# Chapter 5

# File Systems, I/O Systems, Protection and Security

**LEARNING OBJECTIVES**

- *File systems*
- *File management systems*
- *File system architecture*
- *Device drivers*
- *Basic input/output supervisor*
- *Logical input/output*
- *Access methods*

## File Systems

The file system consists of two distinct parts:

1. Collection of files
2. Directory structure

***File*** A file is a named collection of related information that is recorded on secondary storage. The files must have

1. Long-term existence
2. Sharable between processes
3. Structure

***File attributes*** A typical file attributes are

1. Name
2. Identifier
3. Type
4. Location
5. Size
6. Protection
7. Time, date and user identification

***File operations*** The operations that are applied on files are

1. Creation
2. Deletion
3. Closing
4. Reading
5. Writing

***File types*** A common technique for implementing file types is to include the type as part of the file name. The name is split into two parts:

1. A name
2. An extension (usually separated by a period character)

The type of a file may be

1. Executable (exe, com, bin)
2. Object (obj, o)
3. Source code (c, cc, java)
4. Batch (bat, sh)
5. Text (txt, doc)
6. Word processor (wp, text, doc)
7. Library (lib)
8. Print or view (ps, pdf, jpg)
9. Archive (zip, tar)
10. Multimedia (mpeg, mov, rm)

***File structure*** The four common terms of file systems are

**1.** Field
   - Basic element of data
   - Contains a single value
   - Has a particular length and data type

**2.** Record
   - It is a collection of related fields
   - It is treated as a unit

**3.** File
   - It is a collection of similar records
   - Treated as a single entity
   - Has file names
   - Access to file may be restricted or unrestricted

**4.** Database
   - Collection of related data.
   - Relationship exists among elements.

## File Management Systems

It is a set of system software that provides services to users and applications in the use of files. Objectives of file management systems are as follows:

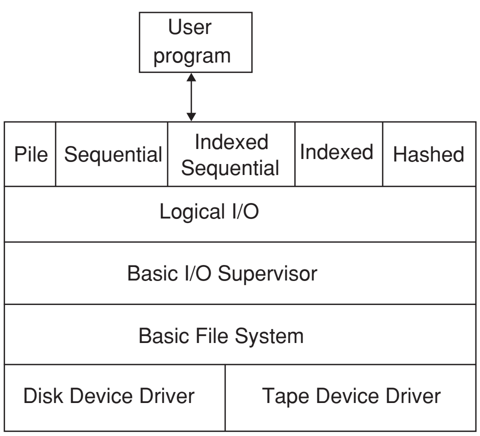
1. To meet the data management needs and requirements of the user including storage of data.
2. To guarantee, that the data in the file are valid.
3. To optimize performance (i.e., throughput, response time).
4. To provide Input/Output (I/O) support for a variety of storage device types.
5. To minimize or eliminate the potential for lost or destroyed data.
6. To provide a standardized set of I/O interface routines.
7. To provide I/O support for multiple users.

The minimal set of requirements from user's point of view for an interactive, general purpose, file system are as follows:

1. Ability to create, delete, read, write and modify files.
2. Controlled access to other users files.
3. Control the type of accesses to files.
4. Restructure the user's files.
5. Able to move data between files.
6. Ability to back up and recover user's files.
7. Able to access a file by name.

# File System Architecture

The file system architecture is shown below:




## Device Drivers

1. It is at the lower part.
2. Communicates directly with peripheral devices.
3. Initiative to start I/O operations on a device.
4. Processes the completion of an I/O request.

## Basic File System

1. Mainly concerned with I/O.
2. Exchanges blocks of data.
3. Deals with placement of blocks.
4. Deals with buffering blocks in main memory.

## Basic I/O Supervisor

1. Responsible for file initiation and termination.
2. Maintain control structures.
3. Selects the device while file I/O is to be performed.
4. Deals with scheduling access to optimize performance.
5. Part of the operating system (OS).

## Logical I/O

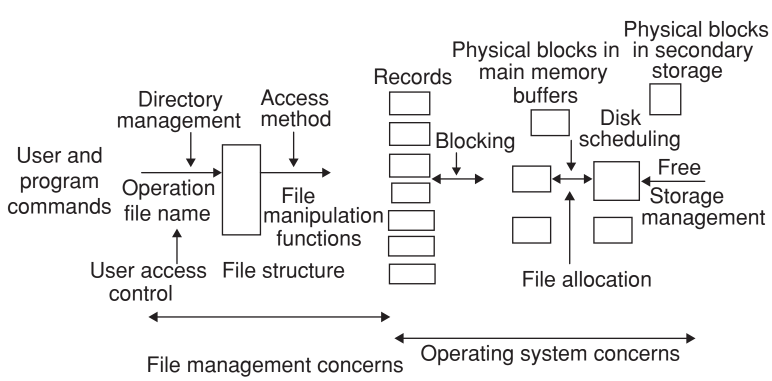
1. Enables users and application to access records.
2. Provides general purpose record I/O capability.
3. Maintains basic data about file.

## Access Methods

1. They reflect different file structures.
2. Provide different ways to access and process data.
3. Access methods are as follows:
   - Sequential: read next, write next
   - Direct: read block n, write block n

## File Management Functions

The functions of a file system is shown below:



## File Organization and Access

File organization refers to the logical structuring of the records as determined by the way in which they are accessed. We choose a particular file organization based on

1. Short access time
2. Ease of update
3. Economy of storage
4. Simple maintenance
5. Reliability

We will discuss five types of file organizations:

1. The pile
2. The sequential file
3. The indexed sequential file
4. The indexed file
5. The direct or hashed file

***The pile*** The pile organization is shown in Figure 1:

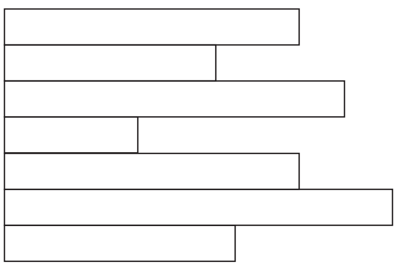


**Figure 1** Pile file.

1. Data are collected in order they arrive.
2. The aim is to accumulate mass of data and save.
3. Records may have different fields.
4. There is no structure.
5. Record are accessed by exhaustive search.
6. There can be variable length records.
7. This type of files are encountered when data are collected and stored prior to processing or when data are not easy to organize.

