Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

CSE-316: OPERATING SYSTEMS

Programming Assignment

Submitted to:

Dr. Baljit Singh Saini

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

Problem Statement

Write a multi-threaded C program that gives readers priority over writers concerning a shared (global) variable. Essentially, if any readers are waiting, then they have priority over writer threads -- writers can only write when there are no readers. This program should adhere to the following constraints: Multiple readers/writers must be supported (5 of each is fine) Readers must read the shared variable X number of times Writers must write the shared variable X number of times Readers must print: The value read The number of readers present when value is read Writers must print:

The written value The number of readers present were when value is written (should be 0) Before a reader/writer attempts to access the shared variable it should wait some random amount of time Note: This will help ensure that reads and writes do not occur all at once Use pthreads, mutexes, and condition variables to synchronize access to the shared variable

Approach to solution

Approach was to make a program which can adhere multiple users and to learn the basic about library so that things can be done smoothly.

Github Link:

https://github.com/rgrajat085/multithreaded-reader-writer-program

Code

```
#include<stdio.h>

#include<pthread.h>

#include<stdlib.h>

#include<unistd.h>


pthread_mutex_t mutex, wrt;

int s, rcount = 0;


void *writer(void *arg){

    pthread_mutex_lock(&wrt);
```

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

```
    int n = rand() % 10;
    int d = ((int)arg);
    printf("--------------------------------------------------\n");
    printf("W%d Wait for Random time between 0ns and 10ns = %d\n", d, n);
    sleep(n);
    printf("Enter the number of time W%d want to write:\n", d);
    int t;
    scanf("%d", &t);
    printf("Now W%d is writing... i.e. ADDING...\n", d);
    int j;
    for(j=0; j<t; j++){
        printf("Enter the %dth INTEGER value to write:\n", (j+1));
        int u;
        scanf("%d", &u);
        s = s + u;
    }
    printf("UPDATED value of Shared variable = %d \n", s);
    printf("--------------------------------------------------\n");
    pthread_mutex_unlock(&wrt);
}


void *reader(void *arg){
    //Entry Part
    pthread_mutex_lock(&mutex);
    rcount++;
    if(rcount==1){
        pthread_mutex_lock(&wrt);//No writer should come
    }
    pthread_mutex_unlock(&mutex);//so next reader can come
    //Exit Part
    int n = rand() % 10;
    int d = ((int)arg);
    printf("R%d wait for Random time between 0ns and 10ns = %d\n", d, n);
```

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

```
    sleep(n);
    printf("Enter the number of time R%d want to read:\n", d);
    int t;
    scanf("%d", &t);
    printf("Now R%d is reading....\n", d);
    int j;
    for(j=0; j<t; j++){
        printf("R%d read the shared value = %d\n", d, s);
    }
    printf("Number of Readers present = %d\n", rcount);
    pthread_mutex_lock(&mutex);
    rcount--;
    if(rcount==0){//Now writer can come if they want
        pthread_mutex_unlock(&wrt);
    }
    pthread_mutex_unlock(&mutex);
}

void main(){
    printf("Enter the 'INTEGER' Initial value of share variable: \n");
    scanf("%d", &s);
    printf("---------------------------------------------\n");
    int rn, wn, i;
    printf("Enter the no. of Reader:\n");
    scanf("%d", &rn);
    for(i=0; i<rn; i++){
        printf("R%d\n", i);
    }
    printf("---------------------------------------------\n");
    printf("Enter the no. of Writer:\n");
    scanf("%d", &wn);
    for(i=0; i<wn; i++){
        printf("W%d\n", i);
```

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

```
    }
    printf("---------------------------------------------\n");

    pthread_t r[rn], w[wn];
    pthread_mutex_init(&wrt, NULL);
    pthread_mutex_init(&mutex, NULL);

    if(rn<0 || wn<0){
        printf("Sorry: You have Entered NEGATIVE number of READER | WRITER\n");
        printf("Program is Terminating....\n");
        return;
    }else if(rn == 0){
        printf("Sorry: You have not taken any READER\n");
        printf("No READER thread will be creaded\n");
    }else if(wn == 0){
        printf("Sorry: You have not taken any WRITER\n");
        printf("No WRITER thread will be creaded\n");
    }else{
        printf("Thread Creating....\n");
    }
    printf("---------------------------------------------\n");

    if(wn==rn){
        for(i=0; i<wn; i++){
            pthread_create(&w[i], NULL, &writer, (int *)i);
            pthread_create(&r[i], NULL, &reader, (int *)i);
        }
        for(i=0; i<wn; i++){
            pthread_join(w[i], NULL);
            pthread_join(r[i], NULL);
        }
    }else if(wn>rn){
        for(i=0; i<rn; i++){
```

