

[Practice](#)[GATE CS](#)[Placements](#)[Videos](#)[Contribute](#)[Login/Register](#)**Quick Links  
for Operating  
Systems**[Recent Articles](#)[MCQ / Quizzes](#)[Practice  
Problems](#)[Last Minute  
Notes \(LMNs\)](#)**Basics**[What happens  
when we turn on  
computer?](#)[Explore More...](#)**Processes &  
Threads**[Process -  
Introduction](#)[Thread](#)[User Level thread  
vs. Kernel Level  
thread](#)[Zombie  
Processes and  
their Prevention](#)[Maximum  
number of  
Zombie process a  
system can  
handle](#)[Maximum  
number of  
threads that can  
be created within  
a process in C](#)

## Northwestern's Online Master's in Information Design and Strategy.

Multi threading  
models[Explore More...](#)**Process  
Synchronization**Introduction &  
Critical SectionInter Process  
CommunicationMutex vs  
Semaphore &  
MonitorsPeterson's  
Algorithm for  
Mutual Exclusion  
| Set 1 & Set 2Readers-Writers  
ProblemPriority Inversion  
: What the heck !Banker's  
Algorithm &  
ProgramPriority Inversion  
vs. Priority  
Inheritance[Explore More...](#)**CPU Scheduling**Process  
Management -  
IntroductionCPU Scheduling &  
Process  
SchedulerFCFS Scheduling |  
Set 1 & Set 2

SJF scheduling

Round Robin  
schedulingPriority  
Scheduling

Northwestern's Online Master's in Information Design and Strategy.

<b>Deadlocks</b>
Introduction
Detection And Recovery
Prevention And Avoidance
Explore More...
<b>Memory Management</b>
Partition Allocation Method
Virtual Memory
Paging
Segmentation
Page Replacement Algorithms
Static and Dynamic Libraries
Working with Shared Libraries   Set 1 & Set 2
Explore More...
<b>File &amp; Disk Management</b>
File System
File Allocation Methods
Disk Scheduling Algorithms
Explore More...
<b>Linux</b>
Linux File Hierarchy Structure



[Some useful  
Linux Hacks](#)[Explore More...](#)

## Priority Inversion : What the heck !


Let us first put 'priority inversion' in the context of the Big Picture i.e. where does this come from. **2.3**

In Operating System, one of the important concepts is Task Scheduling. There are several Scheduling methods such as First Come First Serve, Round Robin, Priority based scheduling etc. Each scheduling method has its pros and cons. As you might have guessed, Priority Inversion comes under Priority based Scheduling. Basically, it's a problem which arises sometimes when Priority based scheduling is used by OS. In Priority based scheduling, different tasks are given different priority so that higher priority tasks can intervene lower priority tasks if possible.

So, in a priority based scheduling, if lower priority task (L) is running and if a higher priority task (H) also needs to run, the lower priority task (L) would be preempted by higher priority task (H). Now, suppose both lower and higher priority tasks need to share a common resource (say access to the same file or device) to achieve their respective work. In this case, since there's resource sharing and task synchronization is needed, several methods/techniques can be used for handling such scenarios. For sake of our topic on Priority Inversion, let us mention a synchronization method say mutex. Just to recap on mutex, a task acquires mutex before entering critical section (CS) and releases mutex after exiting critical section (CS). While running in CS, a task access this common resource. More details on this can be referred [here](#). Now, say both L and H shares a common Critical Section (CS) i.e. same mutex is needed for this CS.

Coming to our discussion of priority inversion, let us examine some scenarios.

- 1) L is running but not in CS ; H needs to run; H preempts L ; H starts running ; H relinquishes or releases control ; L resumes and starts running
- 2) L is running in CS ; H needs to run but not in CS; H preempts L ; H starts running ; H relinquishes control ; L resumes and starts running.
- 3) L is running in CS ; H also needs to run in CS ; H waits for L to come out of CS ; L comes out of CS ; H enters CS and starts running

Please note that the above scenarios don't show the problem of any Priority Inversion (not even scenario 3). Basically, so long as lower priority task isn't running in shared CS, higher priority task can preempt it. But if L is running in shared CS and H also needs to run in CS, H waits until L comes out  The idea is that CS should be small enough so that it doesn't result in H waiting for long time while L was in CS. That's why writing CS code requires careful consideration. In any of the above scenarios, pri-

**Northwestern's Online Master's in Information Design and Strategy.**

H. In our example, M doesn't share the same Critical Section (CS). In this case, the following sequence of task running would result in 'Priority Inversion' problem.

4) L is running in CS ; H also needs to run in CS ; H waits for L to come out of CS ; M interrupts L and starts running ; M runs till completion and relinquishes control ; L resumes and starts running till the end of CS ; H enters CS and starts running.

Note that neither L nor H share CS with M.

Here, we can see that running of M has delayed the running of both L and H. Precisely speaking, H is of higher priority and doesn't share CS with M; but H had to wait for M. This is where Priority based scheduling didn't work as expected because priorities of M and H got inverted in spite of not sharing any CS. This problem is called Priority Inversion. This is what the heck was Priority Inversion ! In a system with priority based scheduling, higher priority tasks can face this problem and it can result in unexpected behavior/result. In general purpose OS, it can result in slower performance. In RTOS, it can result in more severe outcomes. The most famous 'Priority Inversion' problem was what happened at Mars Pathfinder.

If we have a problem, there has to be solution for this. For Priority Inversion as well, there're different solutions such as Priority Inheritance etc. This is going to be our next article 😊

But for the inpatients, **this** can be referred for time being.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

**20% Off Ticwatch S & E**

Powered by Android Wear™. Compatible with Android™ and iOS. Starts from \$159.99

**20% Off Ticwatch S & E**

Powered by Android Wear™. Compatible with Android™ and iOS. Starts from \$159.99

**GATE CS Corner    Company Wise Coding Practice**[Articles](#)[Operating Systems](#)[Process Synchronization](#)[Login to Improve this Article](#)

Northwestern  
INFORMATION DESIGN  
AND STRATEGY

[Find out more!](#) x

Northwestern's Online Master's in Information Design and Strategy.

What's difference between Priority Inversion and Priority Inheritance ?

Operating System | Process Management | Deadlock Introduction

Operating System | Banker's Algorithm

Program for Banker's Algorithm | Set 1 (Safety Algorithm)

Operating System | Critical Section

Elo Rating Algorithm

find command in Linux with examples

Don't Forget The Edge Cases !

Guide for Non-CS students to get placed in Software companies

diff command in Linux with examples

(Login to Rate)

**2.3**

Average Difficulty : **2.3/5.0**  
Based on **28** vote(s)



Add to TODO List



Mark as DONE

Basic

Easy

Medium

Hard

Expert

Writing code in comment? Please use [ide.geeksforgeeks.org](https://ide.geeksforgeeks.org), generate link and share the link here.

Load Comments

Share this post!

@geeksforgeeks, Some rights reserved

Contact Us!

About Us!

Careers!

Privacy Policy

