Tribhuvan University
Institute of Science and Technology
2078
✡

Bachelor Level / Second Year/ Forth Semester/ Science
Computer Science and Information Technology (CSC 259)
(Operating Systems)

Full Marks: 60
Pass Marks: 24
Time: 3 hours.

Candidates are required to give their answers in their own words as for as practicable.
All figures in the margin indicate full marks.

Attempt all the questions.

Section A

Long Answer Questions
Attempt any Two questions

(2x 10=20)

1 What kind of problem arises with sleep and wakeup mechanism of achieving mutual exclusion? Explain with suitable code snippet.

2 Why OPR is best but not pratically feasible page replacement algorithm? Calculate the number of page faults for OPR, LRU, and Clock page replacement algorithm for the reference string: 1, 3, 4, 2, 3, 5, 4, 3, 1, 2, 4, 6, 3, 2, 1, 4, 2. Assume that memory size is 3.

3 How unsafe state differs from deadlocked state? Consider following initial state and identify wheather requested resource is and granted or denies for the given cases.

Process	Has	Max
A	2	6
B	1	5
C	2	3
D	3	8

Free=2

- What will happen if process D request 1 resource?
- What will happen if process A request 1 resource?

Section B

Short Answer Questions
Attempt any Eight questions.

(8x5=40)

4. What is system call? Discuss process of handling system calls briefly.

5. What is lock variable? Discuss its working and problems associated with it in detail.

6. Differentiate between internal and external fragmentation? Suppose that we have memory of 1000 KB with 5 partitions of size 150 KB, 200 KB, 250 KB, 100 KB, and 300 KB. Where the processes A and B of size 175 KB and 125 KB will be loaded, if we used Best-Fit, and Worst-Fit Strategy? (2)

7. What is meant by file attributes? Discuss any one technique of implementing directories in detail. (4)

8. Why the concept of disk interleaving is important? Explain with suitable example.

9. What is resource allocation graph? Explain the process of detecting deadlocks when there is single instance of each resource with suitable example? (4)

10. Discuss the concept of SJF and SRTN scheduling algorithms with suitable example. (2)

11. What approaches are used for managing free disk spaces? Explain linked list approach with example. (2)

12. Write short notes on:
    - IPC in Linux
    - Disk access

Resoure allocation is the process of allocating the resorce to process which are requesting them. So the diagramdically representation of the allocation of the resource and the resources they are holding is known as Resource all

1) Vertex
2) edge

Operating System-2078
-Ankit Pangeni DATE

Section A

Q.1. What kind of problem arises with sleep and wakeup mechanism of achieving mutual exclusion? Explain with suitable code snippet.

⇒ Mutual exclusion means only one process can be inside the critical section at any time. So that race condition won't occur. Sleep and wakeup is a mechanism for achieving mutual exclusion.

Let's take an example of producer consumer problem that uses sleep & wake mechanism. It's code snippet is:

```
#define BUFFER_SIZE 100
int itemCount=0;
void producer()
    {
    while(true){
        item=produceItem();
    if(itemCount == BUFFER_SIZE){sleep;}
    putItemInto Buffer(item);
    itemCount = itemCount+1;
    if(itemCount==1) {wakeup(consumer);}
            } }

void consumer() {
        while(true) {
```

DATE

```
if (itemCount==0) { sleep(); }
    item = removeItemFromBuffer();
    itemCount = itemCount - 1;
    if (itemCount == BUFFER_SIZE-1)
        { wakeup (producer); }

    consumeItem(item);
}
}
```

Here, the problem with sleep & wakeup mechanism is that it contains a race condition that can lead to a deadlock. Consider following scenario:

- The consumer has just read the variable itemCount, noticed it's zero and just about to move inside the if-block.
- Just before calling sleep, the consumer is interrupted & the producer is resumed.
- The producer creates an item, puts it into the buffer & increases itemCount.
- Because the buffer was empty, prior to the last addition, the producer tries to wakeup consumer.
- Unfortunately, consumer wasn't yet sleeping & the wakeup call is lost. When the consumer resumes, it goes to sleep & will never be awakened again. This is because consumer is only awakened by the producer when itemCount

DATE

- The producer will loop until the buffer is full and will go to sleep.
- Since both process will sleep forever, we have run into deadlock. This mechanism therefore is unsatisfactory.

