COP 4600.001 Operating Systems – Project 3

A shared Protected Circular Queue and Communication between threads

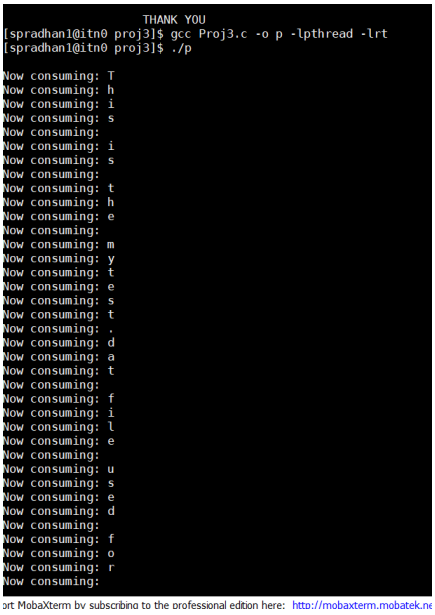
Subhrajyoti Pradhan U79333962

Objective –

The purpose is to learn how to use semaphores to protect a limited size resource. A circular buffer with 15 positions (each position stores 1 character) is to be used to communicate information between two threads (producer and consumer). The producer thread will read characters, one by one from a file and place it in the buffer and continue to do that until the "end-of-file" (EOF) marker is reached. The name of the file must be "mytest.dat" when you are submitting the program – of course you can use your own file while individually testing your program. There should be no more than 150 characters in the file. The producer must inform the consumer when it has finished placing the last character in the buffer. The producer could do this by placing a special character for example, "*" in the shared buffer or by using a shared memory flag that the producer sets to true and the consumer reads at the appropriate time. Consumer thread will read the characters, one by one, from the shared buffer and print it to the screen. A parent process will create both producer and consumer threads and will wait until both are finished to destroy semaphores. The consumer should run slower than producer. We place a one second sleep in the consumer thread between "reads" from the shared memory.

Observation –

Sample output



Analysis of Observation –

Results were according to expectation. The consumers. The test file was read into the buffer by the producer and the consumer placed individual outputs into the stdout. The memory was protected by the semaphore and every thread had mutually exclusive access to the buffer. Overall the threads were able to signal each other as needed.