Model Answer of Operating System (22516)

Summer-2024

Q1. Attempt any FIVE of the following

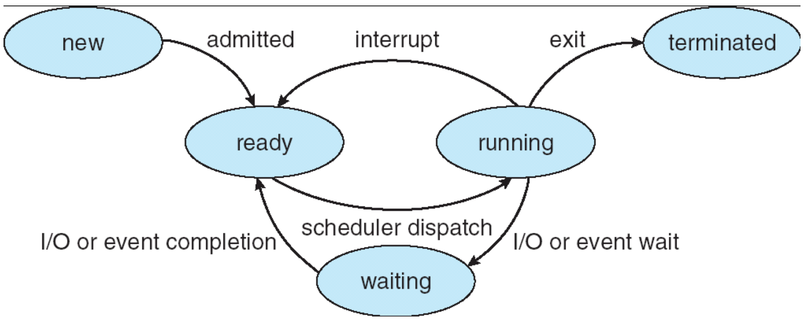
a. List different types of operating system

1. Batch Operating System
2. Multiprogrammed Operating System
3. Multiprocessor Operating System
4. Distributed Operating system
5. Time shared Operating System
6. Real Time Operating System
7. Mobile (Android iOS) Operating System

b. State any four services provided by an operating system

1. Program execution
2. I/O operations
3. File System manipulation
4. Communication
5. Error Detection
6. Resource Allocation
7. Protection

c. Draw process state diagram



d. State two features of non preemptive scheduling

1. Non-preemptive Scheduling is used when a process terminates, or a process switches from running to the waiting state.
2. In this scheduling, once the resources (CPU cycles) are allocated to a process, the process holds the CPU till it gets terminated or reaches a waiting state

e. Define following terms i) Memory compaction ii) Fragmentation

Memory compaction : Compaction is the process in which the free spaces are collected in the large memory chunk to make some space available for processes.
Fragmentation : Fragmentation refers to an unwanted problem that occurs in the OS in which a process is unloaded and loaded from memory, and the free memory space gets fragmented. The processes can not be assigned to the memory blocks because of their small size.

f. Write syntax of "pwd" command and explain its use with the help of suitable example

The 'pwd' (print working directory) command is a built-in command that displays the full pathname of the current directory. It can be used with the simple syntax,

```
$pwd [optional_arguments]
```

g. List any four file operations

Creating a file : Two steps are necessary to create a file; 1. Space in file system
2. An entry for the new file must be made
Writing a file : To write a file, we make a system call specifying both the name of the file and the information to be written to the file
Reading a file : To read from a file, we use a system call that specifies the name of the file and where the next block of the file should be put
Repositioning within a file : The directory is searched for the appropriate entry and the current file position is set to a given value
Deleting a file : To delete a file, we search the directory for the named file

Q2 Attempt any THREE of the following

a) Explain resource management of an operating system

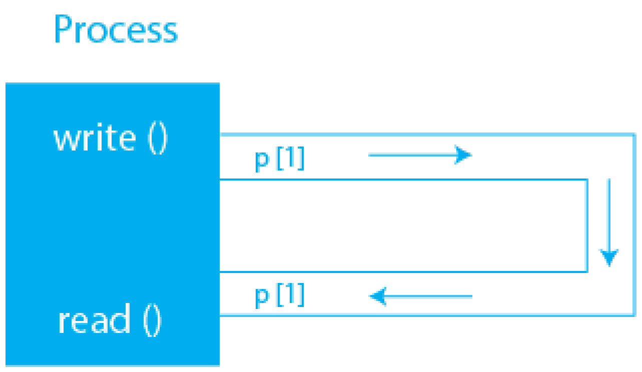
In case of multi-user or multi-tasking environments, resources such as main memory, CPU cycles and files storage are to be allocated to each user or job. Following are the major activities of an operating system with respect to resource management −
The OS manages all kinds of resources using schedulers.
CPU scheduling algorithms are used for better utilization of CPU.

b) Explain different components of operating system

1. Process Management
2. Main Memory Management
3. File Management
4. Secondary Management
5. I/O System Management

c) Describe message passing system of interprocess communication (IPC)

