S.No: 14	Exp. Name: ***Implement the solutions for Readers-Writers problem using inter-process communication technique -Semaphore.***	**Date:2025-04-13**

Aim:

Aim: Implement the solutions for the Readers-Writers problem using the inter-process communication technique -Semaphore.

Description: Consider a situation where we have a file shared between many people.

- If one of the people tries editing the file, no other person should be reading or writing at the same time, otherwise changes will not be visible to him/her.
- However if some person is reading the file, then others may read it at the same time.

Precisely in OS we call this situation as the **readers-writer problem**
Problem parameters:

- One set of data is shared among a number of processes
- Once a writer is ready, it performs its write. Only one writer may write at a time
- If a process is writing, no other process can read it If at least one reader is reading, no other process can write
- Readers may not write and only read

Solution when Reader Has the Priority over Writer
Here priority means, no reader should wait if the share is currently open for reading.
Three variables are used: **mutex, wrt, readcnt** to implement solution

1. **semaphore** mutex, wrt; // semaphore **mutex** is used to ensure mutual exclusion when readcnt is updated i.e. when any reader enters or exit from the critical section and semaphore wrt is used by both readers and writers
2. int reading; // reading tells the number of processes performing read in the critical section, initially 0

Functions for semaphore :

- – wait(): decrements the semaphore value.
- – signal(): increments the semaphore value.

Writer process:
1. Writer requests the entry to critical section.

Source Code:

ReadersWriters.c

```
#include<stdio.h>
#include<conio.h>
#include<stdbool.h>
struct semaphore
{
int mutex;
int rcount;
int rwait;
bool wrt;
};
void addR(struct semaphore *s)
{
   if(s->mutex == 0 && s->rcount == 0)
   {
      printf("Sorry, File open in Write mode.\nNew Reader added to queue.\n");
      s->rwait++;
   }
   else
   {
      printf("Reader Process added\n");
      s->rcount++;
      s->mutex--;
   }
   return;
}
void addW(struct semaphore *s)
{
   if(s->mutex == 1)
   {
      s->mutex--;
      s->wrt=1;
      printf("Writer Process added\n");
   }
   else if(s->wrt)
      printf("Sorry, Writer already operational\n");
   else
      printf("Sorry, File open in Read mode\n");
   return;
}
void remR(struct semaphore *s)
{
   if(s->rcount == 0)
      printf("No readers to remove\n");
   else
   {
      printf("Reader Removed\n");
      s->rcount--;
      s->mutex++;
   }
   return;
}
void remW(struct semaphore *s)
{
   if(s->wrt==0)
      printf("No Writer to Remove\n");
   else
      printf("Writer Removed\n");
   s->mutex++;
   s->wrt=0;
   if(s->rwait!=0)
   {
      s->rcount=s->rwait;
      s->rwait=0;
      printf("%d waiting Readers Added.",s->rcount);
   }
}
int main()
{
   struct semaphore S1={1,0,0};
   while(1)
      {
```

```
iter\n5.Exit\nChoice: ");
         int ch;
         scanf("%d",&ch);
         switch(ch)
            {
               case 1: addR(&S1);
                  break;
               case 2: addW(&S1);
               break;
               case 3: remR(&S1);
               break;
               case 4: remW(&S1);
               break;
               case 5: printf("\tGoodBye!");
               return 0;
               default: printf("Invalid Entry!\n");
               continue;
            }
         printf("<<<<<< Current Status >>>>>>\n\tMutex: %d\n\tActive Readers: %d\n\tW
aiting Readers: %d\n\tWriter Active: %s\n",S1.mutex,S1.rcount,S1.rwait,(S1.mutex==0 &
& S1.rcount==0)?"YES" : "NO");
      }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Options :- 1
1.Add Reader 1
2.Add Writer 1
3.Remove Reader 1
4.Remove Writer 1
5.Exit 1
Choice: 1
Reader Process added 2
<<<<<< Current Status >>>>>> 2
Mutex: 0 2
Active Readers: 1 2
Waiting Readers: 0 2
Writer Active: NO 2
Options :- 2
1.Add Reader 2
2.Add Writer 2
3.Remove Reader 2
4.Remove Writer 2
5.Exit 2
Choice: 2
Sorry, File open in Read mode 3
<<<<<< Current Status >>>>>> 3
Mutex: 0 3
Active Readers: 1 3

```
        Waiting Readers: 0 3
        Writer Active: NO 3
Options :- 3
1.Add Reader 3
2.Add Writer 3
3.Remove Reader 3
4.Remove Writer 3
5.Exit 3
Choice:  3
Reader Removed 4
<<<<<< Current Status >>>>>> 4
        Mutex: 1 4
        Active Readers: 0 4
        Waiting Readers: 0 4
        Writer Active: NO 4
Options :- 4
1.Add Reader 4
2.Add Writer 4
3.Remove Reader 4
4.Remove Writer 4
5.Exit 4
Choice:  4
No Writer to Remove 5
<<<<<< Current Status >>>>>> 5
        Mutex: 2 5
        Active Readers: 0 5
        Waiting Readers: 0 5
        Writer Active: NO 5
Options :- 5
1.Add Reader 5
2.Add Writer 5
3.Remove Reader 5
4.Remove Writer 5
5.Exit 5
Choice:  5
        GoodBye!
```

Test Case - 2

User Output

```
Options :- 1
1.Add Reader 1
2.Add Writer 1
3.Remove Reader 1
4.Remove Writer 1
5.Exit 1
Choice:  1
Reader Process added 6
<<<<<< Current Status >>>>>> 6
        Mutex: 0 6
        Active Readers: 1 6
        Waiting Readers: 0 6
        Writer Active: NO 6
Options :- 6
1.Add Reader 6
2.Add Writer 6
3.Remove Reader 6
4.Remove Writer 6
5.Exit 6
Choice:  6
Invalid Entry! 2
Options :- 2
1.Add Reader 2
2.Add Writer 2
3.Remove Reader 2
4.Remove Writer 2
5.Exit 2
Choice:  2
Sorry, File open in Read mode 4
<<<<<< Current Status >>>>>> 4
        Mutex: 0 4
        Active Readers: 1 4
        Waiting Readers: 0 4
        Writer Active: NO 4
Options :- 4
1.Add Reader 4
2.Add Writer 4
3.Remove Reader 4
4.Remove Writer 4
5.Exit 4
Choice:  4
No Writer to Remove 3
<<<<<< Current Status >>>>>> 3
        Mutex: 1 3
        Active Readers: 1 3
        Waiting Readers: 0 3
        Writer Active: NO 3
Options :- 3
1.Add Reader 3
2.Add Writer 3
3.Remove Reader 3
4.Remove Writer 3
5.Exit 3
Choice:  3
Reader Removed 1
<<<<<< Current Status >>>>>> 1
        Mutex: 2 1
        Active Readers: 0 1
        Waiting Readers: 0 1
        Writer Active: NO 1
Options :- 1
1.Add Reader 1
2.Add Writer 1
3.Remove Reader 1
```

```
4.Remove Writer 1
5.Exit 1
Choice:  1
Reader Process added 5
<<<<<< Current Status >>>>>> 5
        Mutex: 1 5
        Active Readers: 1 5
        Waiting Readers: 0 5
        Writer Active: NO 5
Options :- 5
1.Add Reader 5
2.Add Writer 5
3.Remove Reader 5
4.Remove Writer 5
5.Exit 5
Choice:  5
        GoodBye!
```