Q.2. Why OPR is best but not practicaly feasible page replacement algorithm? Calculate the no. of page faults of OPR, LRU and clock page replacement algorithm for the reference string: 1, 3, 4, 2, 3, 5, 4, 3, 1, 2, 4, 6, 3, 2, 1, 4, 2. Assume memory size is 3.

=> Optimal page replacement Algorithm (OPR) is one of the best page replacement algorithms as it has the lowest page fault occurance rate than any other page replacement algorithm.

But it is not practicaly feasible because in OPR we replace the page which will not be used for longest time in near future - but the operating system in real life doesn't know any future requests. It is just theoriticaly possible & only used as a benchmark to compare with other page replacement algorithms.

classmate PAGE

Using OPR

DATE

Given, reference String:
1, 3, 4, 2, 3, 5, 4, 3, 1, 2, 4, 6, 3, 2, 1, 4, 2
f_i: i^{th} frame

	1	3	4	2	3	5	4	3	1	2	4	6	3	2	1	4	2
f_3			4	4	4	4	4	4	4	4	4	6	6	6	1	4	4
f_2		3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
f_1	1	1	1	2	2	5	5	5	1	2	2	2	2	2	2	2	2
	*	*	*	*	H	*	*	H	*	*	H	*	H	H	*	*	H

∴ Total page faults = 10
& Total page hits = 7
(If asked page fault ratio = $\frac{10}{17} \times 100\%$)

Using LRU

String: 1, 3, 4, 2, 3, 5, 4, 3, 1, 2, 4, 6, 3, 2, 1, 4, 2

	1	3	4	2	3	5	4	3	1	2	4	6	3	2	1	4	2
f_3			4	4	4	5	5	5	1	1	1	6	6	6	1	1	1
f_2		3	3	3	3	3	3	3	3	3	4	4	4	2	2	2	2
f_1	1	1	1	2	2	2	4	4	4	2	2	2	3	3	3	4	4
	*	*	*	*	H	*	*	H	*	*	*	*	*	*	*	*	H

