

WhatsApp Process Synchronization GAT

youtube.com/watch?v=SpyULhXEF0I&list=PLG9aCp4uE-s3kreqEhzbzOBQDg5Ha0U388&index=4

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... cist Slack | design-autu... aapatre/Automatic... Reading list

YouTube Search

GATE 1987

A critical region is

- A. One which is enclosed by a pair of P and V operations on semaphores. *wait()* *Signal*
- ~~B. A program segment that has not been proved bug-free.~~
- ~~C. A program segment that often causes unexpected system crashes.~~
- ~~D. A program segment where shared resources are accessed.~~

unacademy

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_1987_2000
Process Synchronization GATE 1987-2000 | L 4 | Operating System GATE PYQs | GATE 2022
1,060 views · Streamed live on Aug 13, 2021 85 0 SHARE SAVE ...

Top chat replay

- Soniya Dhimani d
- RAJA RAJPUT shared variables are used
- Shivanshi Shrivastava d
- nirlipta baral d
- Suman Singh d
- Vinay Pant d
- Shweta Pandey gd morning sir
- RAJA RAJPUT need not be bug free
- xinsheng A can't be the solution because it is the solution to CS, and not CS itself.
- NISHU KUMAR D
- RAJA RAJPUT unexpectation
- Shweta Pandey sir please create the playlist of this series' 🙏🙏
- Karan saroj good morning sir ❤️❤️❤️❤️❤️
- NISHU KUMAR Hi Sir good morning

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear ENG 12:47 AM 10/26/2021

WhatsApp Process Synchronization GAT

youtube.com/watch?v=SpyULhXEF0I&list=PLG9aCp4uE-s3kreqEhbzOBQDg5Ha0U388&index=4

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Stack | design-autu... aapatre/Automatic... Reading list

YouTube Search

GATE 1990

Match the pairs in the following questions:

(a) Critical region	(p) Hoare's monitor
(b) Wait/Signal	(q) Mutual exclusion
(c) Working Set	(r) Principle of locality
(d) Deadlock	(s) Circular Wait

unacademy

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_1987_2000
Process Synchronization GATE 1987-2000 | L 4 | Operating System GATE PYQs | GATE 2022
1,060 views · Streamed live on Aug 13, 2021

85 0 SHARE SAVE

Top chat replay

- S Shweta Pandey gd morning sir
- R RAJA RAJPUT need not be bug free
- xinsheng A can't be the solution because it is the solution to CS, and not CS itself.
- NISHU KUMAR D
- R RAJA RAJPUT unexpected
- S Shweta Pandey sir please create the playlist of this series' 🙏🙏
- Karan saroj good morning sir ❤️❤️❤️❤️❤️❤️❤️
- NISHU KUMAR Hi Sir good morning
- NISHU KUMAR D is perfectly correct
- R RAJA RAJPUT that is for silutuons using semaphore
- NISHU KUMAR Semaphores]
- R RAJA RAJPUT without semaphore using turn, flag also we can solve
- NISHU KUMAR a - q
- R RAJA RAJPUT 2) a-q, d-s, c-r, b-p

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear ENG 12:48 AM 10/26/2021

WhatsApp Process Synchronization GAT

youtube.com/watch?v=SpyULhXEF0I&list=PLG9aCp4uE-s3kreqEhzbzOBQDg5Ha0U388&index=4

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Slack | design-autu... aapatre/Automatic-... Reading list

YouTube Search

GATE 1996

A critical section is a program segment

- ☒ A which should run in a certain amount of time
- ☒ B which avoids deadlocks
- ☒ C where shared resources are accessed
- ☒ D which must be enclosed by a pair of semaphores P and V

unacademy

12:29 / 40:11

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_1987_2000

Process Synchronization GATE 1987-2000 | L 4 | Operating System GATE PYQs | GATE 2022

1,060 views · Streamed live on Aug 13, 2021

85 0 SHARE SAVE

Top chat replay

- this series???
- xinsheng Are Wait and Signal control signal?
- NISHU KUMAR b - p
- Heba Shakeel Good morning sir
- NISHU KUMAR yes
- Heba Shakeel C
- RAJA RAJPUT 3) CCC
- Rajesh Sah C
- Musafir of Life Style yes sir 😊😊
- NISHU KUMAR C
- Hiren Thakkar C
- Soniya Dhimian c
- RAJA RAJPUT That is bounded wait
- NISHU KUMAR where shared resources are accessed
- RAJA RAJPUT yes sir
- NISHU KUMAR yes sir

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear 12:49 AM 10/26/2021

WhatsApp x Process Synchronization GAT x +

youtube.com/watch?v=SpyULhXEF0I&list=PLG9aCp4uE-s3kreqEhbzOBQDg5Ha0U388&index=4


Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Stack | design-autu... aapatre/Automatic... Reading list

YouTube Search

GATE 1996

A solution to the Dining Philosophers Problem is deadlock is to

- A. ensure that all philosophers pick up the right fork
- B. ensure that all philosophers pick up the left fork
- ☒ C. ensure that one particular philosopher picks up the left fork before the right fork, and that all other philosophers pick up the right fork
- D. None of the above



unacademy

15:40 / 40:11

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_1987_2000

Process Synchronization GATE 1987-2000 | L 4 | Operating System GATE PYQs | GATE 2022

1,060 views · Streamed live on Aug 13, 2021

85 0 SHARE SAVE ...

Top chat replay

- NISHU KUMAR aa jaunga solve karne ke liye
- Heba Shakeel C
- RAJA RAJPUT 4) CCC
- Shweta Pandey opt c
- NISHU KUMAR C
- Rajesh Sah C
- Soniya Dhiman c
- Hiren Thakkar C
- RAJA RAJPUT no that will be deadlock
- RAJA RAJPUT AB are deadlocked
- Heba Shakeel Again deadlock
- NISHU KUMAR khana khate rahite he
- Heba Shakeel all left one right
- Heba Shakeel or all right one left
- RAJA RAJPUT reverse me
- RAJA RAJPUT right will be available so no issue

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear ENG 12:51 AM 10/26/2021

GATE 1997

Each Process P_i , $i = 1 \dots 9$ is coded as follows

```
repeat  $P_1, P_2$   
  P(mutex)  
  {Critical Section}  
  V(mutex)  
forever
```

The code for P_{10} is identical except it uses V(mutex) before P(mutex). What is the largest number of processes that can be inside the critical section?

- A. 1
- B. 2
- C. 3
- D. None

GATE 1997

Each Process $P_i, i = 1 \dots 9$ is coded as follows

```
repeat  $P_1, P_2, \dots, P_9$   
  P(mutex)  
  {Critical section}  
  V(mutex)  
forever
```

The code for P_{10} is identical except it uses $V(mutex)$ in place of $P(mutex)$. What is the largest number of processes that can be inside the critical section at any time?

