The shared variables between the producers and consumers are the Boolean done and the totalProduced and totalConsumed. TotalProduced is shared between the producers, and totalConsumed is shared between the consumers. The consumers are waiting for the Boolean done to be checked for true before it starts to consume from the circular Boolean. There is a circular buffer called buffer, this is an array of items(which is the sales

There are three semaphores in my circular buffer (***empty***, ***full*** and ***mutex***) . The ***empty*** semaphore lets the producers know there is still room left for the circular buffer. The ***full*** semaphore lets the consumers know that the circular buffer needs to have sales read from it. ***Mutex*** semaphore is what allows it to lock out access to the buffer for every other consumer/producer except for the consumer/producer that is currently accessing it. We used a triple nested loop statement that iterates through the two buffers (3,10),the three producers (2, 5,10) and the three consumers(2,5,10). There was a slight problem with running it in the console, so I ended up having it create a report as well. If you ran it multiple times without letting it finish it would corrupt one run with the previous run.

For Buffer = 3; Producer = 2; Consumer =2

Execution time was 11950 ms

For Buffer = 3; Producer = 2; Consumer =5

Execution time was 4619 ms

For Buffer = 3; Producer = 2; Consumer =10

Execution time was 2363 ms

For Buffer = 3; Producer = 5; Consumer =2

Execution time was 11485 ms

For Buffer = 3; Producer = 5; Consumer =5

Execution time was 4706 ms

For Buffer = 3; Producer = 5; Consumer =10

Execution time was 2358 ms

For Buffer = 3; Producer = 10; Consumer =2

Execution time was 11858ms

For Buffer = 3; Producer = 10; Consumer =5

Execution time was 4654ms

For Buffer = 3; Producer = 10; Consumer =10

Execution time was 2371ms

For Buffer = 10; Producer = 2 ; Consumer =2

Execution time was 11775ms

For Buffer = 10; Producer = 2; Consumer =5

Execution time was 4799ms

For Buffer = 10; Producer = 2; Consumer =10

Execution time was 2371ms

For Buffer = 10; Producer = 5 ; Consumer =2

Execution time was 11751ms

For Buffer = 10; Producer = 5; Consumer =5

Execution time was 4784ms

For Buffer = 10; Producer = 5; Consumer =10

Execution time was 2359ms

For Buffer = 10; Producer = 10 ; Consumer =2

Execution time was 11683ms

For Buffer = 10; Producer = 10; Consumer =5

Execution time was 4632ms

For Buffer = 10; Producer = 10; Consumer =10

Execution time was 2344ms

It appears that when there were more consumers the execution time was drastically decreased. This is shown in every set, for example :

For Buffer = 10; Producer = 10 ; Consumer =2

Execution time was 11683ms

For Buffer = 10; Producer = 10; Consumer =5

Execution time was 4632ms

For Buffer = 10; Producer = 10; Consumer =10

Execution time was 2344ms

This shows that when the consumer is at 2 , the execution time is roughly 11700ms, while when the consumer is at 4632ms . Proportionally, the time between consumer 2 and consumer 5 is 2/5 * the execution time for consumer 2 which equals roughly the execution time for consumer 5. The same thing happens with execution time for 5 and the execution time for consumer 10, it's roughly twice as fast as execution time of 5. As for the days, months, register# and sales amount are all randomly generated.

In conclusion, our program demonstrates the producer-consumer problem via semaphores with synchronized access to a shared circular buffer. There is also a copy of the outputs printed in the sampleOutput.txt file that is located inside this folder.