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

```
                pthread_create(&w[i], NULL, &writer, (int *)i);
                pthread_create(&r[i], NULL, &reader, (int *)i);
            }
            for(i=rn; i<wn; i++){
                pthread_create(&w[i], NULL, &writer, (int *)i);
            }
            for(i=0; i<rn; i++){
                pthread_join(w[i], NULL);
                pthread_join(r[i], NULL);
            }
            for(i=rn; i<wn; i++){
                pthread_join(w[i], NULL);
            }
        }else{
            for(i=0; i<wn; i++){
                pthread_create(&w[i], NULL, &writer, (int *)i);
                pthread_create(&r[i], NULL, &reader, (int *)i);
            }
            for(i=wn; i<rn; i++){
                pthread_create(&r[i], NULL, &reader, (int *)i);
            }
            for(i=0; i<wn; i++){
                pthread_join(w[i], NULL);
                pthread_join(r[i], NULL);
            }
            for(i=wn; i<rn; i++){
                pthread_join(r[i], NULL);
            }
        }
        printf("-------------After joining the thread---------\n");
        printf("Final value of share variable = %d\n", s);
}
```

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

Complexity of Code:

O(c(rn+wn))

Where, rn=number of readers

wn=number of writers

c=very small constant of thread functions

Explanation for above code

C does not contain any built-in support for multithreaded applications. Instead, it relies entirely upon the operating system to provide this feature.

This tutorial assumes that you are working on Linux OS and we are going to write multi-threaded C program using POSIX. POSIX Threads, or Pthreads provides API which are available on many Unix-like POSIX systems

pthread_create creates a new thread and makes it executable. This routine can be called any number of times from anywhere within your code. Here is the description of the parameters.

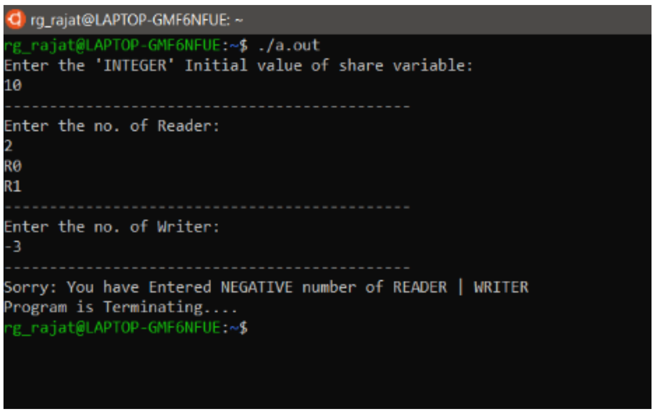
Parameter	Description
thread	An opaque, unique identifier for the new thread returned by the subroutine.
attr	An opaque attribute object that may be used to set thread attributes. You can specify a thread attributes object, or NULL for the default values.
start_routine	The C routine that the thread will execute once it is created.
arg	A single argument that may be passed to start_routine. It must be passed by reference as a pointer cast of type void. NULL may be used if no argument is to be passed.

The maximum number of threads that may be created by a process is implementation dependent. Once created, threads are peers, and may create other threads. There is no implied hierarchy or dependency between threads.

Snapshots and Explanations

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

1) When the User Enter Number of 'WRITER' is Negative



```
rg_rajat@LAPTOP-GMF6NFUE: ~
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
2
R0
R1
---------------------------------------------
Enter the no. of Writer:
-3
---------------------------------------------
Sorry: You have Entered NEGATIVE number of READER | WRITER
Program is Terminating....
rg_rajat@LAPTOP-GMF6NFUE:~$
```

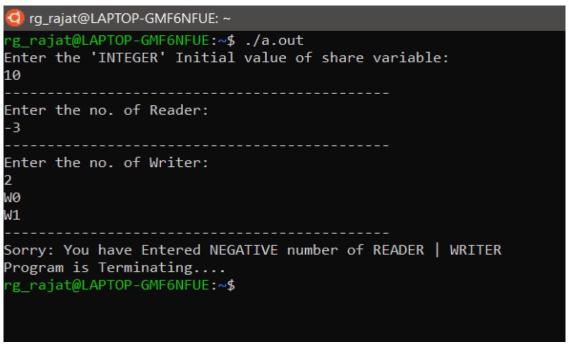
In this case

i. No WRITER and READER thread will be created

ii. No read or write operation done

iii. Program will be terminated

2) When the User Enter Number of 'READER' is Negative



```
rg_rajat@LAPTOP-GMF6NFUE: ~
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
-3
---------------------------------------------
Enter the no. of Writer:
2
W0
W1
---------------------------------------------
Sorry: You have Entered NEGATIVE number of READER | WRITER
Program is Terminating....
rg_rajat@LAPTOP-GMF6NFUE:~$
```