Message passing is a method of Inter Process Communication in OS. It involves the exchange of messages between processes, where each process sends and receives messages to coordinate its activities and exchange data with other processes.



d) What is a CPU scheduler? Explain the preemptive and non preemptive type of scheduling

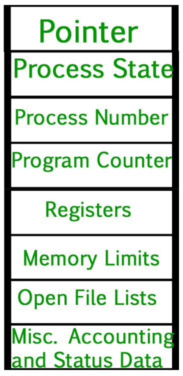
CPU scheduling is a process which allows one process to use the CPU while the execution of another process is on hold(in waiting state) due to unavailability of any resource like I/O etc, thereby making full use of CPU. The aim of CPU scheduling is to make the system efficient, fast and fair.

There are two main types of CPU scheduling, preemptive and non-preemptive. Preemptive scheduling is when a process transitions from a running state to a ready state or from a waiting state to a ready state. Non-preemptive scheduling is employed when a process terminates or transitions from running to waiting state.

Q3 Attempt any THREE of the following

a. Define process. Draw a process Control Block and explain the information in PCB.

A process is defined as a sequence of instructions executed in a predefined order. In simple words, any program that is executed is termed as a process. Processes change their state as they execute and can be either new, ready, running, waiting or terminated.



Process Control Block

1. **Pointer:** It is a stack pointer that is required to be saved when the process is switched from one state to another to retain the current position of the process.
2. **Process state:** It stores the respective state of the process.
3. **Process number:** Every process is assigned a unique id known as process ID or PID which stores the process identifier.
4. **Program counter:** It stores the counter,**:** which contains the address of the next instruction that is to be executed for the process.
5. **Register:** Registers in the PCB, it is a data structure. When a process is running and it's time slice expires, the current value of process specific registers would be stored in the PCB and the process would be swapped out. When the process is scheduled to be run, the register value is read from the PCB and written to the CPU registers. This is the main purpose of the registers in the PCB.
6. **Memory limits:** This field contains the information about memory management systems used by the operating system. This may include page tables, segment tables, etc.
7. **Open files list :** This information includes the list of files opened for a process.

b. Define deadlock. State the condition necessary for deadlock.

A deadlock in OS is a situation in which more than one process is blocked because it is holding a resource and also requires some resource that is acquired by some other process.
The four necessary conditions for a deadlock situation are mutual exclusion, no preemption, hold and wait and circular set.

c. Explain the following terms with respect to memory management: i) Dynamic relocation ii) Swapping

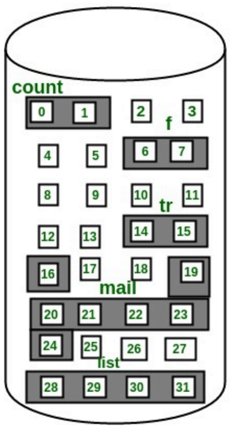
Under dynamic relocation, each program-generated address called a virtual address) also called a logical address), is translated in hardware to a physical address (also called a real address). This happens as part of each memory reference.
Swapping in an operating system involves moving data between RAM and disk to manage memory efficiently, temporarily transferring inactive memory pages to disk to free up RAM for active processes, thereby preventing system slowdowns and crashes due to insufficient memory

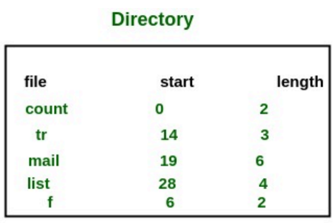
d) With a suitable diagram, explain how contiguous file allocation is performed?

In this scheme, each file occupies a contiguous set of blocks on the disk. For example, if a file requires n blocks and is given a block b as the starting location, then the blocks assigned to the file will be: *b, b+1, b+2,……b+n-1.* This means that given the starting block address and the length of the file (in terms of blocks required), we can determine the blocks occupied by the file.
The directory entry for a file with contiguous allocation contains

- Address of starting block
- Length of the allocated portion.

The *file 'mail'* in the following figure starts from block 19 with length = 6 blocks. Therefore, it occupies *19, 20, 21, 22, 23, 24* blocks.