- A. 1
- B. 2
- C. 3
- ✓ D. None

Ans = 10

mutex = 1

unacademy

GATE 1998

When the result of a computation depends on the order of the processes involved, there is said to be

A. cycle stealing

✓ B. race condition ✓

C. a time lock

▶ D. a deadlock



GATE 2000

Let $m[0] \dots m[4]$ be mutexes (binary semaphores) and $P[0] \dots P[4]$ be processes. Suppose each process $P[i]$ executes the following:

```
wait (m[i]); wait (m[i+1]);
.....
release (m[i]); release (m[i+1]);
```

This could cause

- ☒ A. Thrashing
- ☒ B. Deadlock
- ☐ C. Starvation, but not deadlock
- ☐ D. None of the above

$m[0] \rightarrow P[0]$
 $m[1] \rightarrow P[1]$
 $m[2] \rightarrow P[2]$
 $m[3] \rightarrow P[3]$
 $m[4] \rightarrow P[4]$

$P[1]$	$P[2]$
wait $m[1]$	wait $m[2]$
wait $m[2]$	wait $m[3]$
$P[4]$	
wait $m[4]$	
wait $m[1]$	

unacademy

GATE 2006

Consider the solution to the bounded buffer producer problem by using general semaphores S , F , and E . The semaphore S is the mutex semaphore initialized to 1. The semaphore F corresponds to the number of free spaces in the buffer and is initialized to N . The semaphore E corresponds to the number of elements in the buffer and is initialized to 0.

Producer Process	Consumer Process
Produce an item; $\text{Wait}(E);$ ✓	$\text{Wait}(E);$ ✓
$\text{Wait}(F);$ ✓	$\text{Wait}(S);$ ✓
$\text{Wait}(S);$ ✓	Remove an item from the buffer; ✓
Append the item to the buffer; $\text{Signal}(S);$ ✓	
$\text{Signal}(S);$ ✓	
$\text{Signal}(E);$ ✓	Consume

Free = N
 Semaphore $S = 1$
 $E = 0$

all spaces occupied
 $S = \cancel{N} 0$
 $F = 0$
 $E = \cancel{N} N - 1$

Which of the following interchange operations may be performed?

- I. Interchanging $\text{Wait}(F)$ and $\text{Wait}(S)$ in the Producer process.
 - II. Interchanging $\text{Signal}(S)$ and $\text{Signal}(F)$ in the Consumer process.
- A. I only
 B. II only
 C. Neither I nor II
 D. Both I and II

GATE 2006

Consider the solution to the buffer producer/consumer problem by using general semaphores S , F , and E . The mutual exclusion semaphore initialized to 1. The semaphore F corresponds to free slots in the buffer and is initialized to N . The semaphore E corresponds to elements in the buffer and is initialized to 0.

Producer Process	Consumer Process
Produce an item;	Wait(E);
Wait(F);	Wait(S);
Wait(S);	Remove item from buffer;
Append the item to the buffer;	Signal(S);
Signal(S);	Signal(E);

Free = N
~~any~~ here $S = 1$
 $E = \text{empty}$
 Elements = 0

all spaces occupied

$S = \cancel{X} 0$
 $F = 0$
 $E = \cancel{X} N - 1$

Which of the following may result in a deadlock?

- I. Interchanging the Producer process and the Consumer process
 - II. Interchanging the Consumer process and the Producer process
- A. I only
 B. II only
 C. Neither I nor II
 D. Both I and II

GATE 2007

synchronization construct used by the processes:

<pre>/* P1 */ while (true) { wants1 = true; while (wants2 == true); /* Critical Section */ wants1 = false; } /* Remainder section */</pre>	<pre>/* P2 */ while (true) { wants2 = true; while (wants1 == true); /* Critical Section */ wants2=false; } /* Remainder section */</pre>
--	--

Here, wants1 and wants2 are shared variables, which are initialized to false.

Which one of the following statements is TRUE about the construct?

- A. It does not ensure mutual exclusion.
- B. It does not ensure bounded waiting.
- C. It requires that processes enter the critical section in strict alteration.
- D. It does not prevent deadlocks, but ensures mutual exclusion.

GATE 2007

synchronization construct used by the processes:

<pre>/* P1 */ while (true) { wants1 = true; while (wants2 == true); /* Critical Section */ wants1 = false; } /* Remainder section */</pre>	<pre>/* P2 */ while (true) { wants2 = true; while (wants1 == true); /* Critical Section */ wants2=false; ✓ } /* Remainder section */</pre>
--	--

Here, wants1 and wants2 are shared variables, which are initialized to false

Which one of the following statements is TRUE about the construct?

- A. It does not ensure mutual exclusion.
- B. It does not ensure bounded waiting.
- C. It requires that processes enter the critical section in strict alteration.
- ✓ D. It does not prevent deadlocks, but ensures mutual exclusion.

GATE 2007

Processes P1 and P2 use critical_flag in the following routine to achieve mutual exclusion. Assume that critical_flag is initialized to FALSE in the main program.

```
get_exclusive_access ( )  
{  
    if (critical_flag == FALSE) {  
        critical_flag = TRUE ;  
        critical_region ( ) ;  
        critical_flag = FALSE ;  
    }  
}
```

Critical_flag = false

Consider the following statements.

- i. It is possible for both P1 and P2 to access critical_region concurrently.
- ii. This may lead to a deadlock.

Which of the following holds?

- A. i is false ii is true
- B. Both i and ii are false
- C. i is true ii is false
- D. Both i and ii are true

GATE 2007

Processes P1 and P2 use critical_flag in the following routine to achieve mutual exclusion. Assume that critical_flag is initialized to FALSE in the main program.

```
get_exclusive_access ( )
{
    if (critical_flag == FALSE) {
        critical_flag = TRUE ;
        critical_region ( ) ;
        critical_flag = FALSE;
    }
}
```

Critical_flag = False
True

Consider the following statements.

- ☒ i. It is possible for both P1 and P2 to access critical_region concurrently.
- ☒ ii. This may lead to a deadlock.

Which of the following holds?

- A. i is false ii is true
- B. Both i and ii are false
- ☒ C. i is true ii is false
- D. Both i and ii are true

GATE 2008

The following is a code with two threads, producer and consumer, that can run in parallel. Further, S and Q are binary semaphores quipped with the standard P and V operations.

```
semaphore S = 1, Q = 0;
integer x;

producer:                consumer:
while (true) do          while (true) do
    P(S);                P(Q);
    x = produce ();      consume (x);
    V(Q);                V(S);
done                     done
```

Which of the following is TRUE about the program above?