***The sequential file*** The sequential file is shown in Figure 2.

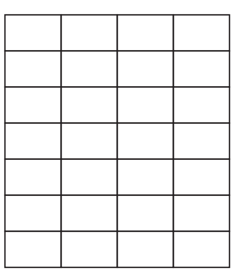


**Figure 2** Sequential file.

1. Fixed format used for records.
2. Records are of same length.
3. All fields are of the same order and length.
4. One field is the key field. It uniquely identifies the record. Records are stored in a key sequence.
5. New records are placed in a log file or transaction file.
6. Batch update is performed to merge the log file with the master file.
7. Used in batch applications.
8. Not suitable for interactive applications.

***Indexed sequential file***

1. Index provides a look up capability to quickly reach the vicinity of the desired record.
2. It contains key field and a pointer to the main file.
3. Index is searched to find highest key value that is equal to or precedes the desired key value.
4. Search continues in the main file at the location indicated by the pointer.
5. New records are added to an overflow file.
6. Record in main file that precedes it is updated to contain a pointer to the new record.
7. The overflow is merged with main file during a batch update.
8. Multiple indexes for the same key can be set up to increase efficiency.

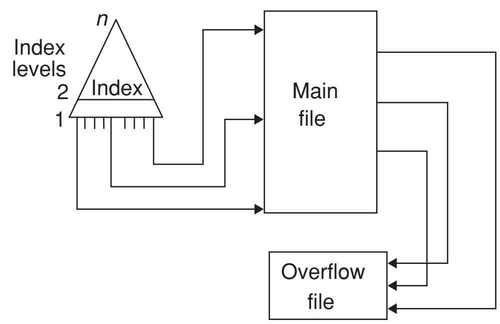
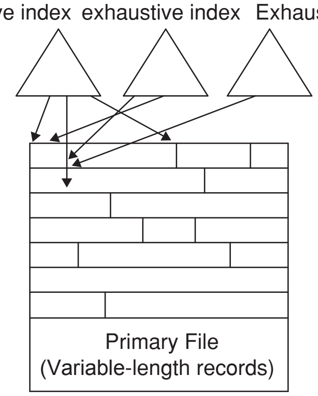



**Figure 3** Indexed file.

9. Key field required for each record.
10. It uses multiple indexes for different key fields.
11. It may contain exhaustive index that contains one entry for every record in the main file or partial index.




12. Used in the applications where timeliness of information is critical and where data are rarely processed exhaustively.

*Direct or Hashed file*

1. This file has the capability to access any block of a known address.
2. Key field required in each record.
3. No concept of sequential ordering.

Figure 4 shows hashed file organization:

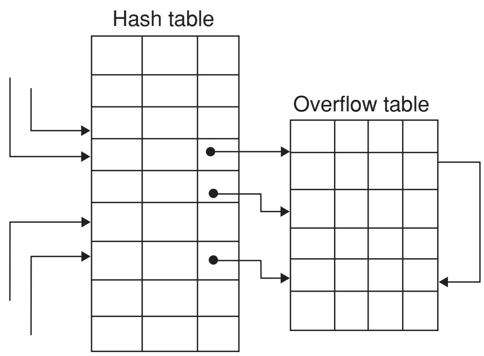



**Figure 4** Hashed file organization.

# File Directories

The collection of files is a file directory.

*Contents*

1. Contain information about files, such as attributes, location, and ownership.
2. Itself a file owned by the OS.
3. It provides mapping between file names and the file themselves.

*Directory Structure*

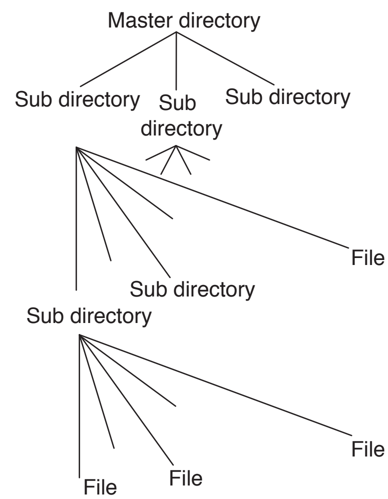
1. It consists of list of entries, one for each file.
2. Sequential file with the name of the file serving as the key.
3. Provides no help in organizing the files.
4. Not scalable (same name cannot be used for two different files).
5. As directory grows in size, searching is too time consuming.

## Two-level Directory Structure

1. Contains one directory for each user and a master directory.
2. The master directory contains entry for each user
3. User directory is a simple list of files for that user.
4. File naming conflict is solved.

## Hierarchical or Tree-structured Directory

1. Contains master directory with user directory underneath it.
2. Each user directory may have subdirectories and files as entries.




3. Files can be located by following a path from the root, or master directory down various branches which is called *pathname* for the file.
4. Current directory is called the *working directory*.
5. Files are referenced relative to the working directory.

***Naming*** The use of a tree-structured directory minimizes the difficulty in assigning unique names. Any file in the system can be located by following a path from the root or master directory down various braches until the file is reached.

***Path name*** The series of directory names, culminating in the file name itself, constitutes a path name for the file.

**Example:** Time/Gate/Exam/OS

The slash is used to delimit names in the sequence. The name of master directory is implicit, because all paths starts at that directory. Files can also be referenced from the working directory.

***File sharing*** It has got two issues, such as:

1. Access rights
2. Management of simultaneous access

***Access rights*** Users or groups of users are granted certain access rights to a file. A wide range of access rights has been used. The access rights may be

1. None
2. Knowledge
3. Execution
4. Reading
5. Appending
6. Updating
7. Changing protection
8. Deletion

These access rights can be specified to specific users or user groups or all users.

***Record Blocking*** For I/O to be performed, records must be organized as blocks. Given the size of a block, there are three methods of blocking that can be used:

1. Fixed blocking
2. Variable length spanned blocking
3. Variable length unspanned blocking.

***Fixed blocking***

1. Fixed-length records (Figure 5) are used and an integral number of records are stored in a block.
2. Possibility of internal fragmentation.
3. Used for sequential files.

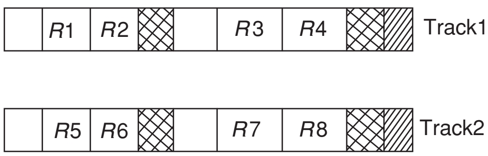



**Figure 5** Fixed blocking.

***Variable length spanned blocking*** Variable length records are used and are packed into blocks with no unused space. Some records can span two blocks. These do not limit the size of records.

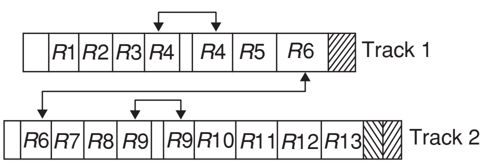



**Figure 6** Variable blocking: spanned.

***Variable length unspanned blocking*** Variable length records (Figure 7) are used, but spanning is not employed. There is wasted space in most blocks and limits record size.