In this case

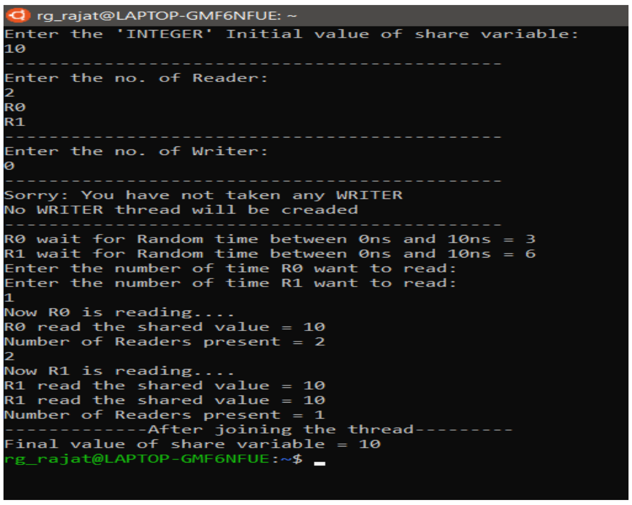
i. No WRITER and READER thread will be created

ii. No read or write operation done

iii. Program will be terminated

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

3) When Number of ‘WRITER=0’ is taken as input



```
rg_rajat@LAPTOP-GMF6NFUE: ~
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
2
R0
R1
---------------------------------------------
Enter the no. of Writer:
0
---------------------------------------------
Sorry: You have not taken any WRITER
No WRITER thread will be creaded
---------------------------------------------
R0 wait for Random time between 0ns and 10ns = 3
R1 wait for Random time between 0ns and 10ns = 6
Enter the number of time R0 want to read:
Enter the number of time R1 want to read:
1
Now R0 is reading....
R0 read the shared value = 10
Number of Readers present = 2
2
Now R1 is reading....
R1 read the shared value = 10
R1 read the shared value = 10
Number of Readers present = 1
-------------After joining the thread---------
Final value of share variable = 10
rg_rajat@LAPTOP-GMF6NFUE:~$ 
```

In This Case:

i. Only READER thread will be created

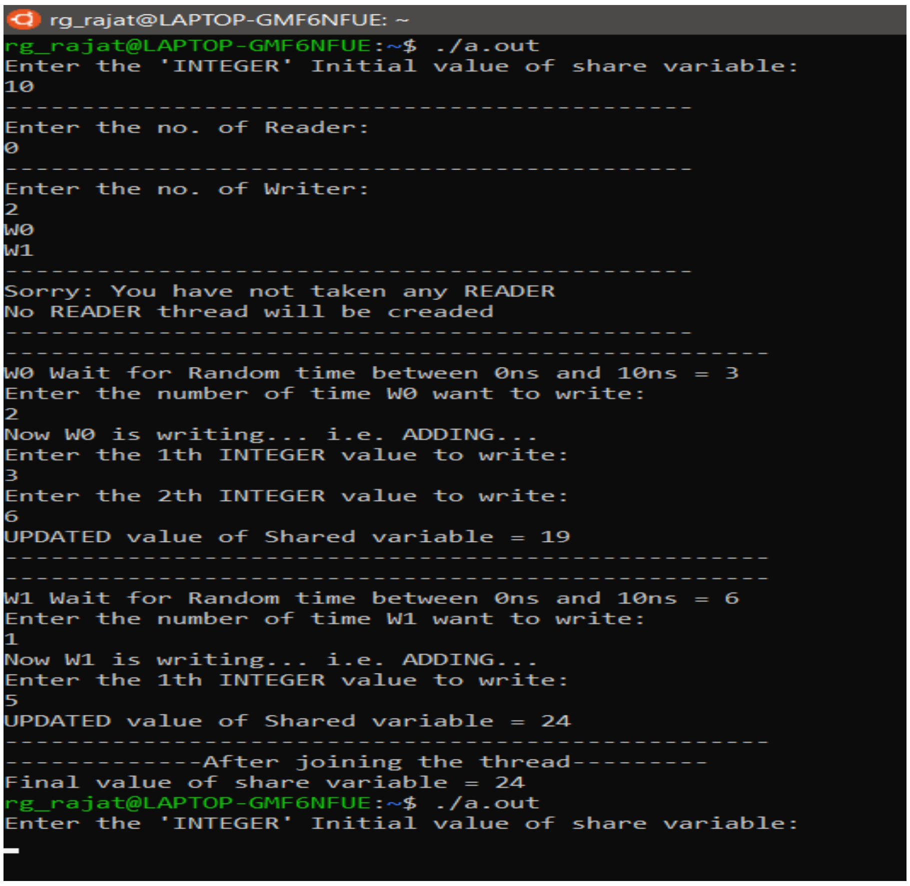
ii. No WRITER thread is created

iii. No updating of the Shared Variable

iv. Only READER is reading the shared variable. From the example given above R1, R0 wait for sometimes. Then R1 read 1 time and R0 read 2 times. Finally Shared variable value will be remain same.