∴ Total page faults = 14
Total page hits = 3

Note: If asked, page fault ratio = $\frac{14}{17} \times 100\%$

Page hit ratio = $\frac{3}{17} \times 100\%$

PAGE

classmate

DATE

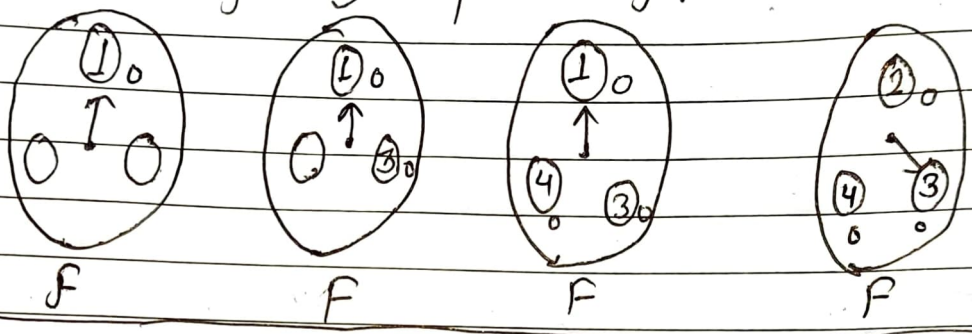
Using clock page replacement algorithm

Input Strings: 1, 3, 4, 2, 3, 5, 4, 3, 2, 2, 4, 6, 3, 2, 1, 4, 2

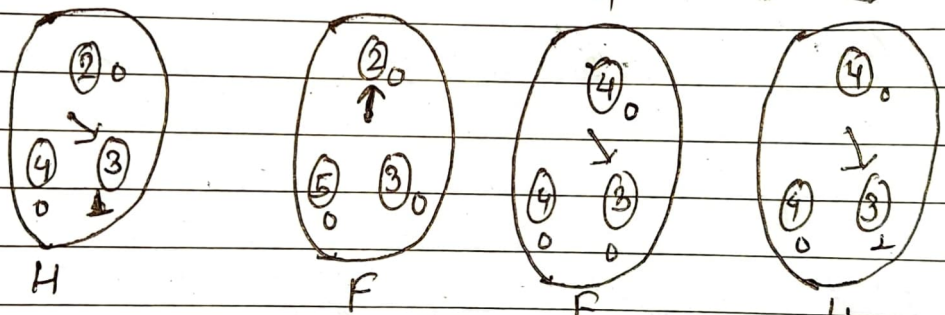
Size of memory = 3.

F = fault, H = Hit

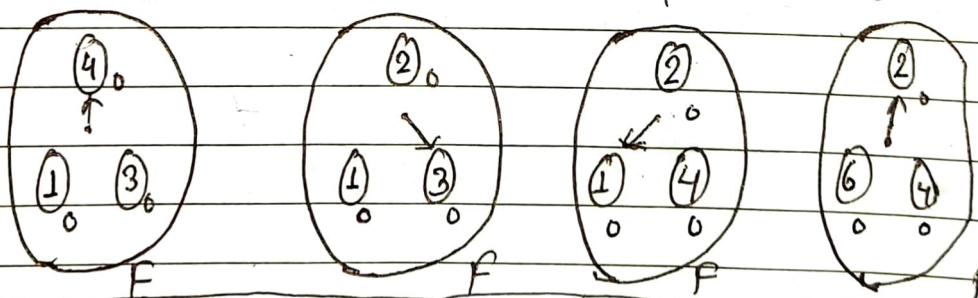
Upon accessing 1 | Upon accessing 3 | Upon accessing 4 | Access 2



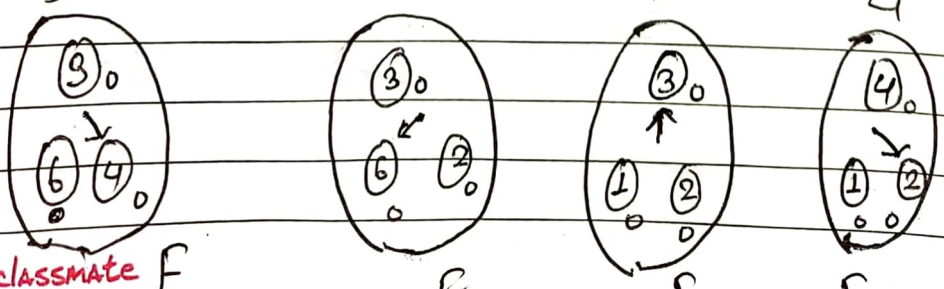
3 | 5 | 4 | 3



1 | 2 | 4 | 6

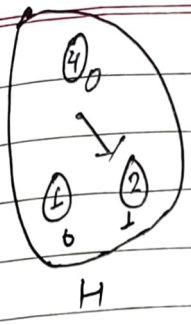


3 | 2 | 1 | 4



PAGE

DATE



Hence, Total page faults = 14

Q.3 How unsafe state differs from deadlocked state? Consider following initial state & identify whether requested resource is granted or denies for the given case.

Process	Has	Max	What will happen if process D request 1 resource?
A	2	6	
B	1	5	
C	2	3	
D	3	8	What if process A request 1 resource?
	Free: 2		

=) If a system goes to an unsafe state, then it may or may not go to a deadlock state. So, unsafe & deadlock state are different. Unsafe state implies that, some unfortunate sequence of events might lead to deadlock. But, it is not guided sure that system will go to deadlock. Thus, Banker's algorithm is used to see if granting each resource requested leads to a safe state or not. If it leads, request is granted, otherwise postponed.

DATE

=> First, let's make the column for need resource.

