BCSE – 3rd year – 1st Semester – 2024-25
Assignment – II [Q1, Q2: CO2, Q3-Q4: CO3, Q5: CO4]
Operating Systems Laboratory

1. Design a CPU scheduler for jobs whose execution profiles will be in a file that is to be read and appropriate scheduling algorithm to be chosen by the scheduler.

   Format of the profile:

   <Job id> <arrival time> <CPU burst(1) I/O burst(1) CPU burst(2) ……. >-1

   (Each information is separated by blank space and each job profile ends with -1. Lesser priority number denotes higher priority process with priority number 1 being the process with highest priority.) Example: 2 4 100 2 200 3 25 -1 7 1 8 60 10 ….. -1 etc.

   Testing:

   a. Create job profiles for 20 jobs and use three different scheduling algorithms (FCFS, Shortest Remaining Time Next (SRTN) and Round Robin (time slice: 25)).
   b. Compare the average waiting time, turnaround time of each process for the different scheduling algorithms.

2. Create child processes: X and Y.

   a. Each child process performs 10 iterations. The child process displays its name/id and the current iteration number, and sleeps for some random amount of time. Adjust the sleeping duration of the processes to have different outputs (i.e. another interleaving of processes' traces).

   b. Modify the program so that X is not allowed to start iteration *i* before process Y has terminated its own iteration *i-1*. Use semaphore to implement this synchronization.

3. Write a program for *p*-producer *c*-consumer problem, *p, c* >= 1. A shared circular buffer that can hold 25 items is there. Each producer process generates a number between 1 to 20 (along with the producer id) and deposits it in the buffer. The item deposited in the buffer looks like: <producerid 2digit number>, e.g., p103, where p1 is the producer id and 03 is the number generated by the producer. Each consumer process reads the number (e.g. 03) from the buffer and adds it to a shared variable TOTAL (initialized to 0). The constraint on the consumer is that, every number written by some producer should be read exactly once by exactly one of the consumers.

   The program reads in the value of *p* and *c* from the user, and forks *p* producers and *c* consumers. After all the producers and consumers have finished (the consumers exit after all the data produced by all producers have been read), the parent process prints the value of TOTAL. Test the program with different values of *p* and *c*.

4. Write a program for the Reader-Writer process for the following situations:

   a) Multiple readers and one writer: writer gets to write whenever it is ready (reader/s wait)

   b) Multiple readers and multiple writers: any writer gets to write whenever it is ready, provided no other writer is currently writing (and reader/s wait)

5. Implement the following applications using different IPC mechanisms. Your choice is restricted to Pipes, FIFOs, and Message Queues (Try to use different mechanisms for each program)
   a. Broadcasting weather information (one broadcasting process and more than one listeners)
   b. Telephonic conversation (between a caller and a receiver)
   c. Broadcasting information regarding pesticides for agricultural fields / prices of agricultural products for marketing with a farmer having the option of putting query (one broadcasting process and more than one listeners with option of calling back)

Last Date for submission: By First week of October 2024