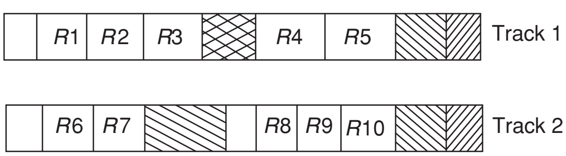
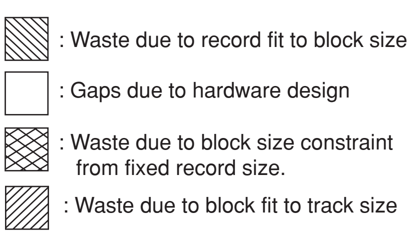



**Figure 7** Variable blocking: unspanned.




## Secondary Storage Management

1. Space (or blocks) must be allocated to files on disk.
2. Need to keep track of the space available (free blocks) for allocation to files.
3. Preallocation of blocks to files can be used to allocate space for files. For this, it needs to know the maximum size of the file at the time of creation.

## File Allocation

### Preallocation Versus Dynamic Allocation

1. A preallocation policy requires that the maximum size of a file be declared at the time of the file creation request. For many applications, it is difficult to estimate the file size.
   It is better to use dynamic allocation, which allocates space to a file in portions as needed.

***Portion size*** The portion size which is allocated to a file may be

1. Variable, large contiguous portions.
2. Blocks

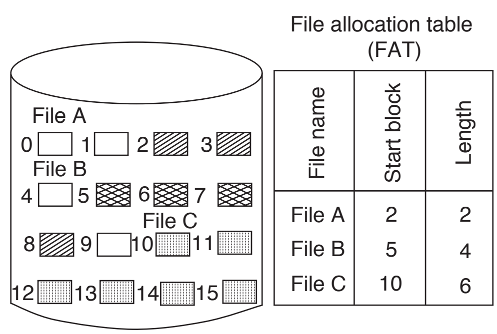
Some strategies for dealing with fragmentation of free space are as follows:

1. *First-fit*: Choose the first unused contiguous group of blocks of sufficient size.
2. *Best-fit*: Choose the smallest unused group that is of sufficient size.
3. *Nearest-fit*: Choose the unused group of sufficient size that is closest to the previous allocation.

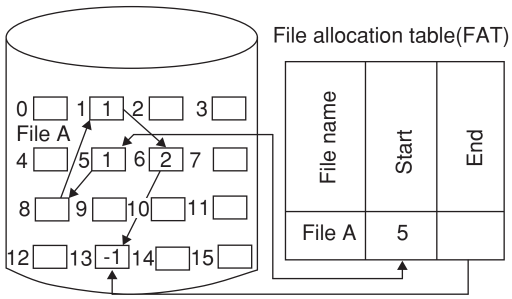
***File allocation methods*** It has three methods as follows:

1. Contiguous allocation
2. Chained allocation
3. Indexed allocation

***Contiguous allocation*** Here, a single set of blocks is allocated to a file at the time of creation. Only a single entry in the file allocation table is created consisting of starting block and length of the file. It exhibits external fragmentation and performs compaction.

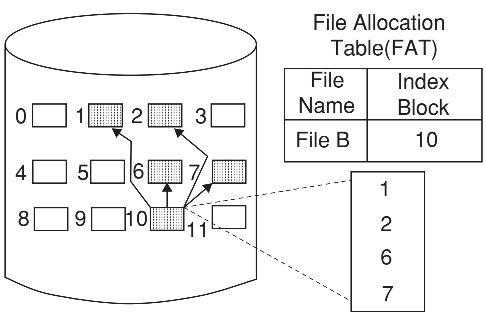



***Linked or chained allocation*** The allocation is done on basis of individual block. Each block contains a pointer to the next block in the chain. Only single entry is created in the file allocation table consisting of starting block and length of file. There occurs no external fragmentation and it is best for sequential files. There is no accommodation of principle of locality. If block size is $n$, then only $n - 1$ units of data are stored and 1 unit stores the link information.

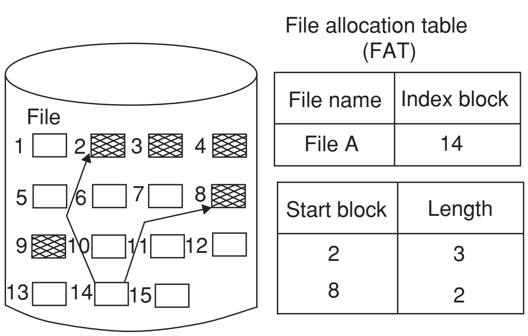



***Indexed file allocation*** The file allocation table contains a separate one level index for each file. The index has one entry for each portion allocated to the file. The file allocation table contains block number for the index. If a file requires *n* blocks, then *n* + 1 blocks are used, where the first block contains index information (pointers to data blocks).

## Index Allocation with Block Pointers




## Indexed Allocation with Variable length Portion




**Example 1:** A direct access of file has fixed size 50 byte records. Assuming the first record is record 1, the first byte of record 10 will be at what logical location?

**Solution:**
Total records = 50 × 10 = 500
First record is record 1. This record is already read.
Logical location of first byte = 500 – 50 = 450
The correct logical location = 450 + 1 = 451.

**Example 2:** A sequential access file has fixed-size 32-byte records. Assuming that the first record is record 0, the first byte of record 20 will be at what location?

**Solution:** Since the first record is record 0, the first byte of record 20 will be at logical location = 32 × 20 = 640

# Free Space Management

In addition to file allocation table, disk allocation table is also required to know what blocks on the disk are available. Some of the free space management techniques are as follows:

1. Bit tables
2. Chained free portions
3. Indexing
4. Free block list

***Bit tables*** This method uses a vector containing, one bit for each block on the disk. Each entry of a '0' corresponds to a free block and each '1' corresponds to a block in use.

### *Advantage*

1. Easy to find one or a contiguous group of free blocks.
2. Smaller in size.

The amount of memory required for a block bitmap will be

$$\frac{\text{Disk size in (bytes)}}{8 \times \text{file system block size}}$$

***Chained free portion*** The free portions may be chained together by using a pointer and length value in each free portion. This method has negligible space overhead. This method is suitable for all file allocation methods. The disk will become quite fragmented, after some use. It is slower for individual block file creation and also for deletion.