Directory

file	start	length
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2

Q4 Attempt any THREE of the following

a) Compare between Time sharing operating system and multiprogramming operating system

Multiprogramming System	Time-Sharing System
A multiprogramming operating system allows you to run numerous processes simultaneously by monitoring their states and switching between them.	Time sharing is the logical expansion of multiprogramming. In this time-share operating system, many users/processes are assigned with IT resources in different time slots.
The problem of underutilization of the processor and memory has been remedied, and the CPU can now run numerous programs. This is why it's known as multiprogramming.	The processor's time is divided among several users. It's dubbed a time-sharing operating system for this reason.
A single processor can run the process in multiprogramming.	Two or more users can use a CPU in their terminal in this operation.
There is no set time slice in a multiprogramming OS.	A fixed time slice exists in a time-sharing OS.
The executive power in a multiprogramming OS system is not delegated before a task is completed.	Executive power is switched off in a time-sharing OS system before execution is completed.
The system does not waste time working on many processes here.	Each process takes the same amount of time or less time here.
The system relies on devices to switch between activities in Multiprogramming OS, such as I/O interrupts.	The operating system uses the time to transition between processes in time-sharing.
Multiple programs are the system model of a multiprogramming system.	Multiple programs and users are part of the time-sharing system's system model.
The multiprogramming system maximizes Response time.	The system of time-sharing maximizes Response time.

Mac OS, Windows OS, and microcomputers like MP/M, XENIX, and ESQview are just a few examples.	Windows NT server, Unix, Linux, Multics, TOPS-10, and TOPS-20 are some examples.

b) Explain any four types of system calls

Process Control : These system calls deal with processes such as process creation, process termination etc.

File Management : These system calls are responsible for file manipulation such as creating a file, reading a file, writing into a file etc.

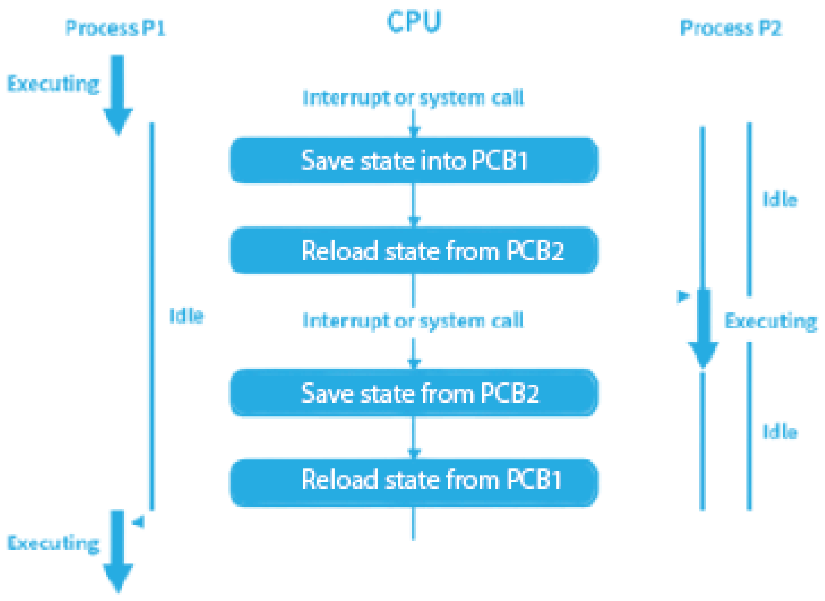
Device Management : These system calls are responsible for device manipulation such as reading from device buffers, writing into device buffers etc.

Information Maintenance : These system calls handle information and its transfer between the operating system and the user program.

Communication :These system calls are useful for interprocess communication. They also deal with creating and deleting a communication connection.

c) Describe how the context switch is executed by the operating system.

Context switching in an operating system involves saving the context or state of a running process so that it can be restored later, and then loading the context or state of another. process and run it.
Context Switching refers to the process/method used by the system to change the process from one state to another using the CPUs present in the system to perform its job.



