 **ASSIGNMENT3**



*Multiple Reader-Writer with synchronization*

Tushar|2018201

**DESCRIPTION OF CODE**

* void \*reader(void \*arg) takes input for reader no., and tries to read from the shared memory whenever data is present and no other writer is writing.
* void \*writer(void \*arg) takes input for writer no., and tries to write in the shared memory whenever no other writer is writing and no reader is reading.
* int main()initializes the shared memory and spawns multiple reader-writer threads.



**IMPLEMENTATION OF MULTIPLE READER-WRITERS**

* Shared memory is used for inter-process communication

between two threads.

* Whenever a writer is writing in shared memory, no other writer can write at that location and no reader can read it. This is made sure using read and write semaphores.

• Whenever a reader is reading from shared memory, other readers can read from shared memory, but no writer can write there.

* Read semaphore is also used to correctly manage the reader count.

**INPUT FORMAT**

***No as such input ix required, but you can increase reader-writer threads by simply changing a loop.***

**EXPECTED OUTPUT**

***For Writer:***

**“Written by Writer <writer no>”**

**“Writer 4 can't write, someone is either writing or reading”**

***For Readers:***

**“Data read by reader <reader no> is written by writer <writer no>”**

**“Reader 2 can't read, someone is writing”**

**ERROR HANDLED**

***No Multiple writing at same time.***

* Multiple writers can’t write at same location simultaneously.

***Readers can’t read while a writer is writing.***

* Reader can’t read while a writer is writing.

***Reader can’t read while data is not present.***

* Reader waits while no data is present in the shared memory.

***Semaphore initialization error handled.***

* Handles error if semaphore isn’t initialised correctly.

***Shared memory is destroyed after using it.***

**DESCRIPTION OF THE SYSTEM**

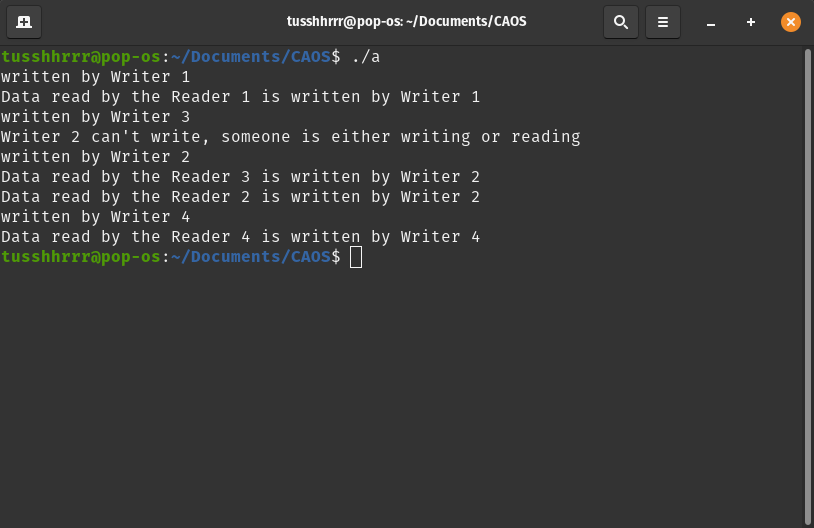
This *Multiple Reader-Writer with synchronization* works on unix/linux based systems.

**TESTING THE SYSTEM CALL**

• Compile rw.c using make

* Open user application in multiple terminal windows using ./rw
* Multiple readers and writers will read and write at same location and corresponding outputs will be printed on terminal.

**WORKING OF MULTIPLE READER-WRITER WITH SYNCHRONIZATION**



**REFERENCES**

<https://gist.github.com/rajabishek/6209a575f00b122fe490>

<https://www.usebackpack.com/resources/23423/download?1571918324>