- A. The process can deadlock
- B. One of the threads can starve
- C. Some of the items produced by the producer may not be consumed before the producer can generate a new value
- D. Values generated and stored in 'x' by the producer may not be consumed before the producer can generate a new value



30:37

30:41 / 1:05:51

Settings Full Screen

GATE 2008

The following is a code with two threads, producer and consumer, running in parallel. Further, S and Q are binary semaphores equipped with the standard P and V operations.

```
semaphore S = 1, Q = 0;
integer x;
```

```
producer:
while (true) do
    P(S);
    x = produce ();
    V(Q);
done
```

```
consumer:
while (true) do
    P(Q);
    consume (x);
    V(S);
done
```

~~2. 10~~

~~10~~ 1
10

Which of the following is TRUE about the program above?

- ☒ A. The process can deadlock
- ☒ B. One of the threads can starve
- ☒ C. Some of the items produced by the producer may be lost
- ☐ D. Values generated and stored in 'x' by the producer can generate a new value



35:32

35:36 / 1:05:51

WhatsApp

Process Synchronization GAT

youtube.com/watch?v=JnCUP4l9H48&list=PLG9aCp4uE-s3klreqEhzbOBQDg5Ha0U38&index=6

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Slack | design-autu... aapatre/Automatic-... Reading list

YouTube

Search

Q

🔊

📺

📑

🔔

M

unacademy

GATE 2009

The enter_CS() and leave_CS() functions to implement critical section of a process using test-and-set instruction as follows:

```
void enter_CS(X)
{
    while(test-and-set(X));
}

void leave_CS(X)
{
    X = 0;
}
```

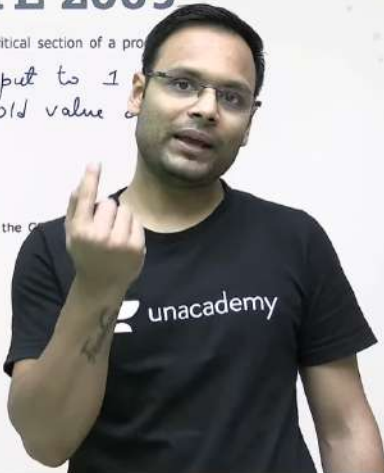
*it sets input to 1
returns old value*

In the above solution, X is a memory location associated with the process. Consider the following statements:

- The above solution to CS problem is deadlock-free
- The solution is starvation free
- The processes enter CS in FIFO order
- More than one process can enter CS at the same time

Which of the above statements are TRUE?

- I only
- I and II
- II and III
- IV only




#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2006_2012


Process Synchronization GATE 2006-2012 | L 6 | Operating System GATE PYQs | GATE 2022


1,081 views · Streamed live on Aug 18, 2021


👍 66 🗨️ 1 ➦ SHARE ➦ SAVE ...

Top chat replay


 durgesh nandan digital ankit sir


 rahul raj yes sir coa

 rahul raj 4.30ok


 rahul raj zehar padhate ho sir aap DANDVAT PARNAM SIR


 Lankesh Ravana sir reply tho kar do plss , big fan hu sir apka ❤️❤️🙏🙏


 rahul raj i am watching these session recorded sir came here to ask timingof coa doubt ...college classes started sir on this time


 Akrisht Kumar d


 Rajesh Sah C


 Rajesh Sah D

 Nikhil Shanbhag D

 Lankesh Ravana does deadlock implies starvation sir ?


 Anand Singh A

 Govind Yadav A

 Lankesh Ravana ok sir understood thank u for answering

HIDE CHAT REPLAY

Type here to search



100%

22°C Mostly clear

ENG

1:14 AM 10/26/2021

WhatsApp

Process Synchronization GAT

youtube.com/watch?v=JnCUP4I9H48&list=PLG9aCp4uE-s3klreqEhbzOBQDg5Ha0U38&index=6

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... cist Slack | design-autu... aapatre/Automatic-... Reading list

YouTube

Search

🔍 🗣️ 📺 📌 🔔 M

unacademy

GATE 2009

The `enter_CS()` and `leave_CS()` functions to implement critical section of a process are realized using test-and-set instruction as follows:

```
void enter_CS(X)
{
    while(!test_and_set(&X));
}

void leave_CS(X)
{
    *X = 0;
}
```


it sets input to 1 and returns old value of input

In the above solution, X is a memory location associated with the CS and is initialized to 0. Now consider the following statements:

- I. The above solution to CS problem is deadlock-free ✓
- II. The solution is starvation free ✗ →
- III. The processes enter CS in FIFO order ✗
- IV. More than one process can enter CS at the same time ✗

Which of the above statements are TRUE?

☒ I only
☒ I and II
☐ I and III
☐ I, II and III



#OperatingSystem_GATEPYQs #VishvadeepGothli #VirtualMemoryGATE_2006_2012

Process Synchronization GATE 2006-2012 | L 6 | Operating System GATE PYQs | GATE 2022

1,081 views · Streamed live on Aug 18, 2021

👍 66 🗨️ 1 ➦ SHARE ➦ SAVE ...

Top chat replay

rahul raj yes sir coa

rahul raj 4.30ok

rahul raj zehar padhate ho sir aap DANDVAT PARNAM SIR

Lankesh Ravana sir reply tho kar do plss . big fan hu sir apka ❤️❤️🙏🙏

rahul raj i am watching these session recorded sir came here to ask timing of coa doubt ...college classes started sir on this time

Akrisht Kumar d

Rajesh Sah C

Rajesh Sah D

Nikhil Shanbhag D

Lankesh Ravana does deadlock implies starvation sir ?

Anand Singh A

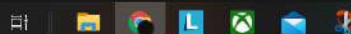
Govind Yadav A

Lankesh Ravana ok sir understood thank u for answering

Rajesh Sah B

HIDE CHAT REPLAY

Type here to search



100%

22°C Mostly clear

ENG

1:16 AM 10/26/2021

WhatsApp

Process Synchronization GAT

youtube.com/watch?v=JnCUP4l9H48&list=PLG9aCp4uE-s3kreqEhbzOBQDg5Ha0U38&index=6

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... cist Slack | design-autu... aapatre/Automatic-... Reading list

YouTube

Search

unacademy

GATE 2010

Consider the methods used by processes P1 and P2 for accessing shared sections whenever needed, as given below. The initial values of shared boolean variables S1 and S2 are randomly assigned.

Method used by P1	Method used by P2
<pre>while (S1==S2); Critical Section S1=S2;</pre>	<pre>while (S1!=S2); Critical Section S2 = not(S1);</pre>

Which one of the following statements describes the properties achieved?

☒ A. Mutual exclusion but not progress
☐ B. Progress but not mutual exclusion
☐ C. Neither mutual exclusion nor progress
☐ D. Both mutual exclusion and progress

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2006_2012

Process Synchronization GATE 2006-2012 | L 6 | Operating System GATE PYQs | GATE 2022

1,081 views · Streamed live on Aug 18, 2021

66 1 SHARE SAVE

Top chat replay

raahul raj zeher padhate ho sir aap DANDVAT PARNAM SIR

Lankesh Ravana sir reply tho kar do plss , big fan hu sir apka ❤️❤️🙏🙏

raahul raj I am watching these session recorded sir came here to ask timingof coa doubt ...college classes started sir on this time

Akrisht Kumar d

Rajesh Sah C

Rajesh Sah D

Nikhil Shanbhag D

Lankesh Ravana does deadlock implies starvation sir ?

