**Experiment 8**

**Aim:** Write a program for the implementation of Readers Writers problem.

**Theory:** The readers-writers problem relates to an object such as a file that is shared between multiple processes. Some of these processes are readers i.e. they only want to read the data from the object and some of the processes are writers i.e. they want to write into the object.

The readers-writers problem is used to manage synchronization so that there are no problems with the object data. For example - If two readers access the object at the same time there is no problem. However, if two writers or a reader and writer access the object at the same time, there may be problems.

To solve this situation, a writer should get exclusive access to an object i.e., when a writer is accessing the object, no reader or writer may access it. However, multiple readers can access the object at the same time. This can be implemented using semaphores.

**Source Code:**

#include <pthread.h>

#include <semaphore.h>

#include <stdio.h>

/\*

This program provides a possible solution for first readers writers problem using mutex and semaphore.

I have used 10 readers and 5 producers to demonstrate the solution.

\*/

sem\_t wrt;

pthread\_mutex\_t mutex;

int cnt = 1;

int numreader = 0;

void \*writer(void \*wno)

{

sem\_wait(&wrt);

cnt = cnt\*2;

printf("Writer %d modified cnt to %d\n",(\*((int \*)wno)),cnt);

sem\_post(&wrt);

}

void \*reader(void \*rno)

{

// Reader acquire the lock before modifying numreader

pthread\_mutex\_lock(&mutex);

numreader++;

if(numreader == 1) {

sem\_wait(&wrt); // If this id the first reader, then it will block the writer

}

pthread\_mutex\_unlock(&mutex);

// Reading Section

printf("Reader %d: read cnt as %d\n",\*((int \*)rno),cnt);

// Reader acquire the lock before modifying numreader

pthread\_mutex\_lock(&mutex);

numreader--;

if(numreader == 0) {

sem\_post(&wrt); // If this is the last reader, it will wake up the writer.

}

pthread\_mutex\_unlock(&mutex);

}

int main()

{

pthread\_t read[10],write[5];

pthread\_mutex\_init(&mutex, NULL);

sem\_init(&wrt,0,1);

int a[10] = {1,2,3,4,5,6,7,8,9,10}; //Just used for numbering the producer and consumer

for(int i = 0; i < 10; i++) {

pthread\_create(&read[i], NULL, (void \*)reader, (void \*)&a[i]);

}

for(int i = 0; i < 5; i++) {

pthread\_create(&write[i], NULL, (void \*)writer, (void \*)&a[i]);

}

for(int i = 0; i < 10; i++) {

pthread\_join(read[i], NULL);

}

for(int i = 0; i < 5; i++) {

pthread\_join(write[i], NULL);

}

pthread\_mutex\_destroy(&mutex);

sem\_destroy(&wrt);

return 0;

}

**Output:**

Reader 7: read cnt as 1

Reader 8: read cnt as 1

Reader 9: read cnt as 1

Reader 10: read cnt as 1

Writer 1 modified cnt to 2

Writer 2 modified cnt to 4

Writer 3 modified cnt to 8

Writer 4 modified cnt to 16

Writer 5 modified cnt to 32

Reader 6: read cnt as 32

Reader 5: read cnt as 32

Reader 4: read cnt as 32

Reader 3: read cnt as 32

Reader 2: read cnt as 32

Reader 1: read cnt as 32