Process	Need = (Max - has)
A	4
B	4
C	1
D	5

Now, let's check the safe sequence using Banker's algorithm

i) For process A: Need 4 Available 2

Since Need > Available; A is not executed

ii) For process B Need 4 Available 2

B is not executed since Need > Available

iii) For process C Need = 1 ≤ Available = 2

So, C is executed.

∴ New available = 2 + 2 = 4

iv) For process D Need = 5 > Available = 4

So, D is not executed

v) For process A: Need = 4 = Available = 4

So, A is executed. New available = 4 + 2 = 6

vi) For process B: Need = 4 < Available = 6

So, B is executed. New available = 6 + 1 = 7

DATE

vii) For process D : Need = 5 < Available = 7
So, D is executed
New available = 7 + 3 = 10

Since, it is in safe state as all the process are executed, so, the safe sequence is <CABD>
Hence, all the requested resource is granted

=> If D requests 1 resource.

New available = 2 - 1 = 1
New need for D = 5 - 1 = 4
New allocation for D = 3 + 1 = 4

Process	Has	Need
A	2	4
B	1	4
C	2	1
D	4	4

Available = 1 or Free

Now, again let's check safe sequence

① For process A, Need = 4 > Available = 1
A isn't executed

② Process B, Need = 4 > Avail = 1
B isn't executed

③ Process C, Need > 1 = Avail = 1
C is executed
New Available = 1 + 2 = 3

④ Process D, Need = 4 > Avail = 3
D isn't executed
Similarly, others wont execute.
Since, all of the process are not granted the request for resource. So, when D request 1, system goes to deadlock.

classmate

PAGE

DATE

⇒ If A requests 1 resource.

Note: These values are changed to original table.

New available = 2 - 1 = 1

New need for A = 4 - 1 = 3

New allocation for A = 2 + 1 = 3

Process	Allocated	Need	
A	3	3	
B	1	4	
C	2	1	Available
D	3	5	

Let's check safe sequence.

(1) For process A, Need = 3 > Available = 1, so A isn't executed.

(2) For process B, Need = 4 > Available = 1, so B isn't executed.

(3) For process C, need = 1 = Available = 1 so, C is executed & New available = 1 + 2 = 3

(4) For process D, Need = 5 > Available = 3, so D isn't executed

(5) For process A, (Need = 3) = (Available = 3) so, A is executed

New available = 3 + 3 = 6

(6) Process B, Need = 4 < Available = 6 So, B is executed.

New available = 6 + 1 = 7

(7) Process D, Need = 5 < Avai. = 7 So, D is executed

New avail = 7 + 3 = 10

classmate

PAGE

DATE

Since, all the requested resources are granted. So, the system is in safe state & the safe sequence is

< A B D >

Section-B

Q.4. What is system call? Discuss process of handling system calls briefly.

=> System call acts as the interface between a process and an operating system. System calls are usually made when a process in user mode requires access to resources. Then it requests the kernel to provide the resource through system call.

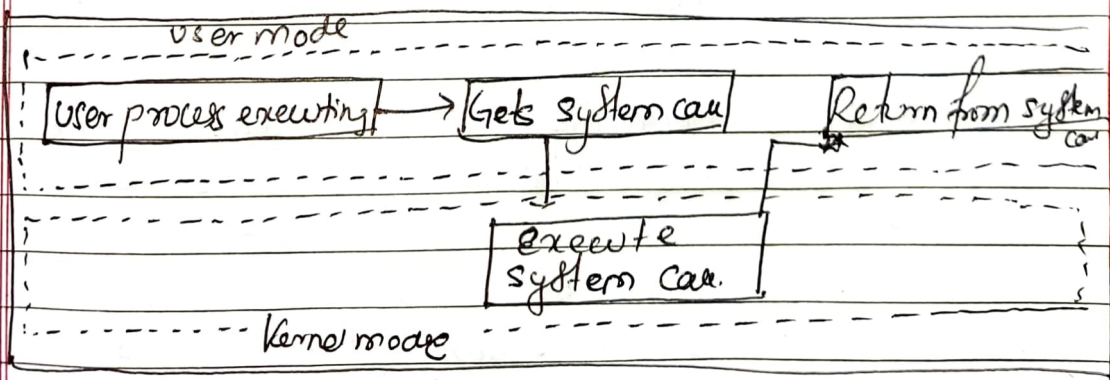


fig: Execution of system call

DATE

The steps in handling system calls are

Step-1:- The process executed in user mode till time a system call interrupts it.

