PAGE NO. :
DATE : / /

Unit III

Q5(a) Define Synchronization Explain with Critical Section Problem & their requirement

(b) What is Semaphore & monitor

Q6(a) What is deadlock Give their Characteristics of Detention. Prevition.

(b) How to recover deadlock explain with banker algo.

Ans5(a) Synchronization ⇒ When more than one process work at a time. Due to this all Process have race Condition which can be solved by Synchronization. So that it can Solved deadlock condition.

Type :-

1) Critical Section Problem
2) Semaphore
3) Monitor

1) Critical Section Problem ⇒ It can given one Process at a time. It use Shared variable It maintain Synchronization.

PAGE NO. :
DATE : / /

Syntax :-

```
do {
    entry section
        Critical section
    exit section
        remainder section
} while (TRUE);
```

Critical Section have to satisfic following three condition :-

i) Mutual Exclusion ⇒ If any Process running in critical section. then it can't allow another process

ii) Progress ⇒ If any process is not running in critical section Then we can't stop any Process to come insi Critical section.

iii) Bounded waiting ⇒ In this we give decided timing for any Process.

Types of Algorithm

i) Algo1
ii) Algo 2
iii) Algo 3

PAGE NO.:
DATE: / /

iii) Algo1 ⇒ It can solve any problem It use single variable like "Turn" If Turn=0 then 1 process works If Turn=1 then other process will work. When 1 process is working then other process can't enter.

iii) Algo2 ⇒ It use array type flag variable which index position is changed by "i". If flag[I]=true then process is work otherwise process can't work. It can run more then 1 process

iii) Algo3 ⇒ It work with algo1 & algo2

do{

```
Flag[i]=true;
while(flag[j]);
```

Critical section

```
Flag[i]=false;
```

remainder section

}while(1);

PAGE NO.:
DATE: / /

Ans(b) Semaphore & monitor ⇒

Definition ⇒ When we share resource for two or more than two process then more than one process move to the critical section. Due to this busy signal will get of CPU time will waste. For this we use semaphore

Method ⇒

1) Semaphore work as a flag which can be on or off.
2) It use wait & signal two variable. Signal start the process & wait will be use for wait for process.
3) It work as a spinlock which can work b/w wait & signal. It can handle process.
4) wait can terminate signal until zero is found then it use signal. Signal will increment & wait will wait.
5) Signal will initilize the process for run. & wait will give wait to the process.
6) If we want to stop collision b/w two train & then we give chance to the one train & other

PAGE NO.:
DATE: / /

Will be wait.
Ex ⇒ Producer Consumer

equation :- wait(s)
{
while(s<=0); /* No operation */
s--;
}

★ Monitor

It use in Concurrent Programming. It use more than one thread. It Give signal to the every thread. It use condition variable. It is a Shared data structure. It is abstract data type. It will helpful to run only one process at a time. So, that it stop deadlock condition.

PAGE NO.:
DATE: / /

	Semaphore	Monitor
i	It use integer variable S	It is a abstract data type.
ii	It Give Share resource info.	It Give Share variable & method.
iii	It use wait & signal variable	It use Monitor variable
iv	It have Condition variable	It have not Condition variable.

Monitor ⇒ It is programming language which can Control Share data. It Control Synchronization b/w more than one process. It give Permission to run one thread at a time. It use waiting queue to run more than one process. Condition variable use three operation.

i) wait
ii) signal
iii) Broadcast

- Broadcast ⇒ The wake up all waiting thread.
- Read or write Problem
- Hoare monitor ⇒ It use if Condition

PAGE NO. :
DATE : / /

- Mesa monitor → It use while condition

* In Java use monitor it can't stop more than one thread to run.

- Hoare

```
if(empty);
```

- mesa

```
while(empty)
wait(condition);
```

Monitor Queues :-

```
Monitor bounded buffer{
    Condition not_full;
    .... other variables
    Condition not empty;
    void put_resource(){
    .... wait(not_full) ....
    .... Signal(not_empty);
    }
    Resource get_resource(){
    }
```

PAGE NO. :
DATE : / /

Ans 6(a) Deadlock

Defination → When we use more than one process at a time due to this all process use to resource, due to this all process have to wait for resource which is called deadlock condition.

Deadlock Condition → It have four condition

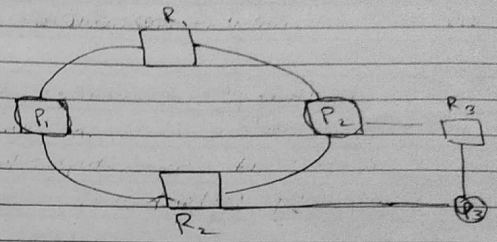
1) Mutual Exclusion
2) Hold & wait
3) No Preemption
4) Circular wait.