The steps that occur during switching between processes P1 and P2 are as follows:

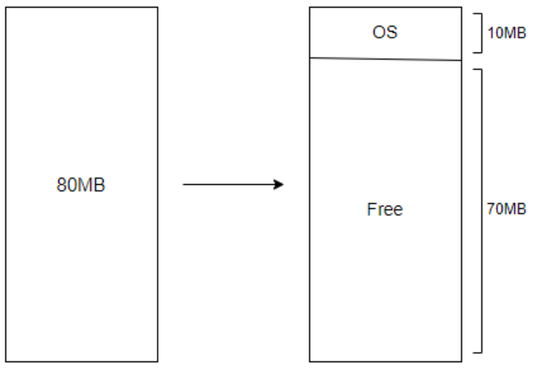
- **Step – 1** The data in the register and program counter will be saved in the PCB of process P1, let's call it PCB1, and the state in PCB1 will be changed.
- **Step – 2** Process P1 will be moved to the appropriate queue, which could be ready, I/O, or waiting.
- **Step – 3** The next process, say P2, will be chosen from the ready queue.
- **Step – 4** The process P2's state will be changed to running, and if P2 was previously executed by the CPU, it will restart execution from where it was put on hold.
- **Step – 5** If we need to execute process P1, we must complete all of the tasks stated in steps 1 to 4

d) Compare short job first (SJF) and shortest remaining time (SRTN) scheduling algorithm (any four points)

Shortest Job First:	Shortest Remaining Job First:
It is a non-preemptive algorithm.	It is a preemptive algorithm.
It involves less overheads than SRJF.	It involves more overhead than SJF.
It is slower in execution than SRJF.	It is faster in execution than SJF.
It leads to comparatively lower throughput.	It leads to increased throughput as execution time is less.
It minimizes the average waiting time for each process.	It may or may not minimize the average waiting time for each process.
It may suffer from priority inversion.	It may suffer from the convoy effect.
It involves a lesser number of context switching.	It involves a higher number of context switching.
Short processes are executed first and then followed by longer processes.	Shorter processes run fast and longer processes show poor response time.

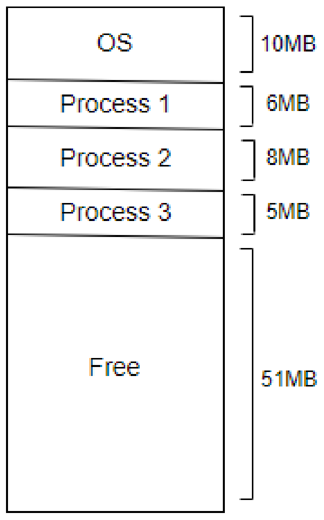
e) Describe variable partitioning with the help of suitable example

It is a part of the Contiguous allocation technique. It is used to alleviate the problem faced by Fixed Partitioning. In contrast with fixed partitioning, partitions are not made before the execution or during system configuration**.**
All memory is available for user processes and is considered as one large block of available memory, called *hole.*
Memory partitions are variable length and number.
When a process arrives & needs memory, it is allocated exactly as much memory as it requires, keeping the rest available to satisfy future requests.
In **variable-sized memory partitioning**, the main memory is divided into blocks of the same or different sizes. Variable-sized memory partitioning takes place at run-time when a process asks for a block of the main memory. If enough main memory is available, the process is assigned a block of the main memory of exactly the same size that is required.
Consider an example of a main memory of 80MB. In the beginning, the memory will contain only the operating system. Let's say that the operating system consumes 10B of the main memory; the main memory will look like this:



Let's say we bring three processes, *process 1*, *process 2*, and *process 3*, from hard disk to main memory. *Process 1*, *process 2*, and *process 3* are of sizes 6MB, 8MB, and 5MB respectively.
To assign the processes in main memory, the operating system will try to find a single block that is large enough to store the process. In other words, the operating system will find a block whose size is greater than or equal to the size of the process.
If the block is exactly the same size as the process, it will be assigned to the process. If the size of the block is greater than the size of the process, the block will be broken down into two blocks, one equal in size to the block, and the other being *size = size of the block before breaking - the size of the process*.