***Indexing*** It treats the free space as a file and uses an index table (same as in file allocation). The index should be on the basis of variable size portions rather than blocks.

***Free block list*** Here, each block is assigned a number sequentially and the list of the numbers of all free blocks is maintained in a reserved portion of the disk.

## Volumes

It is a collection of addressable sectors in a secondary memory that an OS or application can use for data storage. The sectors in a volume need not be consecutive on a physical storage device. (a single disk equals one volume).

# Unix File Management

## I-nodes (Index Node)

UNIX files are administered by the OS by means of i-node. An i-node (index node) is a control structure that contains the key information needed by the OS for a particular file.

The attributes of the file as well as its permissions and other control information are stored in the i-node. The exact i-node structure varies from UNIX implementation to another. The FreeBSD i-node structure is shown in Figure 8.

## File Allocation

1. It is done on a block basis.
2. Allocation is dynamic.
3. The blocks of a file on disk are not necessarily contiguous.
4. An indexed method is used to keep track of each file, with i-node includes a number of direct pointers and three indirect pointers.

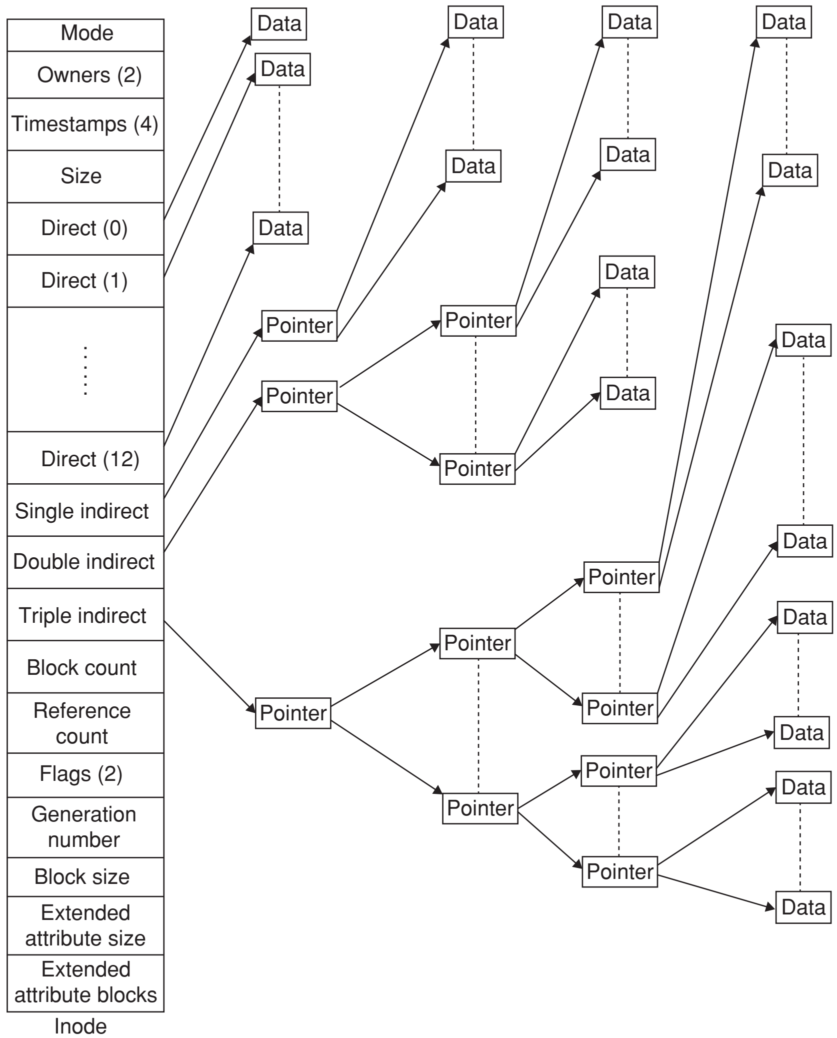



**Figure 8** Structure of free BSD i-node and file.

5. The free BSD i-node includes 120 bytes of address information that is organized as fifteen 64-bit addresses.
6. The first 12 addresses point to the first 12 data blocks of the file.
7. If the file requires more than 12 data blocks, one or more levels of indirection are used as follows:
   - The thirteenth address in the i-node points to a block on disk that contains the next portion of the index. This is referred to as the *single indirect block*.
   - If the file contains more blocks, the fourteenth address in the i-node points to a double indirect block. Each block consists of single indirect blocks, each of which contains pointers to file blocks.
   - If the file contains still more blocks, the fifteenth address in the i-node points to a triple indirect block that is a third level of indexing. This block points to additional double indirect blocks.

The capacity of FreeBSD file with 4 kB block size is shown below:

| Level | Number of Blocks | Number of Bytes |
|---|---|---|
| Direct | 12 | 48 K |
| Single indirect | 512 | 2 M |
| Double indirect | $512 \times 512 = 256$ K | 1 G |
| Triple indirect | $512 \times 256$K = 128 M | 512 G |

The total number of data blocks in a file depends on the capacity of the fixed-size blocks in the system. In FreeBSD, the minimum block size is 4 kB, and each block can hold a total of 512 block addresses. Thus, the maximum size of a file with this block size is over 500 GB.

### Windows NT File System

The windows NT file system provides a combination of reliability, compatibility and performance, which are not available in the FAT file system.

1. It will quickly perform standard file operations, such as write, read and search.
2. It also performs file-system recovery on very large hard disks.
3. NTFS file system formatting on a volume results in the creation of several system files and the master file table (MST), which contains information about all the files and folders on the NTFS (Figure 9).

| | | | |
|---|---|---|---|
| Partition boot sector | Master file table | System files | File area |

**Figure 9** NTFS after formatting.

4. It includes security features required for high-end personal computers and file servers.
5. It supports data access control and ownership privileges.
6. Folders shared on a Windows NT are assigned specific permissions.
7. It allows users to assign permissions to individual files.

## I/O Systems

### Design Objectives of OS

Two objectives of OS for I/O systems are as follows:

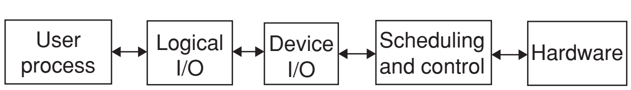
1. Efficiency
2. Generality

### Logical Structure of the I/O Function

The three most important logical structures are as follows:

1. Local peripheral devices
2. Communication ports
3. File system

### Local Peripheral Device




### Logical I/O

Manages general I/O functions on behalf of user processes, allowing them to deal with the device in terms of a device identifier and simple commands, such as open, close, read, write.

