1. The main function contains calls to exit() (line 66) and pthread_exit() (line 80). How will the effect of these two calls differ when they are executed?
2. The main function calls pthread_join() (line 77) with the parameter thread_return. Where does the value stored in thread_return come from when the consumer_thread is joined?
3. Where does the value stored in thread_return come from if the joined thread terminated by calling pthread_exit instead of finishing normally?
4. On the same call to pthread_join() (line 77), what will it do if the thread being joined (consumer_thread, in this case) finishes before the main thread reaches the that line of code (line 77)?
5. In this program, the main thread calls pthread_join() on the threads it created. Could a different thread call pthread_join() on those threads instead? Could a thread call pthread_join() on the main thread (assuming it knew the main thread's thread ID - i.e. pthread_t)?
6. The consumer_routine function calls sched_yield() (line 180) when there are no items in the queue. Why does it call sched_yield() instead of just continuing to check the queue for an item until one arrives?

1. Pthread_exit() function will terminate the calling thread and give a return value, which means other threads can still work well. exit() function will terminate all the threads, even the calling process. And exit() function will also flush up all the buffer and data.

2. The thread_return is a void* pointer which it comes from the return value of the consumer_routine() function. In the end of that function, it returns a (void*) count which stands for the amount of the characters consumer_thread eat. And the thread_return gets its value after the end of the consumer_thread.

3. The value stored in thread_return comes from the pthread_exit() function if a thread terminate by pthread_exit() function. Pthread_exit() function will return the value which is supplied to it. For example, if there is a pthread_exit("It is a pthread exit") code to terminate a thread, the thread_return value would be "It is a pthread exit".

4. It will return immediately.

5. Yes. Other threads could also call the thread_join. Because all the threads in a process are peers and these threads are created as joinable by default.

6. Calling sched_yield() could let other threads that priority are higher or equal to the calling thread run first. Because once there are no characters in the queue, it is meaningless to continue running the consumer_thread. It should let the producer_thread run first in order to create some characters.