Anand Singh A

Govind Yadav A

Lankesh Ravana ok sir understood thank u for answering

Rajesh Sah B

Anand Singh A

Rajesh Sah A,

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear 1:17 AM 10/26/2021

WhatsApp

Process Synchronization GAT

youtube.com/watch?v=JnCUP4I9H48&list=PLG9aCp4uE-s3kIregEhzbOBQDg5Ha0U38&index=6

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Slack | design-autu... aapatre/Automatic... Reading list

YouTube

Search

unacademy

GATE 2010

The following program consists of 3 processes and 3 binary semaphores. The semaphores are initialized as S0=1, S1=0, S2=0.

Process P0	Process P1	Process P2
<pre>while (true) { wait (S0); print '0'; release (S1); release (S2); }</pre>	<pre>wait (S1); release (S0);</pre>	<pre>wait (S2); release (S1);</pre>

How many times will process P1 print '0'?

A. At least twice ✓
B. Exactly twice ✓
C. Exactly three ✗
D. Exactly four ✗

S0 = 1 0 1 0 1 0
S1 = 0 1 0 1
S2 = 0 1 0 1

GATE 2018

52:12

52:37 / 1:05:51

#OperatingSystem_GATEPYQs #VishvadeepGothli #VirtualMemoryGATE_2006_2012

Process Synchronization GATE 2006-2012 | L 6 | Operating System GATE PYQs | GATE 2022

1,081 views · Streamed live on Aug 18, 2021

66 1 SHARE SAVE

Top chat replay

Lankesh Ravana hu sir apka sir reply tho kar do plss , big fan

rahul raj i am watching these session recorded sir came here to ask timing of coa doubt ...college classes started sir on this time

Akrisht Kumar d

Rajesh Sah C

Rajesh Sah D

Nikhil Shanbhag D

Lankesh Ravana does deadlock implies starvation sir ?

Anand Singh A

Govind Yadav A

Lankesh Ravana ok sir understood thank u for answering

Rajesh Sah B

Anand Singh A

Rajesh Sah A

Anand Singh C

HIDE CHAT REPLAY

Type here to search

100%

22°C Mostly clear

ENG

1:19 AM 10/26/2021

Process Synchronization GATE | Internet Speed Test | Fast.com

youtube.com/watch?v=JnCUP4l9H48&list=PLG9aCp4uE-s3klreqEhzbOBQDg5Ha0U38

Apps Settings Gmail YouTube Codeforces Youtube Download... (T) WhatsApp Contest - LeetCode Authorize access to... clist Slack | design-autu... aapatre/Automatic-... Reading list

YouTube

Search

unacademy

GATE 2012

Fetch_And_Add(X,i) is an atomic Read-Modify-Write instruction that reads the value of location X, increments it by the value i, and returns the old value of X. It is used in the p shown below to implement a busy-wait lock. L is an unsigned integer shared variable init The value of 0 corresponds to lock being available, while any non-zero value corresponds being not available.

```
AcquireLock(L){  
    while (Fetch_And_Add(L,1))  
        L = 1;  
}  
  
ReleaseLock(L){  
    L = 0;  
}
```

This implementation

- ☒ fails as L can overflow
- ☒ fails as L can take on a non-zero value when the lock is actually available
- ☒ works correctly but may starve some processes
- ☒ works correctly without starvation

L = 0 1 2 3 4 5 6 7

Top chat replay

Lankesh Ravana sir reply tho kar do plss , big fan hu sir apka

rahul raj i am watching these session recorded sir came here to ask timingof coa doubt ...college classes started sir on this time

Akrisht Kumar d

Rajesh Sah C

Rajesh Sah D

Nikhil Shanbhag D

Lankesh Ravana does deadlock implies starvation sir ?

Anand Singh A

Govind Yadav A

Lankesh Ravana ok sir understood thank u for answering

Rajesh Sah B

Anand Singh A

Rajesh Sah A

Anand Singh C

HIDE CHAT REPLAY

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2006_2012

Process Synchronization GATE 2006-2012 | L 6 | Operating System GATE PYQs | GATE 2022

1,081 views · Streamed live on Aug 18, 2021

66 1 SHARE SAVE

Type here to search

100% 22°C Mostly clear ENG 1:26 AM 10/26/2021

Process Synchronization GATInternet Speed Test | Fast.com

youtube.com/watch?v=2yIqMvQTYzY&list=PLG9aCp4uE-s3klreqEhzbOBQDg5Ha0U38&index=5

AppsSettingsGmailYouTubeCodeforcesYoutube Download...WhatsAppContest - LeetCodeAuthorize access to...cistStack | design-autu...aapatre/Automatic...Reading list

YouTube

Search

unacademy

GATE 2001

Consider Peterson's algorithm for mutual exclusion between two concurrent processes. The program executed by process *i* is shown below.

```
repeat
  flag[i] = true;
  turn = i;
  while (P) do no-op;
  Enter critical section, perform actions, then
  exit critical section
  flag[i] = false;
  Perform other non-critical section actions.
Until false;
```

For the program to guarantee mutual exclusion, the predicate P in the while loop should be

A. $\text{flag}[j] = \text{true and turn} = j$
B. $\text{flag}[j] = \text{true and turn} = i$
✓ C. $\text{flag}[i] = \text{true and turn} = j$
D. $\text{flag}[i] = \text{true and turn} = i$

Process *i*

$\text{flag}[j] \&\& \text{turn} == j$

$\text{flag}[i] = \text{false}$

Top chat replay

Live chat replay is on. Messages that appeared when the stream was live will show up here.

Rajal

good morning sir

Rutvik

good morning sir

Nikhil Shanbhag

Good morning sir

Rutvik

ready

Rutvik

ready

Sourabh Jain

yes sir

Shubham Kumar

sir coa ka class continue hoga kya?

Rutvik

yes @shubham

Vicky Kumar

sir UGC net ke prepration bhi krwa toeh ho

HIDE CHAT REPLAY

1,086 views · Streamed live on Aug 17, 2021

820

SHARE

SAVE

Type here to search

100%

22°C Mostly clear

ENG

1:30 AM

10/26/2021

Process Synchronization GATE 2003

$S = 1, T = 6$

Process P:

```
while(1) {
W: P(S);
print '0';
print '0';
X: V(T);
}
```

Process Q:

```
while(1) {
Y: P(T);
print '1';
print '1';
Z: V(S);
}
```

Synchronization statements can be inserted only at points W, X, Y, and Z.

Which of the following will always lead to an output starting with '0011'?