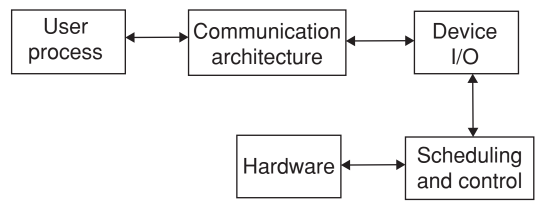
### Device I/O

The requested operations and data are converted into appropriate sequences of I/O instructions, channel commands and controller orders.

### Scheduling and Control

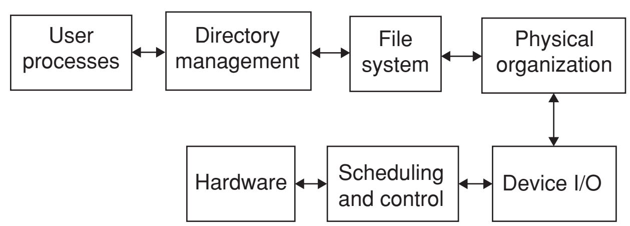
The actual queuing and scheduling of I/O operations occurs at this layer as well as the control of the operations. Interrupts are handled. I/O status is collected and reported.

### Communication Port




Here, the communication architecture may itself consist of a number of layers.

### File System




***Directory management*** Here, the symbolic file names are converted to identifiers that either reference the file directly or indirectly through a file descriptor or index table.

***File system*** This layer deals with the logical structure of files and with the operations that can be specified by users, such as open, close, read, write.

***Physical organization*** Allocation of secondary storage space and main storage buffers is generally treated here.

***I/O buffering*** In buffering, we perform input transfers in advance of requests being made and perform output transfers sometime after the request is made. We will discuss

1. Single buffering
2. Double buffering
3. Circular buffering

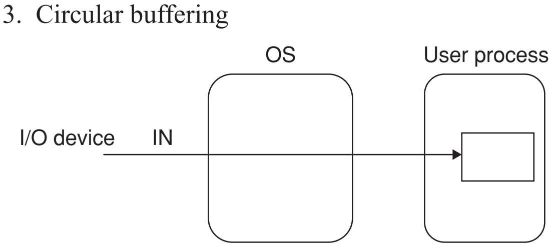



**Figure 10** No buffering.

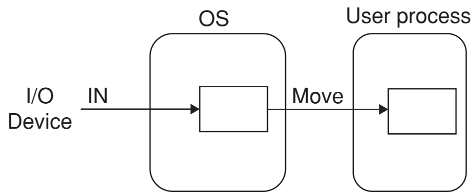



**Figure 11** Single buffering.

- When a user process issues an I/O request, the OS assigns a buffer in the system portion of main memory to the operation.
- Provides a speed up compared to the lack of system buffering.
- For block oriented devices, input transfers are made to the system buffer. When the transfer is complete, the process moves the block into user space and immediately requests another block. This is same for block-oriented output.
- Suppose that $T$ is the time required to input one block and that $C$ is the computation time that intervenes between input requests.
  Without buffering, the execution time per block = $T + C$.
  With a single buffer, the execution time per block = max $[C, T] + M$,
  where $M$ = Time required to move the data from the system buffer to user memory.
- For stream-oriented I/O, the single buffering scheme can be used in a line-get-a-time fashion or a byte-at-a-time fashion.

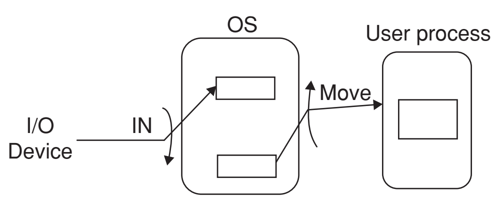



**Figure 12** Double buffer

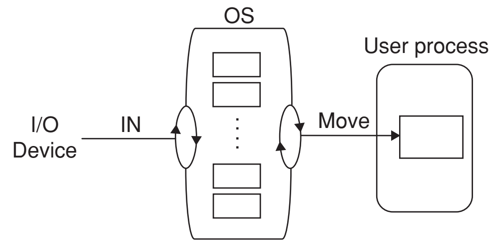
- An improvement over single buffer.
- A process now transfers data to one buffer while the OS empties the other. This is known as *double buffering* or *buffer swapping*.
- For block-oriented transfer, the estimated execution time is max $[C, T]$.
- The stream-oriented transfer may be line-at-a-time or byte-at-a-time.

## Automatic and Explicit Buffering

An indefinite length queue is provided in automatic buffering. Sender never blocks to copy a message. No specifications are present for providing automatic buffering. One method is to reserve large amount of memory. In this method, most of the memory is wasted.

The size of the queue is provided in explicit buffering. Sender will block if the requested space is more than available. In this scheme, memory wastage is less likely to happen.

## Circular Buffer




1. Here, more than two buffers are used. Each individual buffer is one unit of circular buffer.

## Kernal I/O Subsystem

Kernel I/O provides I/O-related services. It is built on the hardware and device driver infrastructure.

One of the responsibilities of Kernal I/O subsystem is to protect itself from the erroneous process and malice users.

Services provided by the I/O subsystem are as follows:

1. Scheduling
2. Buffering
3. Caching
4. Spooling
5. Device reservation
6. Error handling

# Disk Scheduling

## Hard Disk Performance Parameters (Terminologies)

***Seek time*** It is defined as the time required to move the disk arm to the required track. It consists of

1. Initial start-up time
2. The time taken to traverse the tracks that has to be crossed once the access arm is up to speed.

$$\text{Seek time, } T_s = m \times n + S$$

where $T_s$ = Estimated seek time
$n$ = Number of track traversed
$m$ = Constant that depends on the disk drive
$S$ = Start-up time

***Rotational delay*** Time required to reach the desired sector by read/write head. Rotational speed ranges from 5400 to 10,000 rpm.

1. Floppy disks typically rotate between 300 and 600 rpm.
2. For 1,00,000 rpm, the average rotational delay will be 3 ms.

***Transfer time*** The transfer time to or from the disk depends on the speed of the disk.

$$T = \frac{b}{rN}$$

where, $T$ = Transfer time
$b$ = Number of bytes to be transferred
$N$ = Number of bytes on a track
$r$ = Rotation speed in revolutions/second

Total average access time

$$T_a = T_s + \frac{1}{2r} + \frac{b}{rN}$$

$T_s \rightarrow$ average seek time

## Hard Disk Scheduling Algorithms

Disk bandwidth and fast access time are to be considered. Bandwidth is the total number of bytes transferred divided by the total time between the first request for service and completion of the last transfer.