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

4) When Number of “READER=0’ is taken as input



```
rg_rajat@LAPTOP-GMF6NFUE: ~
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
0
---------------------------------------------
Enter the no. of Writer:
2
W0
W1
---------------------------------------------
Sorry: You have not taken any READER
No READER thread will be creaded
---------------------------------------------
--------------------------------------------------
W0 Wait for Random time between 0ns and 10ns = 3
Enter the number of time W0 want to write:
2
Now W0 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
3
Enter the 2th INTEGER value to write:
6
UPDATED value of Shared variable = 19
--------------------------------------------------
--------------------------------------------------
W1 Wait for Random time between 0ns and 10ns = 6
Enter the number of time W1 want to write:
1
Now W1 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
5
UPDATED value of Shared variable = 24
--------------------------------------------------
-------------After joining the thread---------
Final value of share variable = 24
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
Enter the 'INTEGER' Initial value of share variable:
_
```

In This Case:

i. Only WRITER thread is created

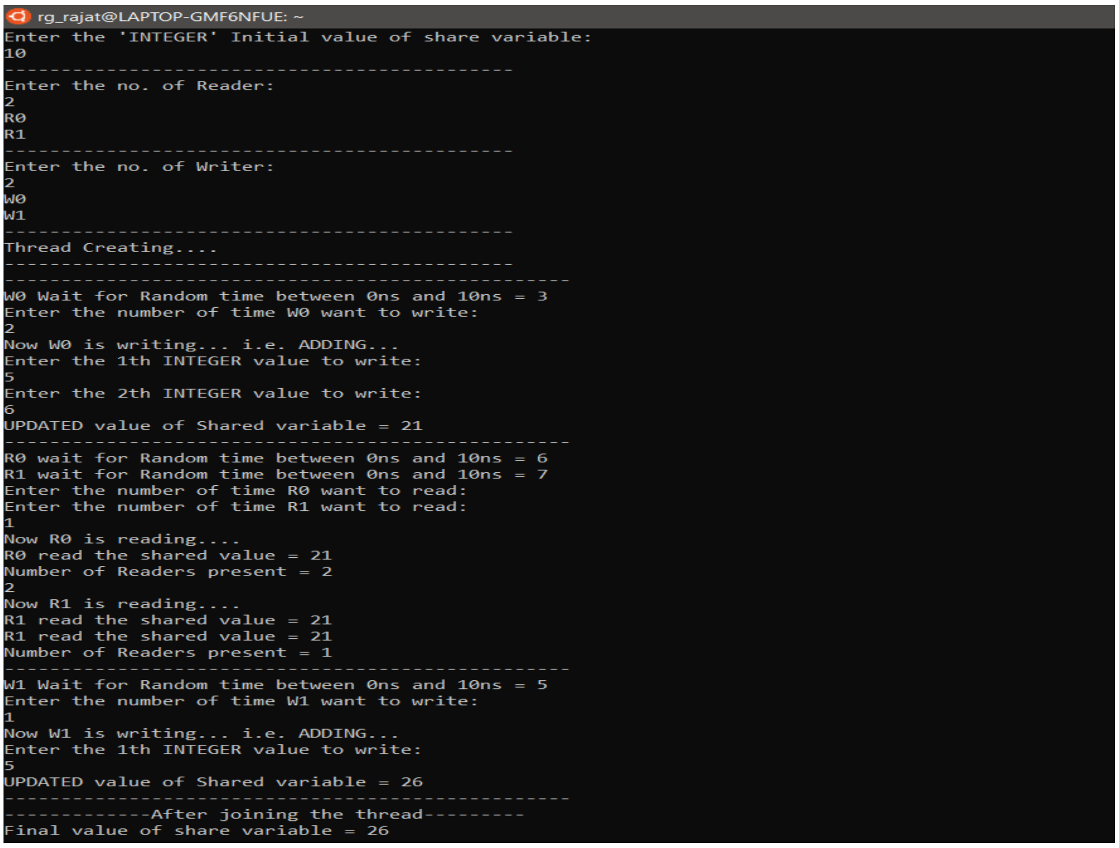
ii. No READER thread is created

iii. No reading of the shared variable

iv. Only WRITER is updating the shared variable. From the same example given above W1, W0 will make change on the shared variable. W1 writes 2 times and W0 writes 1 time. Finally shared variable value will be changes/updated.