Step-2:- After that, the system call is executed in the kernel mode on a priority basis.

Step-3:- Once system call execution is over, control returns to the user mode.

Step-4:- The execution of user processes resumed in kernel mode.

Q.5. What is lock variable? Discuss its working and problems associated with it in detail.

=> Lock variable is one of the software solutions for mutual exclusion. It implements mutual exclusion with busy waiting.

In this mechanism, a lock variable lock is used. When the process wants to enter its critical region it first tests the lock. If the lock is 0, the processor sets it to 1 and enters the critical region. If the lock is already 1, the

DATE

process just waits until it becomes 0. Thus, a 0 means that no process is in its critical region, and 1 means that there is some process in its critical region.

Let's say we have two processes P1 & P2. P1 has lock variable 0 so it goes to its critical section & hence P1 changes its value from 0 to 1. Meanwhile, P1 is preempted by cpu and P2 gets scheduled. Now there is no other process in the critical section & value of lock variable is 0. Then, P2 also enters into the critical section by setting lock variable to 1. Now, cpu changes P1's state from waiting to running. P1 is yet to finish its critical section & it has already checked the value of lock variable & remembers it was 0 when it previously checked it. Hence, it also enters into the critical section without checking the updated value of lock var.

Now, we get two processes in the mutual exclusion which violates the rule of mutual exclusion. Hence, the problem with lock variable is that, it doesn't guarantee the mutual exclusion.

DATE

Q.6. Differentiate between internal & external fragmentation. Suppose that we have memory of 1000 KB with 5 partitions of size 150 KB, 200 KB, 250 KB, 100 KB and 300 KB. Where the processes A & B of size 175 KB & 125 KB will be loaded, if we used Best-fit, and worst-fit strategy?

=> As the processes are loaded & removed from the main memory the free memory space is broken into little pieces. It happens after sometime that process can't be allocated to memory block considering their small size & memory blocks remain unused, this problem is known as fragmentation. There are two types of fragmentation

- Internal:- Memory block assigned to a process is bigger. Some portion of memory is left unused, as it can't be used by another process
- External:- Total memory space is enough to satisfy a request or to reside a process in it, but it isn't contiguous so it can't be used

Given,
Memory = 1000 KB with 5 partitions of 150 KB, 200 KB, 250 KB, 100 KB & 300 KB
& processes A & B of size 175 KB & 125 KB need to be loaded.

For Best-fit Strategy

1) Process A of 175 KB is loaded in 200 KB partition
2) Process B of 125 KB is loaded in 150 KB partition.

classmate

PAGE

DATE

For worse-fit strategy

1) Process A of 175 KB is loaded in 300 KB partition
2) Process B of 125 KB is loaded in 250 KB partition

Q.7. What is meant by file attributes? Discuss any one technique of implementing directories in detail.

=> File attributes are the type of meta-data that describe or modify how files and/or directories in a filesystem behave. They grant or deny certain rights to how a user or OS can access that file. Some of the file attributes of a file are:
Name, type, location, size, read-only, system, etc.

We implement the directories by two methods:
Linear list and hash table.
Let's describe about the linear list

Linear List

The simplest method of implementing a directory is to use a linear list of file names with pointers to the data blocks. This method is simple to program but time consuming to execute.

To create a new file, we add a new entry at the end of the directory. To delete a file, we search the directory for the named file, then release the space allocated to it. The real disadvantage of a linear list of directory entries is that finding a file requires a linear search. A sorted list allows a binary search and decreases the average search time.

Note: When a new file is created, then the entire list is checked whether the file name is existing already or not. And, the list needs to be traversed in case of every operation (creation, deletion, updating) on files, so systems become inefficient.