***FCFS (First Come First Served)*** The disk in controller processes the I/O requests the order in which they arrive, thereby moving backwards and forwards across the surface of the disk to get the next requested location each time (Figure 10).

**Example 3:** A disk queue has the following requests to read tracks:

87, 170, 40, 150, 36, 72, 66, 15

Consider the disk head is initially at cylinder 60. Total head movement = (87 – 60) + (170 – 87) + (170 – 40) + (150 – 40) + (150 – 36) + (72 – 36) + (72 –66) + (66 – 15) = 27 + 83 + 130 + 110 + 114 + 36 + 6 + 51 = 557 cylinders

Average head movement $= \frac{557}{8} = 69.6$ cylinders

### Advantage

Improved response time as a request gets response in fair amount of time.

### Disadvantages

1. Involves a lot of random head movements and disk rotations.
2. Throughput is not efficient.
3. Used in small systems only where I/O efficiency is not very important.

***Shortest Seek Time First (SSTF)*** When a disk operation finishes, choose the request that is closer to the current head position or choose the request that has minimum seek time from the current head position.

**Example 4:** Consider the following requests: 87, 170, 40, 150, 36, 72, 66, 15. Find the average head movement for SSTF.

The initial head position is say 60. Now, closest to the head position is the request at cylinder 66. Then, the closest to 66 is 72, closest request to 72 is 87, and so on. Total head movements = (66 – 60) + (72 – 66) + (87 – 72) + (87 – 40) + (40 – 36) + (36 – 15) + (150 – 15) + (170 – 150) = 6 + 6 + 15 + 47 + 4 + 21 + 135 + 20 = 254 cylinders

Average head movements $= \frac{254}{8} = 31.75$ cylinders

### Advantages

1. It minimizes latency
2. Better throughput than FIFO method

### Disadvantages

1. Starvation occurs if some process has to wait for long time until its requests are satisfied.
2. SSTF services requests for those tracks which are highly localized.

***SCAN/elevator algorithm*** The disk head constantly moves from the most inner cylinder to the outer cylinder and then it changes its direction back towards the centre. As the head moves, if there is a request for the current disk position then it is satisfied.

1. It is known as *elevator algorithm* because it services all the request of going up and then reaching at the top, it goes downward.
2. It needs two information:
   - Direction of head movement
   - Last position of the disk head

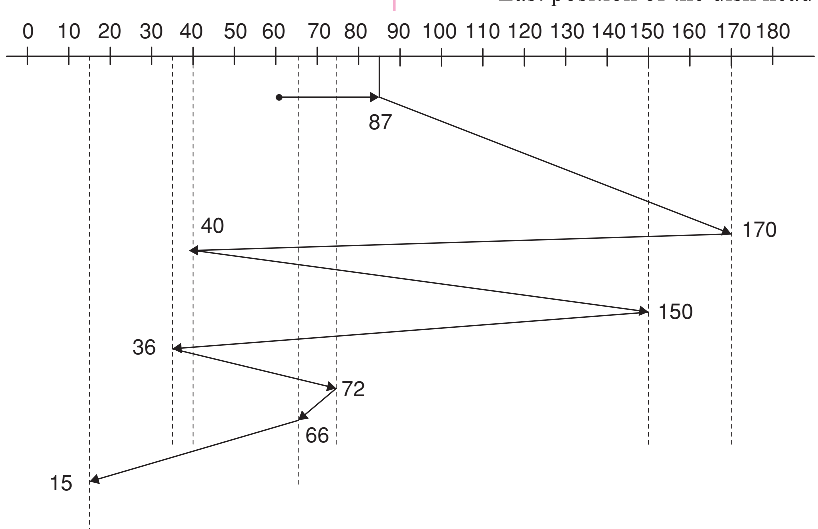



**Figure 13** FCFS.

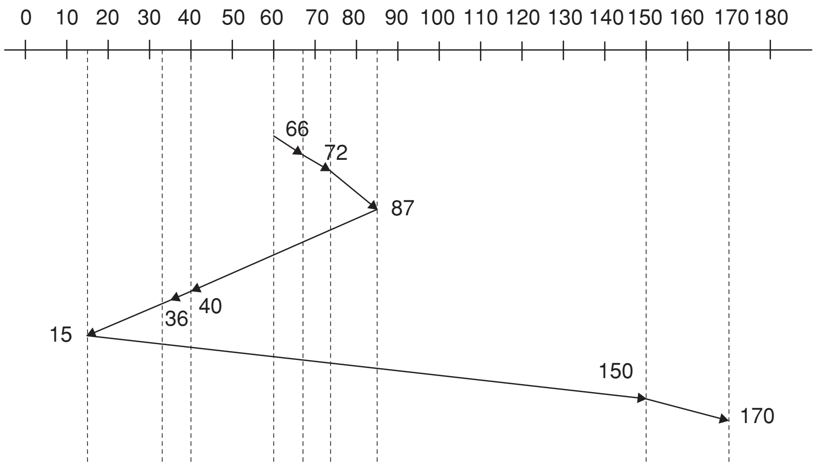



**Figure 14** SSTF

**Example 5:** For the following track requests: 87, 170, 40, 150, 36, 72, 66, 15. (Initially head is at track 60 to the arm is moving outwards.

Total head movement = (66 – 60) + (72 – 66) + (87 – 72) + (150 – 87) + (170 – 150) + (180 – 170) + (180 – 40) + (40 – 36) + (36 – 15)

= 6 + 6 + 15 + 63 + 20 + 10 + 140 + 4 + 21

= 285 cylinders

Average head movement $= \frac{285}{8} = 35.6$ cylinders

## Advantages

1. Throughput better than FIFO.
2. Basic for most scheduling algorithms.
3. Eliminates the discrimination.
4. No starvation.

## Disadvantages

1. Because of the continuous scanning of disk from end to end, the outer tracks are visited less often than the mid-range tracks.
2. Disk arm keeps scanning between two extremes; this may result in wear and tear of the disk assembly.
3. Certain requests arriving ahead of the arm position would get immediate service but some other requests that arrive behind the arm position will have to wait for the arm to return back.

***C–SCAN algorithm (one–way elevator algorithm)*** It treats the cylinder as a circular list. The head sweeps from the innermost cylinder to the outermost cylinder, satisfying the waiting requests in order of their locations. When it reaches the outermost cylinder, it sweeps back to the innermost cylinder without satisfying any requests and then starts again.