5) User taken same number of READER and WRITER i.e. READER = WRITER

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085


rg_rajat@LAPTOP-GMF6NFUE: ~

```
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
2
R0
R1
---------------------------------------------
Enter the no. of Writer:
2
W0
W1
---------------------------------------------
Thread Creating....
---------------------------------------------
--------------------------------------------------
W0 Wait for Random time between 0ns and 10ns = 3
Enter the number of time W0 want to write:
2
Now W0 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
5
Enter the 2th INTEGER value to write:
6
UPDATED value of Shared variable = 21
--------------------------------------------------
R0 wait for Random time between 0ns and 10ns = 6
R1 wait for Random time between 0ns and 10ns = 7
Enter the number of time R0 want to read:
Enter the number of time R1 want to read:
1
Now R0 is reading....
R0 read the shared value = 21
Number of Readers present = 2
2
Now R1 is reading....
R1 read the shared value = 21
R1 read the shared value = 21
Number of Readers present = 1
--------------------------------------------------
W1 Wait for Random time between 0ns and 10ns = 5
Enter the number of time W1 want to write:
1
Now W1 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
5
UPDATED value of Shared variable = 26
--------------------------------------------------
------------After joining the thread---------
Final value of share variable = 26
```

In This Case:

i. Both READER and WRITER threads created

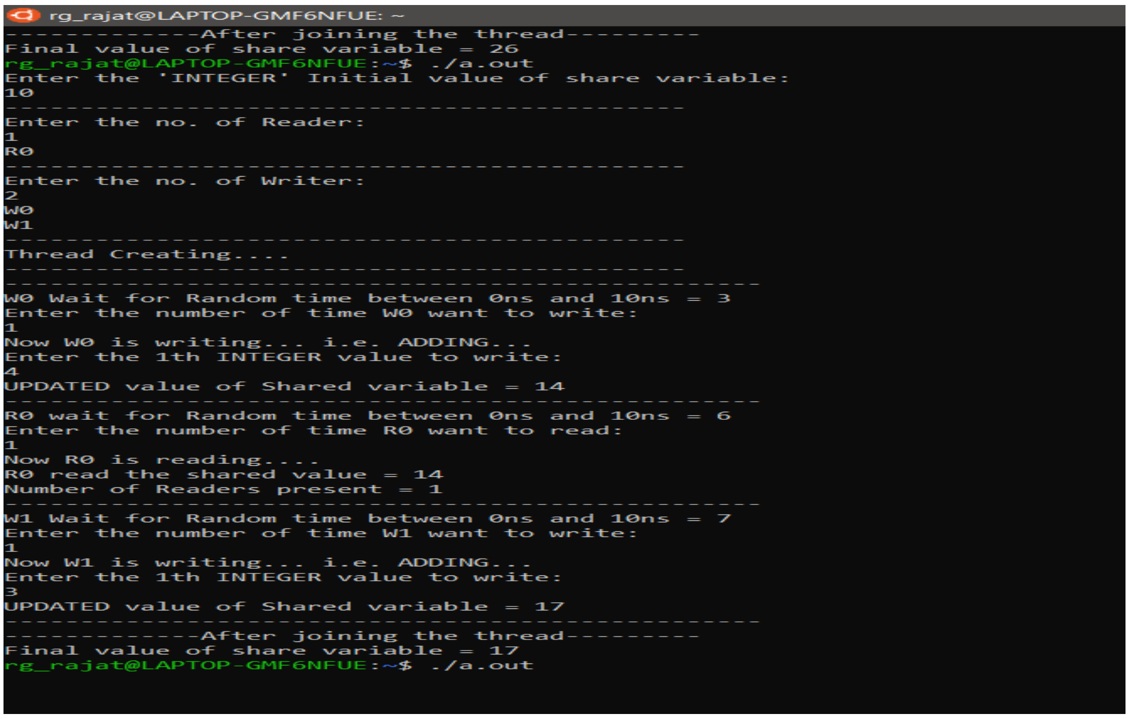
ii. Any WRITER can update the Shared variable any number of times

iii. Any READER can read that Value any number of times

iv. In the screen slot, W0 wait for some random times. Then he has the choice to update the variable any number of times with any number. Here W0 update for 1 time. He added with value 5 and result in 26

6) User taken a greater number of WRITER as compared with READER i.e. READER < WRITER

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085



```
rg_rajat@LAPTOP-GMF6NFUE: ~
------------After joining the thread---------
Final value of share variable = 26
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
1
R0
---------------------------------------------
Enter the no. of Writer:
2
W0
W1
---------------------------------------------
Thread Creating....
---------------------------------------------
--------------------------------------------------
W0 Wait for Random time between 0ns and 10ns = 3
Enter the number of time W0 want to write:
1
Now W0 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
4
UPDATED value of Shared variable = 14
--------------------------------------------------
R0 wait for Random time between 0ns and 10ns = 6
Enter the number of time R0 want to read:
1
Now R0 is reading....
R0 read the shared value = 14
Number of Readers present = 1
--------------------------------------------------
W1 Wait for Random time between 0ns and 10ns = 7
Enter the number of time W1 want to write:
1
Now W1 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
3
UPDATED value of Shared variable = 17
--------------------------------------------------
------------After joining the thread---------
Final value of share variable = 17
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
```