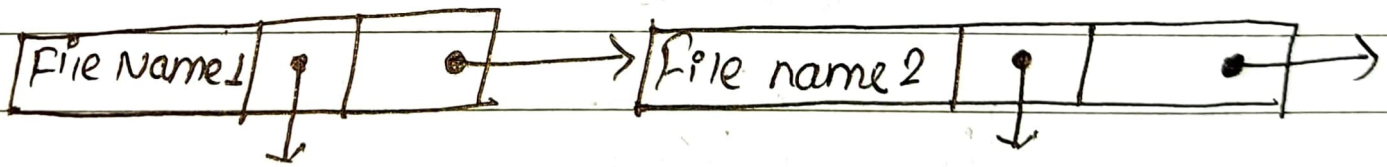


Fig: Linear List.

DATE

Q.8 Why the concept of disk interleaving is important? Explain with suitable example

⇒ Interleaving is a technique used to improve slow system performance by putting data accessed sequentially into non-sequential blocks, typically sectors.

It basically divides memory into small chunks. It is used as a high level technique to solve memory issues for motherboards and chips. By increasing bandwidth so data can access chunks of memory, the overall performance of processor & system increase. It's main purpose is to make the disk drive more efficient. That's why it is important.

Example of Interleaving:- In earlier days when computers weren't quick enough to read the continuous streams of data, interleaving was done. Without interleaving, there would be no gap between the sectors & data would arrive immediately before reading unit is ready. Which causes complete rotation of disk would be required again to read the same data

PAGE

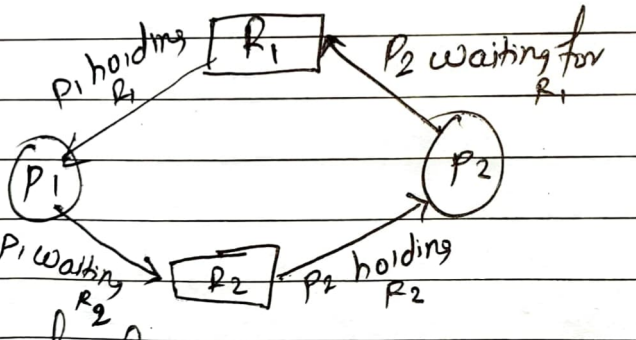
classmate

DATE

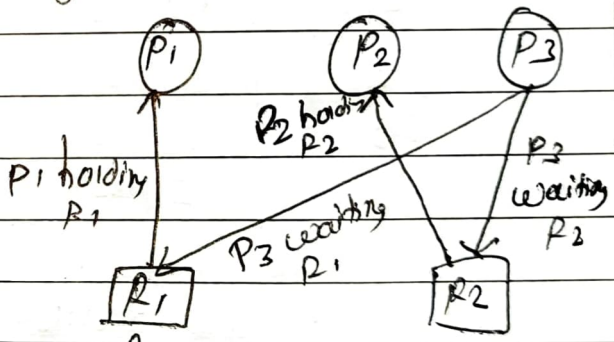
Q9 What is resource allocation graph? Explain the process of detecting deadlocks when there is single instance of each resource with suitable example.

:) Resource allocation graph is a directed graph that briefs about the deadlock more precisely. This is the principal representation showing process requested resources and the assigned resources In RAG, vertices and edges are present. Vertices are of two types: Process vertex and resource vertex and resource are of two types: Single instance and multiple instance.

Let's take an example of Single instance RAG.



(1) fig: Single Instance resource type with deadlock



(2) fig. Single Instance resource type without deadlock

Here, we can see in fig(1) If there is a cycle in RAG and each resource in the cycle provides only one instance, then the process will be in deadlock.
In fig(2), there is no deadlock because there is no circular dependency. So cycle in single-instance resource type is sufficient condition for deadlock

classmate

PAGE

DATE

Q.10. Discuss the concept of SJF and SRTN scheduling algorithms with suitable example.