**Example 6:** Consider the cylinders requests:
87, 170, 40, 150, 36, 72, 66, 15 Starting cylinder = 60th (arm moving outwards)

Total head movement = (66 – 60) + (72 – 66) + (87 – 72) + (150 – 87) + (170 – 150) + (180 – 170) + (180 – 0) + (15 – 0) + (36 – 15) + (40 – 36)

= 6 + 6 + 15 + 63 + 20 + 10 + 180 + 15 + 21 + 4 = 340

Average head movement $= \frac{340}{8} = 35.6 = 42.5$

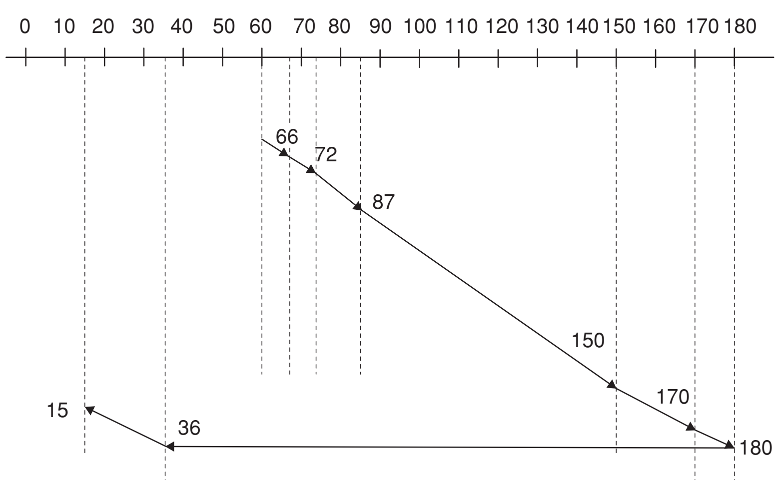



**Figure 15** SSTF

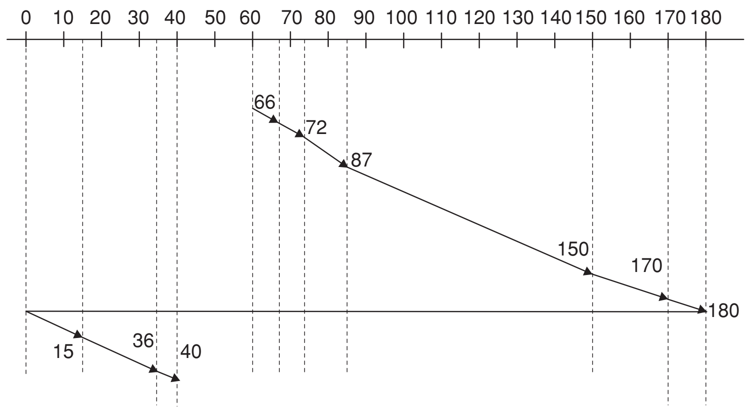



Figure 16 C–SCAN

### Advantage

Lower service variability.

### Disadvantages

1. An average head movement is more compared to SCAN algorithm.
2. Increase in the total seek time

***LOOK/SEEK algorithm*** Look is similar to SCAN, but stops moving inwards (or) outwards when there are no more requests in that direction.

**Example 7:** Consider the following requests:
87, 170, 40, 150, 36, 72, 66, 15 initially head is at track 60 (moves outwards).

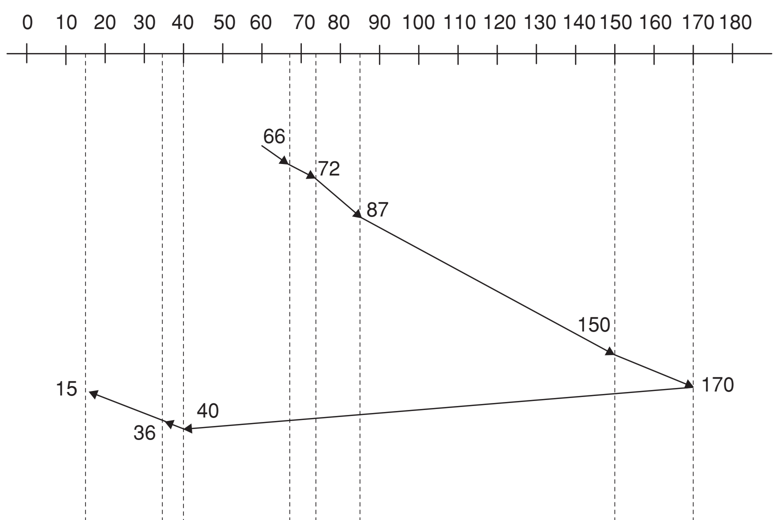



Figure 17 LOOK

Average head movement $= \frac{275}{8} = 34.37$

***C–LOOK/C–SEEK algorithm*** The head moves inwards servicing requests until there are no more requests in that direction. Then it jumps to the outer most outstanding request.

## Magnetic Disk and Factors that Determine Access Speed

The factors that determine the access speed are rotational speed, which is measured in revolutions per minute are seek time, the time taken to read or write the particular sector of the disk.

The other factors are sequential read, sequential write, random read and random write. These vary enormously, but for spinning disks one can expect 25 Mbps to 150 Mbps for sequential read and write it is about 3 Mbps to 50 Mbps for the random read and write.

## RAID (Redundant Array of Independent Disks)

1. It is multiple disk database design.
2. It consists of seven levels, zero through six.
3. Characteristics of RAID Levels:
   - RAID is a set of physical disk drives viewed by the OS as a single logical drive.
   - Data are distributed across the physical drives of an array in a scheme known as striping.
   - Redundant disk capacity is used to store parity information, which guarantees data recoverability in case of a disk failure.

**RAID Level 0:**

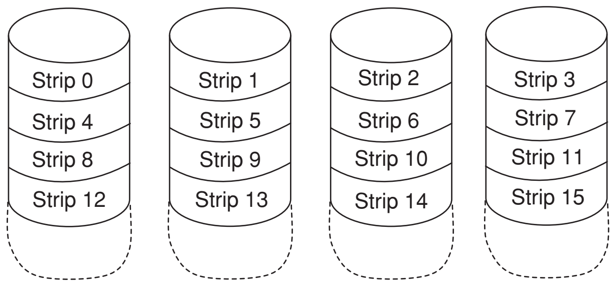



**Figure 18** Non-redundant (RAID0)

1. It does not include redundancy.
2. *N* disks are required.
3. Data available in RAID level 0 is lower than single disk.
4. It has high data transfer capacity.
5. It has high I/O request rate.

