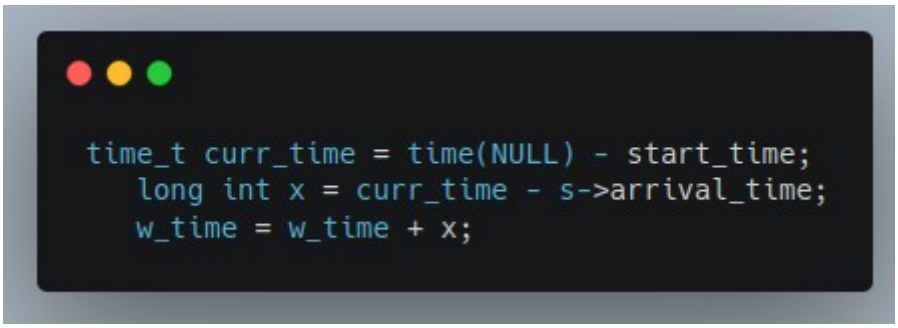
Time Analysis:

- For Round Robin:
  - Average rtime 15, wtime 121
- For PBS:
  - Average rtime 10, wtime 110
- This show that priority based scheduling is better than Round Robin Scheduling.

2)Cafe Sim:

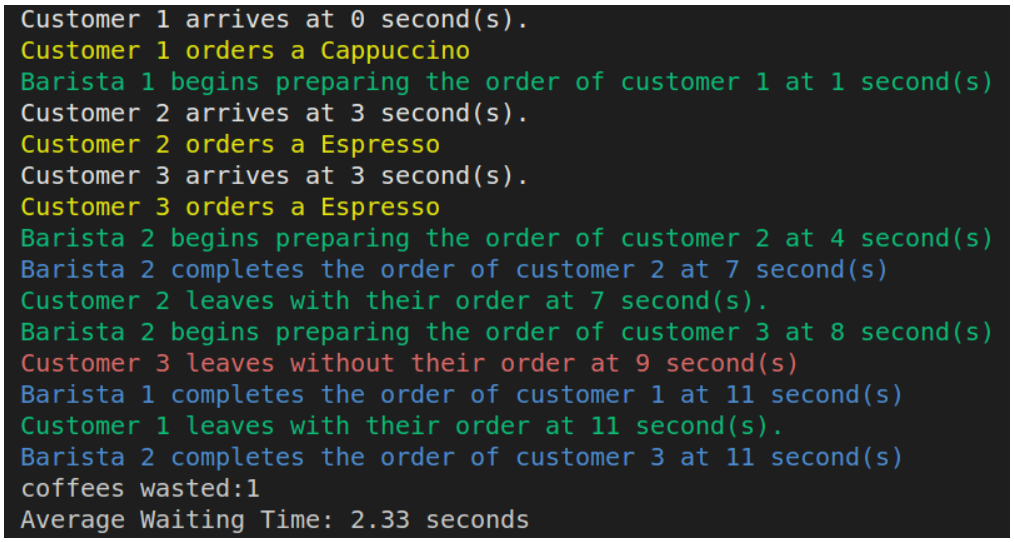
1)Waiting time:

- Average waiting time of the customer is calculated as follows.
- Every time a the barista takes the order, waiting time of the customer is updated.



```
time_t curr_time = time(NULL) - start_time;
   long int x = curr_time - s->arrival_time;
   w_time = w_time + x;
```

- Average waiting time of customers is w_time/no_of_customers.
- For the given test case Average waiting time of customer is displayed as follows.



2) Coffees Wasted:

- Every time a customer leaves without taking his/her order, no_of_coffees_wasted (declared globally) is incremented.
- This valued is printed finally.