1) Mutual Exclusion → When all process are using there resource. but their are not sharing that it increase deadlock condition because any process has to wait for the resource.
Ex:- Five Process, Four resource.

2) Hold & wait → when any process hold any resource & wait of other resource than it is called hold & wait condition Due to this it increase waiting.

3) No Preemption → If any process is not using any resource & do not bent to free any resource forcefully than it increase deadlock. it is called no Preemption.

(iv) Circular wait → When one process wait for any resource & this resource is requested by any process then this condition is called circular wait



Deadlock Prevition (method for handeling deadlock) → We can handle deadlock using two method:-

(i) Direct → It use circular wait concept.

(ii) Indirect → It include mutual exclusions hold & wait & no Preemption.

(a) Prevent for mutual Exclusion → When we want to share any resource by more & more process then it is necessary to stop sharing otherwise it give mutual exclusion Condition to prevent mutual exclu

Problem in deadlock we make two types of resource.

(a) Shareable Resource
(b) Non-Shareable Resource

In Shareable resource, resource can be share but in non Shareable resource it can stop sharing using read only attribute.

(2) Prevent hold & wait → If any process request for any resource then if this resource is not used by any process, if any process want to use other resource which is new resource then it is necessary to leave previous resource which is not used by process. If any process wait for any resource for long time then it is called Starvation.

(3) Prevent No-Premption → In this we will stop any process to use resource which is used by other process. It is necessary to leave any

process resource before using other resource.

If any process have a resource which is not used by any process then we will free from force the resource from process which are not using for long time so that this resource can be used by another process.

(4) Circular wait Prevent → In this we make the resource allocation graph & check the cycle blw process & resource. It remove the cycle blw all graph & then make the list of process & resource. then allocate resource according to their process.

Ans(b) Deadlock Recovery

If deadlock is deducted then we use deadlock recovery technique so that over system will work. It is called deadlock recovery.

PAGE NO. :
DATE : / /

Types of method in deadlock recovery

1) Deadlock Recovery through Premption
2) Deadlock Recovery through Roll back
3) Deadlock Recovery through killing Process

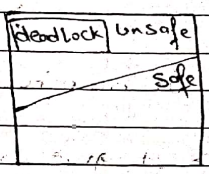
1) Deadlock recovery through Premption ⇒ In this we will take the resource away from process. So, that the other Process can use resource. it is very difficult process.

2) Deadlock recovery through Roll back ⇒ When ever deadlock is detected then we understand that which resource are require So that we roll back the deadlock Steps until deadlock is remove at any specific position.

3) Deadlock recovery through killing Process ⇒ In this we will kill resource and Process for recovery. It can be done using two method
i) Using / killing one by one Process at a time
ii) kill all Process at a time

PAGE NO. :
DATE : / /

It have two state
i) Safe State ⇒ If deadlock is not found
ii) Unsafe State ⇒ If deadlock is found



Bankers Algo

Introduction ⇒ When we use resource allocation graph then it require do not bankers algo.

⇒ It use in banking System like allocation of Cash in bank So, that Cash will we allocate according to requirement So, that Shortage of Cash is not found all Customers are satisfied after getting Cash.

Processing
⇒ When new Process enter in System

then we declare the process & allocate resource.

2) System should be in safe state so, that resource can be allocate otherwise some resources can't allocated to the process.

States → It have four states

1) Available → It give no. of resource which is provided.

2) Max → It give maximum demand from process. So, that maximum resource can be used.

3) Allocation → It give no of resource which will be allocated. It allocate one or more resource at a time.

4) Need → It give no. of resource which is necessary to used by any process.

SAFTY Algo

Step-1 → Work = Available work & Finished work = false

Step 2 - If table is finished then it require need.

Step3 - Link - work = allocation

Step4 - Finish work = true

Resource Request Algo

Step1 - It request less than need the error can be found because it require to be more

Step2 - If request greater than available work then we have to wait

Example

Process	Allocation	Have ABC	Available ADC
P_0	A B C	7 6 3	7 3 2
P_1	0 1 0	3 2 2	
P_2	2 0 0	1 0 2	
P_3	3 0 2	2 2 2	
P_4	2 1 1	4 3 3	
	0 0 2		

Request = (1, 0, 2)

Available = Available - Request

Need = Max - allocation

Req = (1, 0, 2)

Available = Available - Request.

Need = Max - allocation

Process	Allocation	Need	available
	ABC	ABC	ABC
P_0	010	743	230
P_1	302	020	
P_2	302	600	
P_3	602	011	
P_4			

Alc = Alc + Req

	Need	Need = Need - Req
	ABC	
P_0	743	
P_1	122	
P_2	600	

} (80)