☒ A. P(S) at W, V(S) at X, P(T) at Y, V(T) at Z, S and T initially 1

☒ B. P(S) at W, V(T) at X, P(T) at Y, V(S) at Z, S initially 1, and T initially 0

☒ C. P(S) at W, V(T) at X, P(T) at Y, V(S) at Z, S and T initially 1

☐ D. P(S) at W, V(S) at X, P(T) at Y, V(T) at Z, S initially 1

10/31

unacademy

Process Synchronization GATE 2000-2010 | L 5 | Operating System GATE PYQs | GATE 2022

1,086 views · Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Top chat replay

Live chat replay is on. Messages that appeared when the stream was live will show up here.

Rajat good morning sir

Rutvik good morning sir

Nikhil Shanbhag Good morning sir

Rutvik ready

Rutvik ready

Sourabh Jain yes sir

Shubham Kumar sir coa ka class continue hoga kya?

Rutvik yes @shubham

Vicky Kumar sir UGC net ke prepration bhi krwa toeh ho

durgesh nandan sir I am going Inn2 md year some tips

Rajat b

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear 1:34 AM 10/26/2021

Process Synchronization GATE 2004

Semaphore All GATE Questions

WhatsApp

youtube.com/watch?v=2y1qMvQTYzY&list=PLG9aCp4uE-s3klreqEhbzOBQDg5Ha0U38&index=5

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Slack | design-autu... aapatre/Automatic... Reading list

YouTube

Search

unacademy

GATE 2004

Consider two processes P_1 and P_2 accessing the shared variables X and Y protected by two semaphores S_X and S_Y respectively, both initialized to 1. P and V denote the usual semaphore operators, where P decrements the semaphore value, and V increments the semaphore value. The pseudo-code of P_1 and P_2 is as follows:

P_1 :	P_2 :
While true do {	While true do {
L_1 :.....	L_3 :.....
L_2 :.....	L_4 :.....
$X = X + 1$;	$Y = Y + 1$;
$Y = Y - 1$;	$X = X - 1$;
$V(S_X)$;	$V(S_Y)$;
$V(S_Y)$;	$V(S_X)$;

In order to avoid deadlock, the correct operators at L_1 , L_2 , L_3 and L_4 are respectively

A. $P(S_Y), P(S_X); P(S_X), P(S_Y)$

B. $P(S_X), P(S_Y); P(S_Y), P(S_X)$

C. $P(S_X), P(S_X); P(S_Y), P(S_Y)$

D. $P(S_X), P(S_Y); P(S_X), P(S_Y)$

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2000_2010

Process Synchronization GATE 2000-2010 | L 5 | Operating System GATE PYQs | GATE 2022

1,086 views · Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Top chat replay

naajal: good morning sir

Rutvik: good morning sir

Nikhil Shanbhag: Good morning sir

Rutvik: ready

Rutvik: ready

Sourabh Jain: yes sir

Shubham Kumar: sir coa ka class continue hoga kya?

Rutvik: yes @shubham

Vicky Kumar: sir UGC net ke prepration bhi krwa taeh ho

durgesh nandan: sir i am going inn2 md year some tips

Rajat: b

Shweta Pandey: sir p t reason pls repeat on process q?

Rajat: C

Shweta Pandey: Sir suppose if we've 3print then opt will vary?

Shweta Pandey: like 3times print instead of 2times print

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear ENG 1:42 AM 10/26/2021

Process Synchronization GATE 2010Semaphore All GATE QuestionsWhatsApp

youtube.com/watch?v=2yIqMvQTYzY&list=PLG9aCp4uE-s3klreqEhbzOBQDg5Ha0U38&index=5

AppsSettingsGmailYouTubeCodeforcesYoutube Download...WhatsAppContest - LeetCodeAuthorize access to...cistStack | design-autu...aapatre/Automatic...Reading list

YouTube

Search

unacademy

GATE 2004

Consider two processes P_1 and P_2 accessing the shared variables X and Y protected by two binary semaphores S_X and S_Y respectively, both initialized to 1. P and V denote the usual semaphore operators, where P decrements the semaphore value, and V increments the semaphore value. The pseudo-code of P_1 and P_2 is as follows:

P_1 :	P_2 :
While true do {	While true do {
$L_1: P(S_X)$	$L_3: P(S_X)$
$L_2: P(S_Y)$	$L_4: P(S_Y)$
$X = X + 1;$	$Y = Y + 1;$
$Y = Y - 1;$	$X = X - 1;$
$V(S_X);$	$V(S_Y);$
$V(S_Y);$	$V(S_X);$

In order to avoid deadlock, the correct operators at L_1, L_2, L_3 and L_4 are

- ☒ $P(S_Y), P(S_X); P(S_X), P(S_Y)$
- ☒ $P(S_X), P(S_Y); P(S_Y), P(S_X)$
- ☒ $P(S_X), P(S_X); P(S_Y), P(S_Y)$
- ☐ $P(S_X), P(S_Y); P(S_X), P(S_Y)$

Handwritten notes: $P_1 \rightarrow$ and P_2 with arrows pointing to the respective process code blocks.

#OperatingSystem_GATEPYQs #VishvadeepGothli #VirtualMemoryGATE_2000_2010

Process Synchronization GATE 2000-2010 | L 5 | Operating System GATE PYQs | GATE 2022

1,086 views · Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Top chat replay

Nikhil Shanbhag Good morning sir

Rutvik ready

Rutvik ready

Sourabh Jain yes sir

Shubham Kumar sir coa ka class continue hoga kya?

Rutvik yes @shubham

Vicky Kumar sir UGC net ke prepration bhi krwa taeh ho

durgesh nandan sir i am going inn2 md year some tips

Rajrat b

Shweta Pandey sir p t reason pls repeat on process q?

Rajrat C

Shweta Pandey Sir suppose if we've 3print then opt will vary?

Shweta Pandey like 3times print instead of 2times print

Rajrat d?

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear ENG 1:43 AM 10/26/2021

unacademy

GATE 2004

The semaphore variables full, empty and mutex are initialized to 0, n and 1, respectively. Process P_1 repeatedly adds one item at a time to a buffer of size n , and process P_2 repeatedly removes one item at a time from the same buffer using the programs given below. In the programs, K , L , M and N are unspecified statements.

```

while (1) {
    K; wait(empty); // Producer
    P(mutex);
    Add an item to the buffer;
    V(mutex);
    L; signal(full);
}

while (1) {
    M; // Consumer
    P(mutex);
    Remove an item from the buffer;
    V(mutex);
    N;
}
    
```

full = no. of full spaces
empty = no. of empty spaces
mutex = mutual exclusion

The statements K , L , M and N are respectively

- $P(full)$, $V(empty)$, $P(full)$, $V(empty)$
- $P(full)$, $V(empty)$, $P(empty)$, $V(full)$
- $P(empty)$, $V(full)$, $P(empty)$, $V(full)$
- $P(empty)$, $V(full)$, $P(full)$, $V(empty)$

Top chat replay

kya?

