Name: Heman Baral

CMSC 312

**Assignment 3 Report**

1. **First give an outline of the working portions of your code. For each part not functioning list them out.**

FIFO.C

My program can take 2 arguments number of producer and number of consumers. I have defined print job and print queue for 30. So, each user can submit up to 30 jobs and a queue can hold 30 jobs. In order to randomize job size, I have created a function that randomly generates job sizes for each user. Each of the job gets passed into user\_thread method in order to get it to the queue. I check if the queue is full, if not then that job will be added to the queue by calling job\_add function then it will be passed along to printer thread to deque the process by calling method delete\_job and rm\_queue to remove the queue after all the job Is done. I have current\_job method that works on current job that is assigned to work which is called from the main method.

SJF.C

My program can take 2 arguments number of producer and number of consumers. All the job are handled from the main method but I have created few other function that gets called in order to do job in organized fashion, assigning job, removing job from the queue, add jobs to the queue and check for queue is empty or not. Similar to FIFO I have I have created a function that randomly generates job sizes for each user. Each of the job gets passed into user\_thread method in order to get it to the queue. I check if the queue is full, if not then that job will be added to the queue by calling job\_add function then it will be passed along to printer thread to deque the process by calling delete\_job method. At the end I will call rm\_queue to remove delete the queue after all the jobs are done.

1. **Second: show the following: (a) logic used to identify the terminating condition, (b) how were the semaphores and other book-keeping variables shared between processes and threads, (c) what was done in the signal handler for graceful termination and (d) how did the FCFS and SJF implementations differ in terms of your usage of the buffer\_index variable from the producer- consumer code of Assignment-2.**

Logic used to delete a job

A screenshot of a cell phone

Description automatically generated

For semaphore: I have created sem\_wait to decrement the process and sem\_post to increment semaphores pointed by sem. I have declared full, empty and mutex which are used to add jobs to the queue, and I have initialized in main to single the end of current job.

For graceful termination I am removing the queue first and using sem\_destroy (&mutex), sem\_destroy (&empty), sem\_destroy (&full) commands.

1. **Third: show two sample runs of your code one for FCFS and other one for SJF. Just copy-paste the code output from the server.**