In This Case:

i. Both READER and WRITER threads created

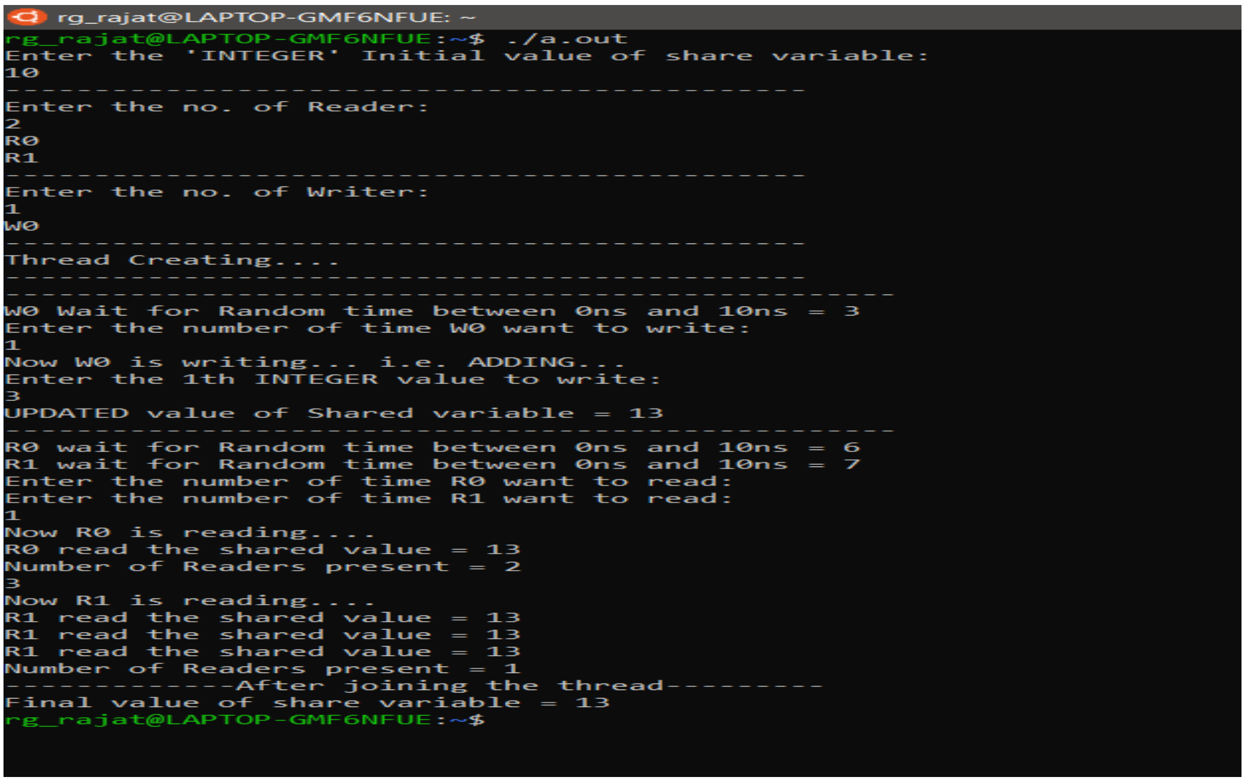
ii. Any WRITER can update the Shared variable any number of times

iii. Any READER can read that Value any number of times

iv. In this screen slot, W0 wait for some time and add some value with shared variable. Then R0 given the choice to read any number of times. Then again, the turn of next writer i.e. W1 came.

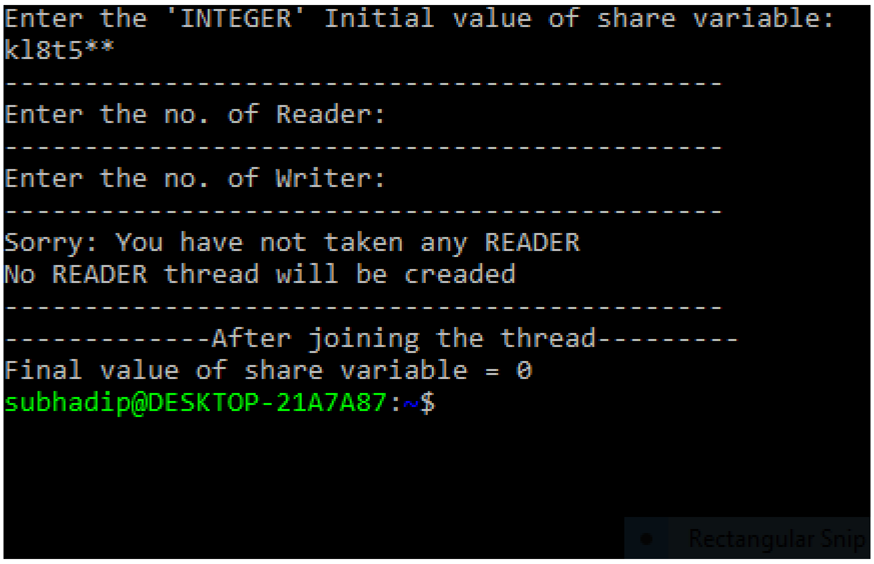
Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