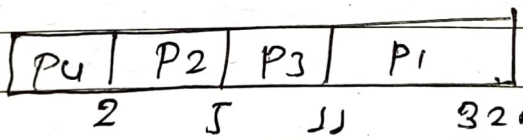
⇒ **Shortest-Job First (SJF)**

- It is a non-preemptive scheduling algorithm.
- It is the best approach to minimize waiting time.
- It selects the waiting process with smaller execution time to execute next. This is the advantage as short process are handled very quickly.
- It is not applicable on timesharing system.

Ex:

Process	Burst time
P_1	21 ms
P_2	3 ms
P_3	6 ms
P_4	2 ms

⇒ Let's make Gantt Chart



∴ Average waiting time for
$P_4 = 0$, $P_2 = 2$, $P_3 = 5$, $P_1 = 11$

Average waiting time for a process
$= \frac{0+2+5+11}{4} = 4.5$ ms

DATE

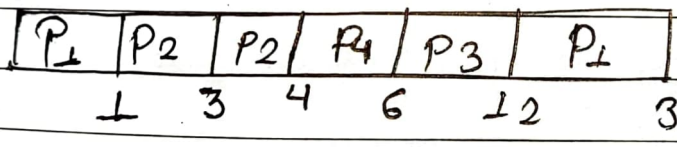
Shortest Remaining Time next (SRTN)

- This is the preemptive Scheduling algorithm
- Here, process with the smallest amount of time remaining until completion is selected to execute.
- It is useful in time sharing system
- It is little overhead than SJF.

Ex:

Process	Arrival time	Burst time (ms)
P_1	0	21
P_2	1	3
P_3	2	6
P_4	3	2

Let's make Grantt Chart



Process	Arrival time	Burst Time	Completion time	Turn around time	Waiting time
P_1	0	21	32	32	11
P_2	1	3	4	3	0
P_3	2	6	12	10	4
P_4	3	2	6	3	1

$\therefore$ Average waiting time $= \frac{11+0+4+1}{4}$

$= 4$ ms.

classmate

PAGE

DATE

Q. 11. What approaches are used for managing free disk space? Explain linked list approach with example.

=> As the disk space is limited, we need to reuse the space from deleted files for new files, if possible. To keep track of free space, the system maintains a free space list. The free space list records all free disk blocks those not allocated to some file or directory. There are mainly two approaches by which the free blocks in the disk are managed:

- Bitmap free space management
- Linked list free space management

Linked list free space management

It is one approach for free space management. This approach suggests linking together all the free blocks and keeping a pointer in cache which points to the first free block. So, all the free blocks on the disk will be linked together with a pointer.

When a block gets allocated, its previous free block will be linked together to its next free block. We would keep a pointer to block 2, as the first free block. Block 2 would contain a pointer to block 3, which would point to block 4.

classmate

PAGE

DATE

which would point to block 5 & so on. However, this scheme is not efficient; to traverse the list, we must read each block, which requires a lot of time

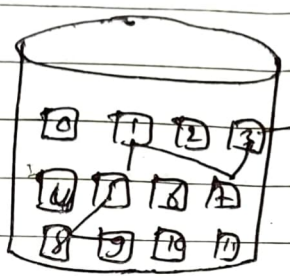


fig: Linked list free space in the disk.

Q.10. Write short notes on:

- IPC in Linux.

IPC (Interprocess Communication) refers to a mechanism in linux where the OS allows various process to communicate with each other. This involves synchronizing their actions & managing shared data. For IPC, linux has three main components:

→ Module Management: Defines all the required Communication interfaces for newly inserted module.
→ Driver registration: Registers the drivers which is maintained in a registration table of the module
→ Conflict resolution: Resolves conflict among large range of drivers.

DATE

- Disk access.

In modern computers, most of the secondary storage is in the form of magnetic disks. Hence, knowing the structure of a magnetic disk is necessary to understand how data in the disk is accessed by the computer.

Disk access time: Rotational latency + seek time + Transfer time.

Where

Seek time: Time taken in locating the disk arm to be a specified track where the read/write request will be satisfied.

Rotational latency: Time taken by the desired sector of disk to rotate into a position

Transfer time: Time to transfer the data.