

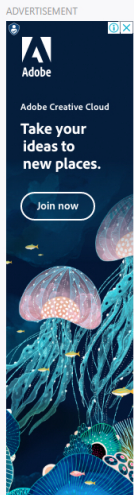
INTRODUCTION TO OS ▾

PROCESS & MULTITHREADING ▾

[Operating System Processes](#)
[Process Scheduling](#)
[CPU Scheduling](#)
[First Come First Serve](#)
[Shortest Job First](#)
[Priority Scheduling](#)
[Round Robin Scheduling](#)
[Multilevel Queue Scheduling](#)
[Comparison of Scheduling Algorithms](#)
[Introduction to Threads](#)
[Process Synchronization](#)
[Classical Synchronization Problems](#)
[Bounded Buffer Problem](#)
[Dining Philosophers Problem](#)
[Readers Writer Problem](#)
[Semaphores in OS](#)
[Deadlocks](#)
[Classical Problems of Synchronization](#)
[Deadlock Prevention in OS](#)
[Deadlock Avoidance in OS](#)
[Deadlock Detection and Recovery](#)

CPU SCHEDULING ▾

MEMORY MANAGEMENT ▾



What is Readers Writer Problem?

ADVERTISEMENT



Readers writer problem is another example of a classic synchronization problem. There are many variants of this problem, one of which is examined below.

The Problem Statement

There is a shared resource which should be accessed by multiple processes. There are two types of processes in this context. They are **reader** and **writer**. Any number of **readers** can read from the shared resource simultaneously, but only one **writer** can write to the shared resource. When a **writer** is writing data to the resource, no other process can access the resource. A **writer** cannot write to the resource if there are non zero number of readers accessing the resource at that time.

The Solution

From the above problem statement, it is evident that readers have higher priority than writer. If a writer wants to write to the resource, it must wait until there are no readers currently accessing that resource.

Here, we use one **mutex** `m` and a **semaphore** `w`. An integer variable `read_count` is used to maintain the number of readers currently accessing the resource. The variable `read_count` is initialized to 0. A value of 1 is given initially to `m` and `w`.

Instead of having the process to acquire lock on the shared resource, we use the mutex `m` to make the process to acquire and release lock whenever it is updating the `read_count` variable.

The code for the **writer** process looks like this:

```
while(TRUE)
{
    wait(w);

    /* perform the write operation */

    signal(w);
}
```

And, the code for the **reader** process looks like this:

```
//acquire lock
wait(m);
read_count++;
if(read_count == 1)
    wait(w);

//release lock
signal(m);

/* perform the reading operation */

// acquire lock
wait(m);
read_count--;
if(read_count == 0)
    signal(w);

// release lock
signal(m);
}
```

Here is the Code uncoded(explained)

- As seen above in the code for the writer, the writer just waits on the `w` semaphore until it gets a chance to write to the resource.
- After performing the write operation, it increments `w` so that the next writer can access the resource.
- On the other hand, in the code for the reader, the lock is acquired whenever the `read_count` is updated by a process.
- When a reader wants to access the resource, first it increments the `read_count` value, then accesses the resource and then decrements the `read_count` value.
- The semaphore `w` is used by the first reader which enters the critical section and the last reader which exits the critical section.
- The reason for this is, when the first readers enters the critical section, the writer is blocked from the resource. Only new readers can access the resource now.

OS MCQ Tests

Prepare for operating system related Interview questions.

[Explore](#)

GATE MCQ Tests

Prepare for GATE 2022

[Explore](#)



ADVERTISEMENT



- Similarly, when the last reader exits the critical section, it signals the writer using the **w** semaphore because there are zero readers now and a writer can have the chance to access the resource.

[< Prev](#)[Next >](#)

studytonight.com



About Us
Testimonials
Authors
Collaborate
Privacy Policy
Terms
Contact Us
Suggest
© 2021 Studytonight Technologies Pvt. Ltd.

[Learn Coding \(for beginners\)](#)[Tutorial Library](#)[Interview Tests](#)[# Curious](#)[Practice Coding](#)[Educators Program](#)

Coding Courses

[Learn HTML](#)
[Learn CSS](#)

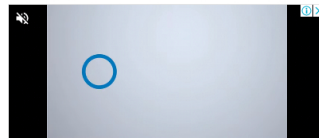
Resources

[C Language](#)
[C++/STL](#)
[Java](#)
[DBMS](#)
[Python](#)
[PHP](#)
[Android](#)
[Game Development](#)
[Data Structure & Alog.](#)
[Operating System](#)
[Computer Network](#)
[Computer Architecture](#)
[More...](#)

Interview Tests

[Java Interview Tests](#)
[Python Interview Tests](#)
[DBMS Interview Tests](#)
[Linux Interview Tests](#)
[Aptitude Tests](#)
[GATE 2022 Tests](#)
[More...](#)

ADVERTISEMENT



[Bring your sugar under control](#)

[Dr. Mohan's Diabetes Specialities Centre](#)



Now Beard It Your Way

[Philips OneBlade](#)

[Buy Now](#)

