Our ProductConsumer file starts with a request struct that contains an id and length. We then have a queue that we created ourselves complete with enqueue and dequeue.

We first ask the user to insert values for consumer_ct, max_req_length, producer_max_sleep_duration. we then allocate the necessary size for the queue. We then create a new master thread and create the appropriate number of slave threads and allocate a specific ID to each of the slave threads upon creation.
Once the threads are created, they will run their respective thread code.
For Master Thread, it runs the master_thread() code. Basically, the infinite while loop keeps the single master thread alive. One iteration is: it creates a new Request at random length. It then waits until the variables empty_count, mutex are available for locking, thus enabling it exclusive access to the queue, at which it will enqueue the new request. It then releases it's lock on the resources. Because of empty_count, if it ever hits zero (meaning no more empty spaces), it will stall the thread at wait(&empty_count) line until the queue opens up another spot.

For Slave thread, it runs the slave_thread() code. The slave thread is suppose to dequeue from the queue and simply report what it dequeue'd. This process simply waits for the fill_ct and mutex to be available. The fill count represents the number of requests in the queue. If there are no requests, the thread will wait there before anything happens. It will wait until something is put into the queue. After it's done waiting and processing the next request, the thread releases its lock on the variables.