

MIT WORLD PEACE UNIVERSITY

Operating Systems
Second Year B. Tech, Semester 3

PROCESS SYNCHRONIZATION - SIMULATION OF
READER-WRITER PROBLEM IN C

ASSIGNMENT 2
PRACTICAL REPORT

Prepared By
Krishnaraj Thadesar
Cyber Security and Forensics
Batch A2, PA 20
November 3, 2022

1 Code

```
1 // Reader writer problem.
2
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <unistd.h>
6 #include <pthread.h>
7 #include <semaphore.h>
8
9 sem_t sem_wrt;
10 sem_t sem_mutex;
11 int shared_variable = 0;
12 int number_of_readers;
13
14 void *reader()
15 {
16     sem_wait(&sem_mutex);
17     printf("\nRead: %d\n", shared_variable);
18     printf("Reader finished its CS so releasing mutex\n");
19     sem_post(&sem_mutex);
20 }
21
22 void *writer()
23 {
24     sem_wait(&sem_wrt);
25     sem_wait(&sem_mutex);
26     printf("Blocking sem wait and mutex variable so no other writer can write rn.
27 \n");
28     shared_variable++;
29     printf("Wrote to the shared variable %d\n", shared_variable);
30     sem_post(&sem_wrt);
31     sem_post(&sem_mutex);
32 }
33
34 int main()
35 {
36     pthread_t t1, t2;
37     sem_init(&sem_mutex, 0, 1);
38     sem_init(&sem_wrt, 0, 1);
39
40     printf("Enter how many readers and Writers you want (Same number of both are
41 taken by default): ");
42     scanf("%d", &number_of_readers);
43
44     for (int i = 0; i < number_of_readers; i++)
45     {
46         pthread_create(&t2, NULL, writer, NULL);
47         pthread_create(&t1, NULL, reader, NULL);
48     }
49
50     pthread_join(t1, NULL);
51     pthread_join(t2, NULL);
52     sem_destroy(&sem_mutex);
53 }
```

Listing 1: Assignment 5.Cpp

2 Input and Output

```
1 Enter how many readers and Writers you want (Same number of both are taken by
  default): 8
2 Blocking sem wait and mutex variable so no other writer can write rn.
3 Wrote to the shared variable 1
4
5 Read: 1
6 Reader finished its CS so releasing mutex
7 Blocking sem wait and mutex variable so no other writer can write rn.
8 Wrote to the shared variable 2
9
10 Read: 2
11 Reader finished its CS so releasing mutex
12 Blocking sem wait and mutex variable so no other writer can write rn.
13 Wrote to the shared variable 3
14
15 Read: 3
16 Reader finished its CS so releasing mutex
17 Blocking sem wait and mutex variable so no other writer can write rn.
18 Wrote to the shared variable 4
19
20 Read: 4
21 Reader finished its CS so releasing mutex
22 Blocking sem wait and mutex variable so no other writer can write rn.
23 Wrote to the shared variable 5
24
25 Read: 5
26 Reader finished its CS so releasing mutex
27 Blocking sem wait and mutex variable so no other writer can write rn.
28 Wrote to the shared variable 6
29
30 Read: 6
31 Reader finished its CS so releasing mutex
32
33 Read: 6
34 Reader finished its CS so releasing mutex
35
36 Read: 6
37 Reader finished its CS so releasing mutex
38 Blocking sem wait and mutex variable so no other writer can write rn.
39 Wrote to the shared variable 7
40 Blocking sem wait and mutex variable so no other writer can write rn.
41 Wrote to the shared variable 8
```

Listing 2: Input and Output.Cpp