Rutvik yes @shubham

Vicky Kumar sir UGC net ke preparation bhi krwa taeh ho

durgesh nandan sir I am going inn2 md year some tips

Rajrat b

Shweta Pandey sir p t reason pls repeat on process q?

Rajrat C

Shweta Pandey Sir suppose if we've 3print then opt will vary?

Shweta Pandey like 3times print instead of 2times print

Rajrat d?

Dipalok Banarjee_23 n

Dipalok Banarjee_23 b

Operating System apki pichli saal wali COA ki videos dek ra hu...apne sahi me bola tha ...ki mai COA ka bhoot bhaga deta hu...sahi me boht jbrdst pdaya h sir aapne

Rajrat b?

HIDE CHAT REPLAY

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2000_2010

Process Synchronization GATE 2000-2010 | L 5 | Operating System GATE PYQs | GATE 2022

1,086 views · Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Process Synchronization GATE 2005

Semaphore All GATE Questions

WhatsApp

youtube.com/watch?v=2yIqMvQTYzY&list=PLG9aCp4uE-s3klreqEhbzOBQDg5Ha0U38&index=5

Apps Settings Gmail YouTube Codeforces Youtube Download... (1) WhatsApp Contest - LeetCode Authorize access to... clist Slack | design-autu... aapatre/Automatic... Reading list

YouTube

Search

unacademy

GATE 2005

Given below is a program which when executed spawns two concurrent processes :

semaphore X := 0 ;
/* Process now forks into concurrent processes P1 & P2

P1

repeat forever
V (X) ;
Compute ;
P(X) ;

P2

repeat forever
P(X) ;
Compute ;
V(X) ;

Consider the following statements about processes P1 and P2 :

I. It is possible for process P1 to starve
II. It is possible for process P2 to starve

Which of the following holds?

A. Both I and II are true.
B. I is true but II is false.
C. II is true but I is false
D. Both I and II are false

40:09

38:19 / 1:12:55

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2000_2010

Process Synchronization GATE 2000-2010 | L 5 | Operating System GATE PYQs | GATE 2022

1,086 views · Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Top chat replay

Vicky Kumar sir UGC net ke prepration bhi krwa taeh ho

durgesh nandan sir i am going inn2 md year some tips

Rajat b

Shweta Pandey sir p t reason pls repeat on process q?

Rajat C

Shweta Pandey Sir suppose if we've 3print then opt will vary?

Shweta Pandey like 3times print instead of 2times print

Rajat d?

Dipalok Banarjee_23 n

Dipalok Banarjee_23 b

Operating System: apki pichli saal wali COA ki videos dek ra hu...apne sahi me bola tha ..ki mai COA ka bhoot bhaga deta hu...sahi me boht jrdst pdaya h sir aapne

Rajat b?

Rajat d

HIDE CHAT REPLAY

Type here to search

100% 22°C Mostly clear ENG 1:46 AM 10/26/2021

unacademy

GATE 2005

Given below is a program which when executed spawns two processes :

```
semaphore X := 0;
/* Process now forks into concurrent processes P1 & P2 */
```

P1

```
repeat forever
V(X);
Compute ;
P(X);
```

P2

```
repeat forever
P(X);
Compute ;
V(X);
```

Consider the following statements about processes :

- It is possible for process P1 to starve. ✓
- It is possible for process P2 to starve. ✓

Which of the following holds?

- Both I and II are true. ✓
- I is true but II is false.
- II is true but I is false
- Both I and II are false

Top chat replay

- durgesh nandan sir i am going inn2 mid year some tips
- Rajat: b
- Shweta Pandey sir p t reason pls repeat on process q?
- Rajat: C
- Shweta Pandey Sir suppose if we've 3print then opt will vary?
- Shweta Pandey like 3times print instead of 2times print
- Rajat: d?
- Dipalok Banarjee_23 n
- Dipalok Banarjee_23 b
- Operating System: apki pichli saal wali COA ki videos dek ra hu... apne sahi me bola tha... ki mai COA ka bhoot bhaga deta hu... sahi me boht jbrdst pdaya h sir aapne
- Rajat: b?
- Rajat: d
- Rajat: a?
- Rutvik: A

HIDE CHAT REPLAY

#OperatingSystem_GATEPYQs #VishvadeepGothi #VirtualMemoryGATE_2000_2010

Process Synchronization GATE 2000-2010 | L 5 | Operating System GATE PYQs | GATE 2022

1,086 views · Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Process Synchronization GATE 2005 Sol.

2 semaphores A, B
 $A = \phi + \phi + 0$
 $B = \phi + 0$

P1	P2
Use R1	Use R1
Use R2	Use R2
Use R3	Use R3
Use R4	Use R4

wait(A)
Use R1
Use R2
signal(B)
wait(A)
Use R3
Use R4
signal(B)
wait(A)
Use R1
Use R2
signal(A)
wait(A)
Use R3
Use R4
signal(B)
wait(A)
Use R1
Use R2
signal(A)
wait(A)
Use R3
Use R4
signal(B)

Top chat replay

- Dipalok Banarjee_23 b
- Operating System apki pichli saal wali COA ki videos dek ra hu... apne sahi me bola tha ..ki mai COA ka bhoot bhaga deta hu... sahi me boht jirdst pdaya h sir apne
- Rajrat b?
- Rajrat d
- Rajrat a?
- Rutvik A
- umesh pathak A
- Boya Pushpalatha sir pls explian it in english
- SHUBHAM KUMAR 2 semaphore
- SHUBHAM KUMAR *binary
- umesh pathak 2 semaphore
- Rutvik sir me Abhi 5 th sem me hu or mere university exam bhi offline he to me GATE ke liye is bar kitne subject prepare karu
- SHUBHAM KUMAR signal (a). in p1.
- SHUBHAM KUMAR b*

1,086 views • Streamed live on Aug 17, 2021

82 0 SHARE SAVE

Type here to search

100% 22°C Mostly clear 1:57 AM 10/26/2021

GATE 2006

The atomic *fetch-and-set* x, y instruction unconditionally sets memory location x to 1 and fetches the old value of x in y without allowing any intervening access to the memory location x . Consider the following implementation of P and V functions on a binary semaphore S.

```
void P (binary_semaphore *s) {  
    unsigned y;  
    unsigned x = &(s->value);  
    do {  
        fetch-and-set x, y;  
    } while (y);  
}  
  
void V (binary_semaphore *s) {  
    S->value = 0;  
}
```

$S = +01$
↑
x

Which one of the following is true?

- ☒ A. The implementation may not work if context switch occurs.
- ☐ B. Instead of using *fetch-and-set*, a pair of normal instructions would work.
- ☐ C. The implementation of V is wrong.
- ☐ D. The code does not implement a binary semaphore.