After breaking down, the process will be assigned the first block that was of equal size to the process. After loading the three processes into the memory, the memory will look like this:



Q5 Attempt any TWO of the following

a) Explain the use of following OS tools
i) Device manager ii) Task scheduler

Device Management in the Operating system manages all the hardware or virtual devices of a computer or PC. The device management system allocates input/output devices to the process based on priority and deallocated as well either temporarily or permanently depending upon the conditions.
The Task Scheduler enables you to automatically perform routine tasks on a chosen computer. The Task Scheduler does so by monitoring whatever criteria you choose (referred to as triggers) and then executing the tasks when those criteria are met.

b) Explain user level thread and kernel level thread with its advantages and disadvantages

The user-level threads are implemented by users and the kernel is not aware of the existence of these threads. It handles them as if they were single-threaded processes. User-level threads are small and much faster than kernel level threads. They are represented by a program counter(PC), stack, registers and a small process control block. Also, there is no kernel involvement in synchronization for user-level threads.

Advantages of User-Level Threads

- User-level threads are easier and faster to create than kernel-level threads.
- They can also be more easily managed.
- User-level threads can be run on any operating system.
- There are no kernel mode privileges required for thread switching in user-level threads.

Disadvantages of User-Level Threads

- Multithreaded applications in user-level threads cannot use multiprocessing to their advantage.
- The entire process is blocked if one user-level thread performs blocking operation.

Kernel-level threads are handled by the operating system directly and the thread management is done by the kernel. The context information for the process as well as the process threads is all managed by the kernel. Because of this, kernel-level threads are slower than user-level threads.

Advantages of Kernel-Level Threads

- Multiple threads of the same process can be scheduled on different processors in kernel-level threads.
- The kernel routines can also be multithreaded.
- If a kernel-level thread is blocked, another thread of the same process can be scheduled by the kernel.

Disadvantages of Kernel-Level Threads

- A mode switch to kernel mode is required to transfer control from one thread to another in a process.
- Kernel-level threads are slower to create as well as manage as compared to user-level threads.

c) Consider the string:

0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3, 4, 5, 6, 7 with frame size 3 and 4, calculate page fault in both the cases using FIFO algorithm.

Frame	**0**	**1**	**2**	**3**	**0**	**1**	**2**	**3**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
F1	0	0	0	**3**	3	3	**2**	2	2	**1**	1	1	**4**	4	4	**7**
F2		1	1	1	**0**	0	0	**3**	3	3	**2**	2	2	**5**	5	5
F3			2	2	2	**1**	1	1	**0**	0	0	**3**	3	3	**6**	6
	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F	F

16 Page faults

Frame	**0**	**1**	**2**	**3**	**0**	**1**	**2**	**3**	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
F1	0	0	0	0	0	0	0	0	0	0	0	**3**	3	3	3	**7**
F2		1	1	1	1	1	1	1	1	1	1	1	**4**	4	4	4
F3			2	2	2	2	2	2	2	2	2	2	2	**5**	5	5
F4				3	3	3	3	3	3	3	3	3	3	3	**6**	6
	F	F	F	F	N	N	N	N	N	N	N	F	F	F	F	F

9 Page faults

Q6 Attempt any TWO of the following

a) What is the average turnaround time for the following process using:
i) FCFS scheduling algorithm
ii) SJF non-preemptive scheduling algorithm
iii) Round Robin scheduling algorithm

Process	**Arrival Time**	**Burst Time**
P1	0	8
P2	1	4
P3	2	1

FCFS scheduling :

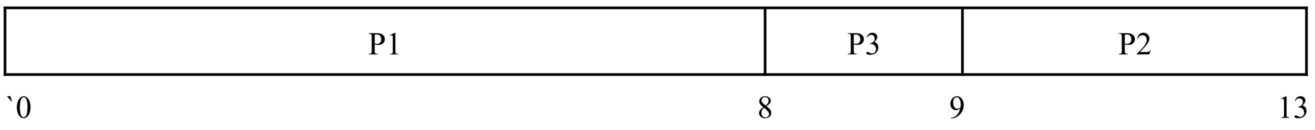
P1	P2	P3

`0 8 12 13

Process	Arrival Time	Burst Time	Completion Time	Turnaround Time CT-AT	Waiting Time (TA-BT)
P1	0	8	8	8	0
P2	1	4	12	11	7
P3	2	1	13	11	10

Average Turnaround Time is : (8+11+11)/3 = 10

SJF non-preemptive scheduling :

