CSE 7343 Operating Systems
Program 6 – Producer / Consumer using Messages
Due: Friday, May 4 – Late programs will not be accepted (CSE 7343 Students Only)

You are to write a c program named msgQueue that implements the producer / consumer problem using a UNIX message queue as the data structure shared by a single producer and three consumers. Your program is to create a child process and the child process will in turn create three threads. The parent process is to be the producer and the three threads will be the consumers. The number of items, N, that are to be produced will be provided to the program via the command line. Use a default value of 10 if N is not provided.

After spawning the child process the parent will enter an N-iteration loop. In each iteration of the loop the parent will do the following:

1) Generate a random number, R, in the range of 0-999.
2) Send a message containing R.
3) Add R to a running total of all values produced.
4) Display the string “Producer produced a R”.
5) Put the producer to sleep for 0-1 seconds using sleep(rand()%2).

After the N iterations have completed display the string "Total produced = XXXX” (where XXXX is the sum of all R values produced) and wait for the child to terminate. It is the parent’s responsibility to create and destroy the queue.

The child process will create three consumer threads, 0, 1 and 2. Each thread will enter an N/3 iteration loop (be careful if (N%3) != 0). In each iteration of the loop each consumer thread will do the following:

1) Read a message containing a value, C, consumed from the queue.
2) Add C to a running total maintained by each consumer thread.
3) Display the string “Consumer thread Z consumed a C” where Z is the thread number – 0,1 or 2.
4) Put the consumer thread to sleep for 1-3 seconds using sleep((rand()%3)+1)

After N/3 iterations display the string "Total consumed by consumer thread Z = YYYY" where YYYY is the sum of all N/3 values consumed. Write your program in such a way that it is easy to change the number of consumer threads.

If your program is working correctly the producer's total will be equal to the sum of the three consumer's totals. A good explanation of UNIX messages may be found here. Include in your in-line comments an explanation of how mutual exclusion and synchronization are achieved between the producer and consumers.