GATE 2006

19/31

Barrier is a synchronization construct where a set of processes synchronizes globally i.e., each process in the set arrives at the barrier and waits until all processes arrive and then all processes leave the barrier. Let the number of processes in the set be n . Let S be a binary semaphore with the usual P and V functions. Consider the following implementation of a barrier with line numbers shown next to it.

```
void barrier(void) {
```

```
    process_arrived = 0;
    V(S);
    while (process_arrived != n)
        P(S);
    process_left = 0;
    if (process_arrived == n)
        process_left = n;
    V(S);
}
```

GATE 2006

The variables *process_arrived* and *process_left* are shared among all processes and are initialized to zero. In a concurrent program all the three processes call the barrier function when they want to synchronize globally.

The above implementation of barrier is incorrect. Which one of the following is true?

- A. The barrier implementation is wrong due to the use of binary semaphore S
- B. The barrier implementation may lead to a deadlock if two barrier invocations are used in immediate succession.
- C. Lines 6 to 10 need not be inside a critical section
- D. The barrier implementation is correct if there are only two processes instead of three.



GATE 2006

Barrier is a synchronization construct where a set of processes synchronizes globally i.e., each process in the set arrives at the barrier and waits for others to arrive and then all processes leave the barrier. Let the number of processes be n . Let $process_arrived$ and S be a binary semaphore with the usual P and V functions. Consider the implementation of a barrier with line numbers shown on left.

```
void barrier (void) {
```

```
1. P(S);  
2. process_arrived++;  
3. V(S);  
4. while (process_arrived != n);  
5. P(S);  
6. process_left++;  
7. if (process_left == n) {  
8.     process_arrived = 0;  
9.     process_left = 0;  
10. }  
11. V(S);  
12. }
```

$process_arrived = 4$
 $process_left = 1$

GATE 2006

The variables `process_arrived` and `process_left` are shared among all processes and are initialized to zero. In a concurrent program all the three processes call the barrier function when they need to synchronize globally.

Which one of the following rectifies the problem in the implementation?

- A. Lines 6 to 10 are simply replaced by `process_arrived--`
- B. At the beginning of the barrier the first process to enter the barrier waits until `process_arrived` becomes zero before proceeding to execute `P(S)`.
- C. Context switch is disabled at the beginning of the barrier and re-enabled at the end.
- D. The variable `process_left` is made private instead of shared



GATE 2006

The variables `process_arrived` and `process_left` are shared among all processes and are initialized to zero. In a concurrent program all the three processes call the barrier function when they need to synchronize globally.

Which one of the following rectifies the problem in the implementation?

- A. Lines 6 to 10 are simply replaced by `process_arrived--`
- ☒ B. At the beginning of the barrier the first process to enter the barrier waits until `process_arrived` becomes zero before proceeding to execute `P(S)`.
- C. Context switch is disabled at the beginning of the barrier and re-enabled at the end.
- D. The variable `process_left` is made private instead of shared

GATE-2001

Two concurrent processes P_1 and P_2 want to use resources R_1 and R_2 in a mutually exclusive manner. Initially, R_1 and R_2 are free. The programs executed by the processes are given below.

Program for P_1 :

S1: While (R_1 is busy) do no-
op;
S2: Set $R_1 \leftarrow$ busy;
S3: While (R_2 is busy) do no-
op;
S4: Set $R_2 \leftarrow$ busy;
S5: Use R_1 and R_2 ;
S6: Set $R_1 \leftarrow$ free;
S7: Set $R_2 \leftarrow$ free;

Program for P_2 :

Q1: While (R_1 is busy) do no-
op;
Q2: Set $R_1 \leftarrow$ busy;
Q3: While (R_2 is busy) do no-
op;
Q4: Set $R_2 \leftarrow$ busy;
Q5: Use R_1 and R_2 ;
Q6: Set $R_2 \leftarrow$ free;
Q7: Set $R_1 \leftarrow$ free;

- A. Is mutual exclusion guaranteed for R_1 and R_2 ? If not show a sequence of statements of P_1 and P_2 such that mutual exclusion is violated (i.e., both P_1 and P_2 use R_1 and R_2 at the same time). $S_1, Q_1, S_2, Q_2, S_3, Q_3, S_4, Q_4, S_5, Q_5$
- B. Can deadlock occur in the above program? If yes, show a possible sequence of P_1 and P_2 leading to deadlock.
- C. Exchange the statements Q_1 and Q_3 and repeat the above questions. Is mutual exclusion guaranteed now? Can deadlock occur?

GATE-2001

Two concurrent processes P_1 and P_2 want to use resources R_1 and R_2 in an exclusive manner. Initially, R_1 and R_2 are free. The programs executed by the processes are given below.

Program for P_1 :	Program for P_2 :
S1: While (R_1 is busy) do no-op;	Q1: While (R_2 is busy) do no-op;
S2: Set $R_1 \leftarrow$ busy;	Q2: Set $R_2 \leftarrow$ busy;
S3: While (R_2 is busy) do no-op;	Q3: While (R_1 is busy) do no-op;
S4: Set $R_2 \leftarrow$ busy;	Q4: Set $R_1 \leftarrow$ busy;
S5: Use R_1 and R_2 ;	Q5: Use R_1 and R_2 ;
S6: Set $R_1 \leftarrow$ free;	Q6: Set $R_2 \leftarrow$ free;
S7: Set $R_2 \leftarrow$ free;	Q7: Set $R_1 \leftarrow$ free;

- Is mutual exclusion guaranteed for R_1 and R_2 ? (Consider the statements of P_1 and P_2 such that mutual exclusion is violated. $S_1, Q_1, S_2, Q_2, S_3, Q_3$).
- Can deadlock occur in the above program? If yes, give the sequence of P_1 and P_2 leading to deadlock. *No deadlock*
- Exchange the statements Q_1 and Q_3 . Is mutual exclusion guaranteed now? Can deadlock occur?

→ Mutual Exclusion holds. Deadlock possible

GATE-2004

In a certain operating system, deadlock prevention is implemented using the following scheme. Each process is assigned a unique timestamp, and is restarted with the same timestamp if killed. Let P_h be the process holding a resource R , P_r be a process requesting the same resource R , and $T(P_h)$ and $T(P_r)$ be their timestamps respectively. The decision to preempt one of the processes is based on the following algorithm.

```
if  $T(P_r) < T(P_h)$  then  
    kill  $P_r$   
else wait
```

Which one of the following is TRUE?

- A. The scheme is deadlock-free, but not starvation-free.
- B. The scheme is not deadlock-free, but starvation-free.
- C. The scheme is neither deadlock-free nor starvation-free.
- D. The scheme is both deadlock-free and starvation-free.