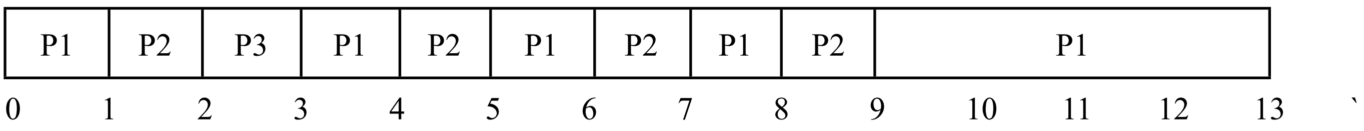


Process	Arrival Time	Burst Time	Completion Time	Turnaround Time CT-AT	Waiting Time (TA-BT)
P1	0	8	8	8	0
P2	1	4	13	12	8
P3	2	1	9	7	6

Average Turnaround Time is : (8+12+7)/3 = 9

Round Robin scheduling :

As Time Slice is not given in the problem, let us consider it as 1



Process	Arrival Time	Burst Time	Completion Time	Turnaround Time CT-AT	Waiting Time (TA-BT)
P1	0	8	13	13	5
P2	1	4	9	8	4
P3	2	1	3	1	0

Average Turnaround Time is : (13+8+1)/3 = 7.33

b) Explain bitmap vector and linked list free space management techniques with its advantages and disadvantages

Bitmap or Bit vector – A Bitmap or Bit Vector is a series or collection of bits where each bit corresponds to a disk block. The bit can take two values: 0 and 1: *0 indicates that the block is allocated* and 1 indicates a free block. The given instance of disk blocks on the disk in *Figure 1* (where green blocks are allocated) can be represented by a bitmap of 16 bits as: 0000111000000110.

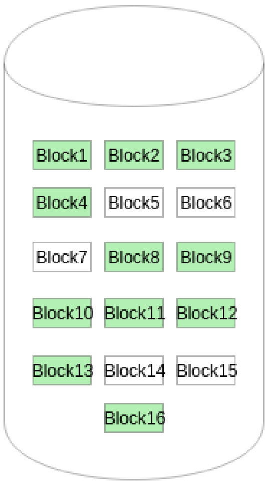


Figure - 1

Advantage :

- This technique is relatively simple.
- This technique is very efficient to find the free space on the disk.
- Easy to find free blocks.

Disadvantage :

- It may not be feasible to keep the bitmap in memory for large disks.

Linked List – In this approach, the free disk blocks are linked together i.e. a free block contains a pointer to the next free block. The block number of the very first disk block is stored at a separate location on disk and is also cached in memory.

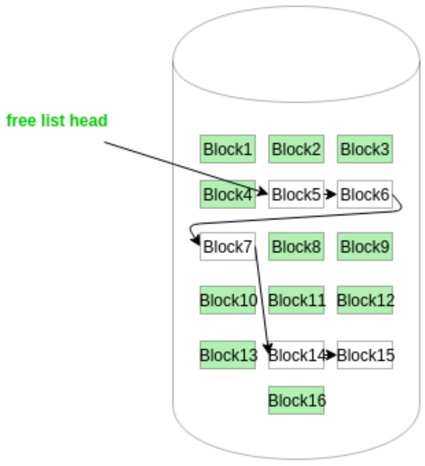


Figure - 2

In *Figure-2*, the free space list head points to Block 5 which points to Block 6, the next free block and so on. The last free block would contain a null pointer indicating the end of the free list. A drawback of this method is the I/O required for free space list traversal.

Advantages

- Whenever a file is to be allocated a free block, the operating system can simply allocate the first block in the free space list and move the head pointer to the next free block in the list.

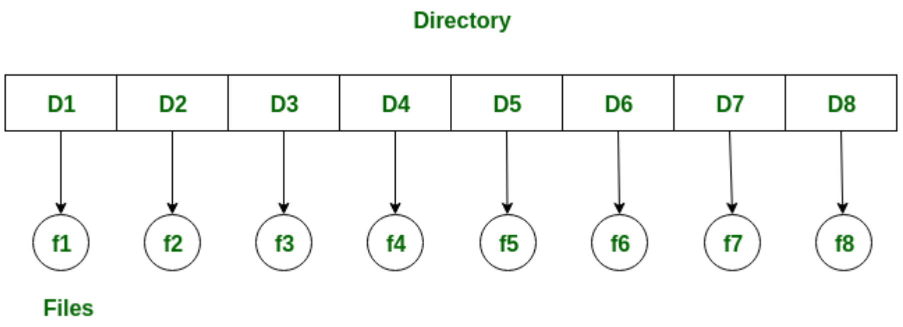
Disadvantages

- Searching the free space list will be very time consuming; each block will have to be read from the disk, which is read very slowly as compared to the main memory.
- Not Efficient for faster access.