7) User taken a greater number of READER as compared with WRITER i.e. READER > WRITER



```
rg_rajat@LAPTOP-GMF6NFUE: ~
rg_rajat@LAPTOP-GMF6NFUE:~$ ./a.out
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
2
R0
R1
---------------------------------------------
Enter the no. of Writer:
1
W0
---------------------------------------------
Thread Creating....
---------------------------------------------
--------------------------------------------------
W0 Wait for Random time between 0ns and 10ns = 3
Enter the number of time W0 want to write:
1
Now W0 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
3
UPDATED value of Shared variable = 13
--------------------------------------------------
R0 wait for Random time between 0ns and 10ns = 6
R1 wait for Random time between 0ns and 10ns = 7
Enter the number of time R0 want to read:
Enter the number of time R1 want to read:
1
Now R0 is reading....
R0 read the shared value = 13
Number of Readers present = 2
3
Now R1 is reading....
R1 read the shared value = 13
R1 read the shared value = 13
R1 read the shared value = 13
Number of Readers present = 1
-------------After joining the thread---------
Final value of share variable = 13
rg_rajat@LAPTOP-GMF6NFUE:~$
```

8) Shared Variable value is taken non-INTEGER



```
Enter the 'INTEGER' Initial value of share variable:
kl8t5**
---------------------------------------------
Enter the no. of Reader:
---------------------------------------------
Enter the no. of Writer:
---------------------------------------------
Sorry: You have not taken any READER
No READER thread will be creaded
---------------------------------------------
-------------After joining the thread---------
Final value of share variable = 0
subhadip@DESKTOP-21A7A87:~$
```

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085

In This Case:

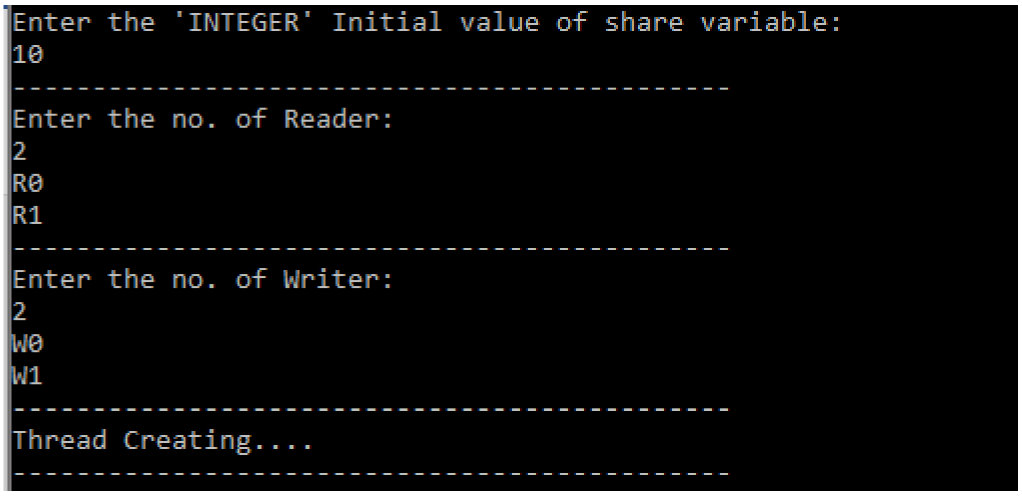
i. User unable to take any input

ii. No READER and WRITER thread were created

iii. We will not get output as we expected

iv. But get the Shared variable value as 0

Program Working with a Proper example



```
Enter the 'INTEGER' Initial value of share variable:
10
---------------------------------------------
Enter the no. of Reader:
2
R0
R1
---------------------------------------------
Enter the no. of Writer:
2
W0
W1
---------------------------------------------
Thread Creating....
---------------------------------------------
```

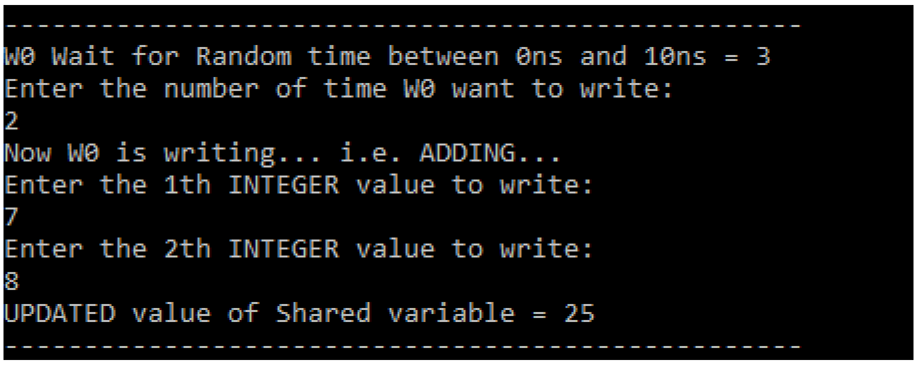
ii. User taken the initial value of share variable(S) as 10