GATE-2004

In a certain operating system, deadlock prevention is attempted using the following algorithm. Each process is assigned a unique timestamp, and is restarted with the same timestamp. Let P_h be the process holding a resource R , P_r be a process requesting for the same resource, and $T(P_r)$ and $T(P_h)$ be their timestamps respectively. The decision to wait or preempt one of them is based on the following algorithm.

```
if  $T(P_r) < T(P_h)$  then  
    kill  $P_r$   
else wait
```

} → It prevents circular wait

Which one of the following is TRUE?

- ☒ A. The scheme is deadlock-free, but not starvation-free
- ☐ B. The scheme is not deadlock-free, but starvation-free
- ☐ C. The scheme is neither deadlock-free nor starvation-free
- ☐ D. The scheme is both deadlock-free and starvation-free

GATE-2005

Suppose n processes P_1, P_2, \dots, P_n share m identical resource units, which can be reserved and released one at a time. The maximum resource requirement of process P_i is s_i , where $s_i > 0$. Which one of the following is a sufficient condition for ensuring that deadlock does not occur?

- A. $\forall i, s_i < m$
- B. $\forall i, s_i < n$
- C. $\sum_{i=1}^n s_i < (m+n)$
- D. $\sum_{i=1}^n s_i < m$

	max	allocation
P_1	s_1	$s_1 - 1$
P_2	s_2	$s_2 - 1$
		$s_3 - 1$

To avoid deadlock
 $(s_1 - 1) + (s_2 - 1) + (s_3 - 1) + 1$

$$\sum_{i=1}^n s_i - n + 1 \leq m$$

$$\sum_{i=1}^n s_i + 1 \leq (m + n)$$

GATE-2005

Two shared resources R_1 and R_2 are used by processes P_1 and P_2 . Each process has a certain priority for accessing each resource. Let T_{ij} denote the priority of process P_i for accessing R_j . A process P_i can snatch a resource R_k from process P_j if T_{ik} is greater than T_{jk} . Given the following :

- I. $T_{11} > T_{21}$ ✓
- II. $T_{12} > T_{22}$ ✓
- III. $T_{11} < T_{21}$ ✓
- IV. $T_{12} < T_{22}$ ✓

Which of the following conditions ensures mutual exclusion?

- ~~A. I and IV~~
- ~~B. II and III~~
- ~~C. I and II~~
- D. None of the above

GATE-2006

no resource is free

Consider the following snapshot of a system with n processes. Process i is holding x_i instances of a resource R , $1 \leq i \leq n$. Currently, y_k instances of R are occupied. Further, for all i , process i has placed a request for an additional y_i instances while holding the x_i instances it already has. There are exactly two processes p and q such that $Y_p = Y_q = 0$. Which one of the following can serve as a necessary condition to guarantee that the system is not approaching a deadlock?

- A. $\min(x_p, x_q) < \max_{k \neq p, q} y_k$
- B. $x_p + x_q \geq \min_{k \neq p, q} y_k$
- C. $\max(x_p, x_q) > 1$
- D. $\min(x_p, x_q) > 1$

$$\min(y_k) \leq (x_p + x_q)$$

Process	Request	Available
p	$y_p = 0$	0
	0	x_p After p
		$(x_p + x_q)$ After q



x y z
5 5 5

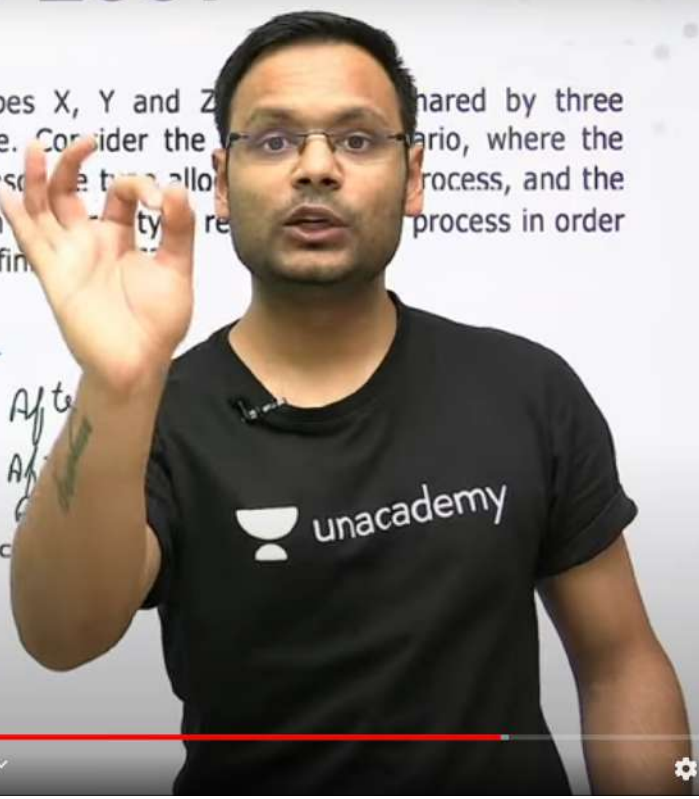
GATE-2007

A single processor system has three resource types X, Y and Z shared by three processes. There are 5 units of each resource type. Consider the scenario, where the column **alloc** denotes the number of units of each resource type allocated to a process, and the column **request** denotes the number of units of each resource type requested by a process in order to complete execution. Which of these processes will finish its execution?

	alloc			request		
	X	Y	Z	X	Y	Z
P0	1	2	1	1	0	3
P1	2	0	1	0	1	2
P2	2	2	1	1	2	0

Available		
X	Y	Z
0	1	2
2	1	3
3	3	4
5	5	5

- A. P0
- B. P1
- ☒ C. P2
- D. None of the above, since the system is in a deadlock



GATE-2008

Which of the following is NOT true of deadlock prevention and avoidance schemes?

- ☒ A. In deadlock prevention, the request for resources is granted if the resulting state is safe
- ☒ B. In deadlock avoidance, the request for resources is granted if the resulting state is safe
- ☒ C. Deadlock avoidance is less restrictive than deadlock prevention
- ☐ D. Deadlock avoidance requires knowledge of resource requirements



51:50

50:59 / 59:01

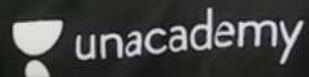


GATE-2008

An operating system implements a policy that requires a process to release all resources before making a request for another resource. Select the TRUE statement from the following:

- A. Both starvation and deadlock can occur
- B. Starvation can occur but deadlock cannot occur
- C. Starvation cannot occur but deadlock can occur
- D. Neither starvation nor deadlock can occur

→ Trying to prevent hold



53:51 / 59:01



GATE-2008

An operating system implements a policy that process to release all resources before making a request for another resource. Select the correct statement from the following:

- A. Both starvation and deadlock can occur
- ☒ B. Starvation can occur but deadlock cannot occur
- C. Starvation cannot occur but deadlock can occur
- D. Neither starvation nor deadlock can occur

→ Trying to

unacademy



55:45

54:41 / 59:01