c) Explain with diagram single level directory structure and two level directory structure with its advantages and disadvantages

Single-level directory:
The single-level directory is the simplest directory structure. In it, all files are contained in the same directory which makes it easy to support and understand.

A single level directory has a significant limitation, however, when the number of files increases or when the system has more than one user. Since all the files are in the same directory, they must have a unique name. If two users call their dataset test, then the unique name rule is violated.



Advantages:

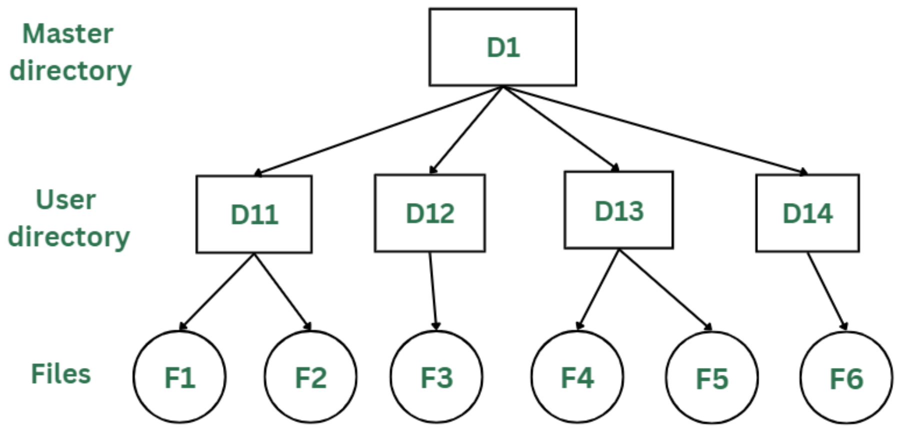
- Since it is a single directory, its implementation is very easy.
- If the files are smaller in size, searching will become faster.
- The operations like file creation, searching, deletion, updating are very easy in such a directory structure.
- Logical Organization: Directory structures help to logically organize files and directories in a hierarchical structure. This provides an easy way to navigate and manage files, making it easier for users to access the data they need.
- Increased Efficiency: Directory structures can increase the efficiency of the file system by reducing the time required to search for files. This is because directory structures are optimized for fast file access, allowing users to quickly locate the file they need.
- Improved Security: Directory structures can provide better security for files by allowing access to be restricted at the directory level. This helps to prevent unauthorized access to sensitive data and ensures that important files are protected.
- Facilitates Backup and Recovery: Directory structures make it easier to backup and recover files in the event of a system failure or data loss. By storing related files in the same directory, it is easier to locate and backup all the files that need to be protected.
- Scalability: Directory structures are scalable, making it easy to add new directories and files as needed. This helps to accommodate growth in the system and makes it easier to manage large amounts of data.

Disadvantages:

- There may be a chance of name collision because two files can have the same name.
- Searching will become time taking if the directory is large.
- This can not group the same type of files together.

Two-level directory:
As we have seen, a single level directory often leads to confusion of files names among different users. The solution to this problem is to create a separate directory for each user.

In the two-level directory structure, each user has their own user files directory (UFD). The UFDs have similar structures, but each lists only the files of a single user. System's master file directory (MFD*)* is searched whenever a new user id is created.



Two-Levels Directory Structure

Advantages:

- The main advantage is there can be more than two files with the same name, and would be very helpful if there are multiple users.
- Security would be there which would prevent users from accessing other user's files.
- Searching for the files becomes very easy in this directory structure.

Disadvantages:

- Users cannot share the file with the other users.
- Unlike the advantage users can create their own files, users don't have the ability to create subdirectories.
- Scalability is not possible because one use can't group the same types of files together.