Assignment 4

To run use the Makefile which generates the object files and then use ./a.out and ./b.out to run them

Patch file has been attached and is named newPatch.patch

Syscall Working

- I've created 2 syscalls writer and reader and 2 function enqueue and dequeue in sys.c
- These syscalls tackle the producer consumer problem by maintaining a buffer array of size 7
- The consumer or producer whoever enters the critical section first will create the buffer which is initially null
- Then the consumer waits if the buffer is empty and the producer proceeds to add elements
- Once the buffer is full the producer waits and the consumer proceeds to consume elements till it is not empty which means at a time 7 elements are moved
- For the implementation a cyclic index is used with a normal array of unsigned long ints and sempahores to attain mutual exclusion
- There are 3 semaphores empty, full and mutex.
- Empty is initialized with 7, full with 0 and mutex with 1 in producer as well as consumer whoever reaches first initializes them
- Enqueue and Dequeue Function are used by Writer and Reader respectively to add and remove elements from the buffer
- down_interruptible() is used over down() as during research I realized down is deprecated in favour of down_interruptible()
- down_interruptible() is essentially wait so it acts like it and waits for empty and mutex semaphore in enqueue and full and mutex in dequeue respectively of course based on the .value property
- Similarly up() works in the same fashion
- %7 in index is used to implements cyclic index as the buffer is reused
- These syscalls are then added in table_64 file

Testcodes

- There are 2 test files test.c and test2.c
- test.c is analogous to the producer and test2.c is analogous to the consumer
- Inside test.c random characters are generated by reading from /dev/urandom into the buffer
- Inside num the elements read into the buffer array are used to generate the random long by using a mizture of bit shifts (bit manipulation) and bit operation to concatenate the entries of the buffer array
- Using the writer syscall it is then sent to the buffer cyclic array is sys.c
- At the same time test2.c is executed which reads from the cyclic buffer array and stores it using the syscall
- It then returns this extracted value which is retrieved as the return value of the syscall and printed in test2.c
- Adding sleep statements makes this entire proces more apparent
- Noticebly is test2.c is started first we can see no outputs till test.c is run while on the contrary we can see 7 elements have been produced if we start with test.c which then waits for the producer