iii. User taken the no. of reader(R) is 2 as input i.e. R0 & R1

iv. User taken the no. of writer(W) is 2 as input i.e. W0 & W1

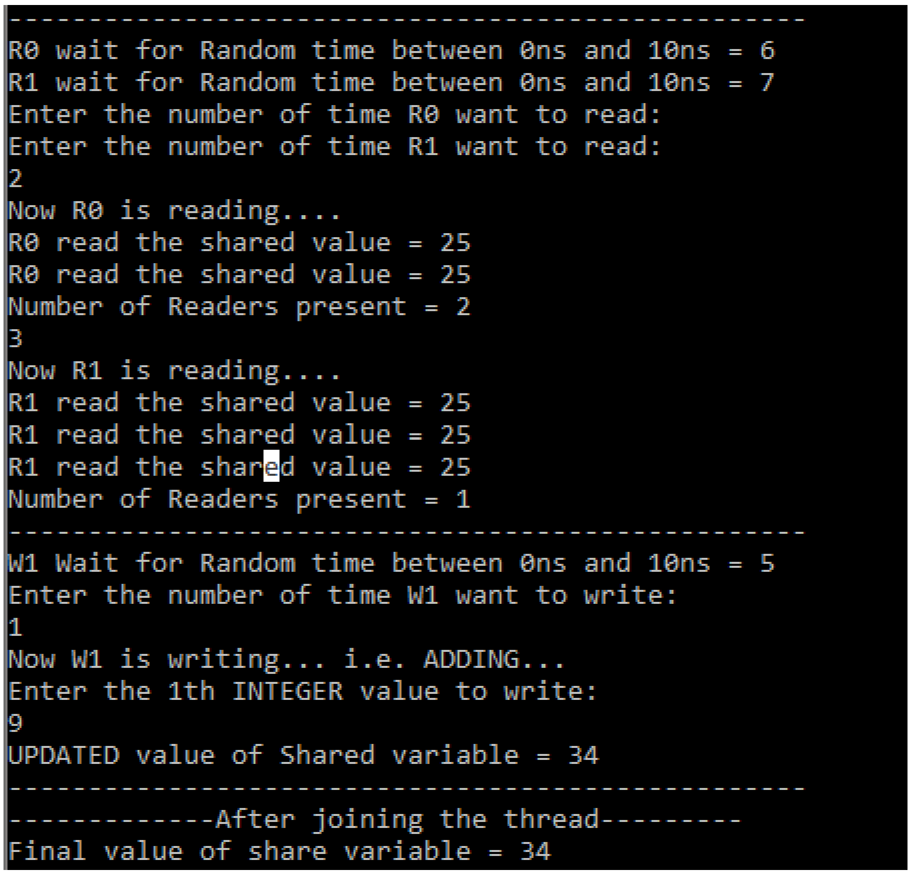
v. Now READER and WRITER threads were created successfully

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085



```
--------------------------------------------------
W0 Wait for Random time between 0ns and 10ns = 3
Enter the number of time W0 want to write:
2
Now W0 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
7
Enter the 2th INTEGER value to write:
8
UPDATED value of Shared variable = 25
--------------------------------------------------
```

vii. W0 wait for some random time. Then we choose that he will update S two time. First time we add 7 to S. S = 10 + 7 = 17 then second time we add 8 to S. Now S = 17 + 8 = 25. So, New updated value is 25.



```
--------------------------------------------------
R0 wait for Random time between 0ns and 10ns = 6
R1 wait for Random time between 0ns and 10ns = 7
Enter the number of time R0 want to read:
Enter the number of time R1 want to read:
2
Now R0 is reading....
R0 read the shared value = 25
R0 read the shared value = 25
Number of Readers present = 2
3
Now R1 is reading....
R1 read the shared value = 25
R1 read the shared value = 25
R1 read the shared value = 25
Number of Readers present = 1
--------------------------------------------------
W1 Wait for Random time between 0ns and 10ns = 5
Enter the number of time W1 want to write:
1
Now W1 is writing... i.e. ADDING...
Enter the 1th INTEGER value to write:
9
UPDATED value of Shared variable = 34
--------------------------------------------------
-------------After joining the thread---------
Final value of share variable = 34
```

ix. Now R0 & R1 wait for some random times. Then R0 wants that we want to read the shared variable 2 time. Then R1 read for 3 times. At this instant, 1 reader is present. Then W1 wait for sometimes and he want to add 9 to the shared variable for 1 time. S = 25 + 9 = 34.

x. Finally, after joining the thread, we got the shared variable value as 34.

Rajat Garg
11807535
rajat.11807535@lpu.in
Github: rgrajat085