**FIFO output:**

**➜ Assignment 3** ./FIFO 2 4

Producer 259522448 added 426 to buffer

Producer 260059024 added 784 to buffer

Consumer ID:260595616 dequeue 260595612 426 from buffer

Consumer ID:261132192 dequeue 261132188 784 from buffer

Consumer ID:261668768 dequeue 261668764 784 from buffer

Producer 260059024 added 207 to buffer

Producer 259522448 added 985 to buffer

Consumer ID:262205344 dequeue 262205340 985 from buffer

Producer 260059024 added 569 to buffer

Producer 259522448 added 734 to buffer

Producer 260059024 added 273 to buffer

Producer 259522448 added 108 to buffer

Producer 260059024 added 240 to buffer

Producer 259522448 added 376 to buffer

Producer 260059024 added 817 to buffer

Producer 259522448 added 330 to buffer

Consumer ID:260595616 dequeue 260595612 569 from buffer

Consumer ID:261668768 dequeue 261668764 734 from buffer

Consumer ID:261132192 dequeue 261132188 273 from buffer

Consumer ID:262205344 dequeue 262205340 108 from buffer

Consumer ID:261132192 dequeue 261132188 240 from buffer

Consumer ID:260595616 dequeue 260595612 376 from buffer

Consumer ID:261668768 dequeue 261668764 817 from buffer

Consumer ID:262205344 dequeue 262205340 330 from buffer

Total Print Jobs: 12

Avarage Time in msec: 0.213250

Total Run Time: 2.559000

**SJF Output:**

**➜ Assignment 3** ./SJF 2 4

Producer 189177744 added 476 to buffer

Producer 189714320 added 341 to buffer

Consumer ID:191860640 dequeue 191860636 341 from buffer

Consumer ID:190250912 dequeue 190250908 476 from buffer

Consumer ID:191324064 dequeue 191324060 341 from buffer

Producer 189177744 added 392 to buffer

Consumer ID:190787488 dequeue 190787484 392 from buffer

Producer 189714320 added 142 to buffer

Producer 189177744 added 606 to buffer

Producer 189714320 added 526 to buffer

Producer 189714320 added 574 to buffer

Producer 189177744 added 534 to buffer

Producer 189714320 added 590 to buffer

Producer 189714320 added 295 to buffer

Producer 189714320 added 361 to buffer

Producer 189714320 added 355 to buffer

Producer 189714320 added 605 to buffer

Producer 189714320 added 501 to buffer

Consumer ID:191860640 dequeue 191860636 142 from buffer

Consumer ID:191324064 dequeue 191324060 355 from buffer

Consumer ID:190250912 dequeue 190250908 295 from buffer

Producer 189714320 added 305 to buffer

Consumer ID:190787488 dequeue 190787484 305 from buffer

Producer 189714320 added 825 to buffer

Producer 189714320 added 673 to buffer

Producer 189714320 added 890 to buffer

Consumer ID:191860640 dequeue 191860636 361 from buffer

Consumer ID:191324064 dequeue 191324060 501 from buffer

Consumer ID:190250912 dequeue 190250908 526 from buffer

Consumer ID:190787488 dequeue 190787484 534 from buffer

Consumer ID:191860640 dequeue 191860636 574 from buffer

Consumer ID:191324064 dequeue 191324060 590 from buffer

Consumer ID:190250912 dequeue 190250908 605 from buffer

Consumer ID:190787488 dequeue 190787484 606 from buffer

Consumer ID:191324064 dequeue 191324060 673 from buffer

Consumer ID:191860640 dequeue 191860636 673 from buffer

Consumer ID:190250912 dequeue 190250908 825 from buffer

Consumer ID:190787488 dequeue 190787484 890 from buffer

Total Print Jobs: 18

Average Time in msec: 0.190167

Total Run Time 3.423000

1. **Finally, show the execution time plots for FCFS and SJF for different values of number of producer processes and number of consumer threads. Plots will be good to show here or you can simply use a table format; vary both #producers and #consumers.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FIFO** |  | |  | |  | |  | |
| **USER (PRODUCER)** | **PRINTER(CONSUMER)** | | **TOTAL JOB** | | **AVG TIME** | | **TOTAL TIME** | |
| 2 | 10 | | 21 | | 0.261286 | | 5.487 | |
| 4 | 14 | | 50 | | 0.254280 | | 12.714 | |
| 6 | 18 | | 78 | | 0.183615 | | 14.322 | |
| 8 | 22 | | 99 | | 0.219343 | | 21.715 | |
| 10 | 24 | | 112 | | 0.173652 | | 19.449 | |
|  |  |  | |  | |  | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SJF** |  |  |  |  |
| **USER (PRODUCER)** | **PRINTER(CONSUMER)** | **TOTAL JOB** | **AVG TIME** | **TOTAL TIME** |
| 2 | 10 | 36 | 0.193500 | 6.966 |
| 4 | **14** | 51 | 0.215000 | 10.965 |
| 6 | 18 | 64 | 0.196203 | 12.557 |
| 8 | 22 | 99 | 0.188030 | 18.615 |
| 10 | 24 | 131 | 0.151901 | 19.899 |
|  |  |  |  |  |

1. **Two options:** 
   1. **3-D graph: you will just have a single 3-D graph for FCFS, and another 3-D graph for SJF.**
   2. **2-D graphs: 5 different graphs are needed for #producers = 2, 4, 6, 8, 10 respectively. In**

**each graph, plot execution time VS number of consumers (varied as 2, 4, 6, 8, 10) for FCFS and for SJF. So each 2-D graph will have two lines, one for FCFS and other one for SJF.**

**3-D Graph**

**2D Graph for Consumer = 10**

**2D Graph for Consumer = 14**

**2D Graph for Consumer = 18**

**2D Graph for Consumer = 22**

**2D Graph for Consumer = 24**