**RAID Level 1:**

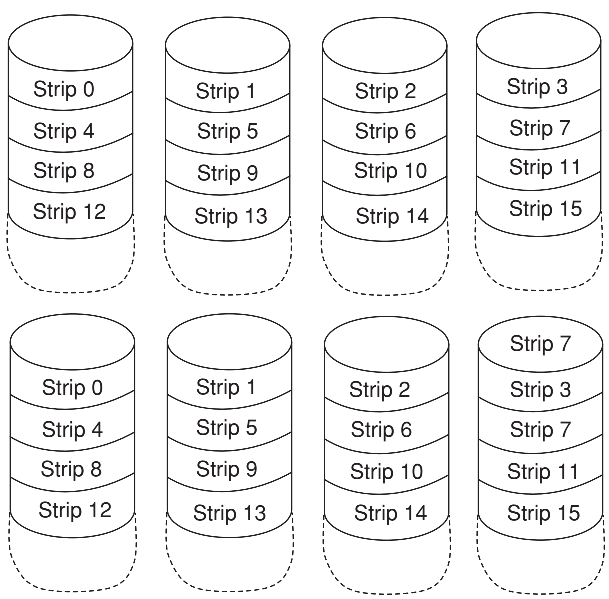
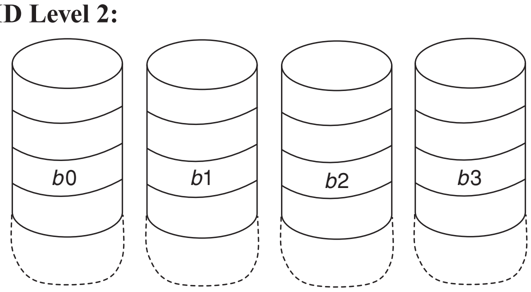



**Figure 19** RIAD 1 (Mirrored)

1. Redundancy is achieved by the simple expedient of duplicating all the data.
2. 2N disks required.
3. Data availability is higher than RAID 2, 3, 4, or 5. But lower than RAID 6.
4. Recovery from failure is simple.
5. RAID1 costs more.

**RAID Level 2:**



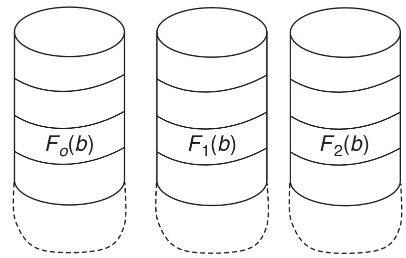




**Figure 20** RAID 2 (Redundancy through Hamming code).

6. Here redundancy is achieved through hamming code.
7. $N + m$ disks required.

**RAID Level 3:**

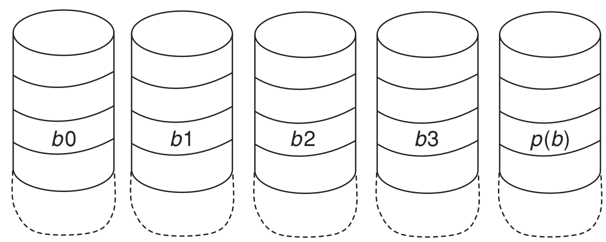



**Figure 21** RAID 3 (bit-interleaved parity)

1. It has bit interleaved parity.
2. Provides parallel access.
3. $N + 1$ disks required.

**RAID Level 4:**

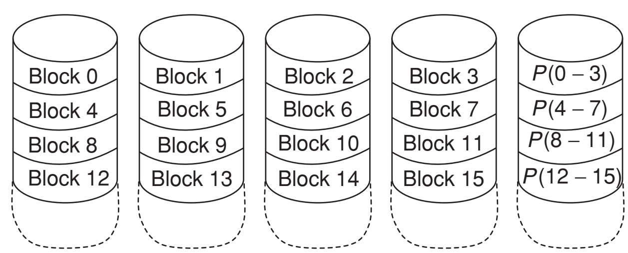



**Figure 22** RAID 4 (Block-level parity)

1. $N + 1$ disks required.
2. Provides independent access.
3. Consider an array of five drives in which $X_0$ through $X_3$ contains data and $X_4$ is the parity disk.

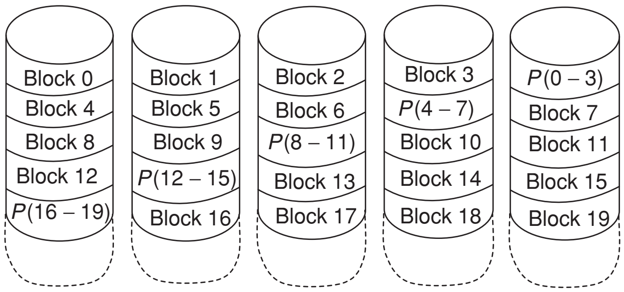
Suppose that a write is performed that only involves a strip on disk $X_1$. Initially for each bit $i$, we have the following relationship:

$$X_4(i) = X_3(i) \oplus X_2(i) \oplus X_1(i) \oplus X_0(i)$$

After the update, with potentially altered bits indicated by a prime symbol:

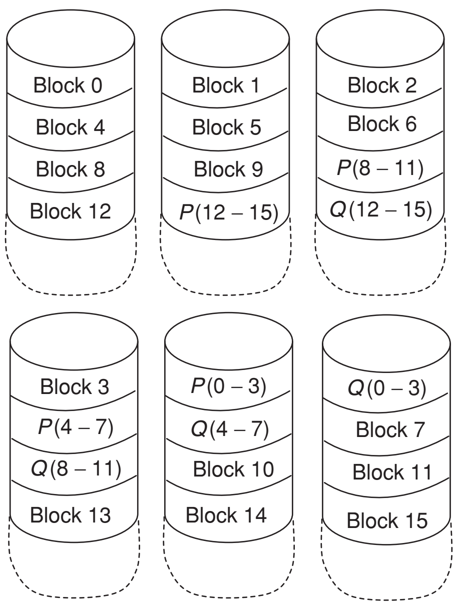
$$\begin{aligned} X_4'(\mathrm{i}) &= X_3(i) \oplus X_2(i) \oplus X_1'(i) \oplus X_0(i) \\ &= X_3(i) \oplus X_2(i) \oplus X_1'(i) \oplus X_0(i) \oplus X_1(i) \oplus X_1(i) \\ &= X_4(i) \oplus X_1(i) \oplus X_1'(i) \end{aligned}$$

**RAID Level 5:**




1. It has block level distributed parity.
2. Provides independent access.
3. It has $N + 1$ disks.

**RAID Level 6:**

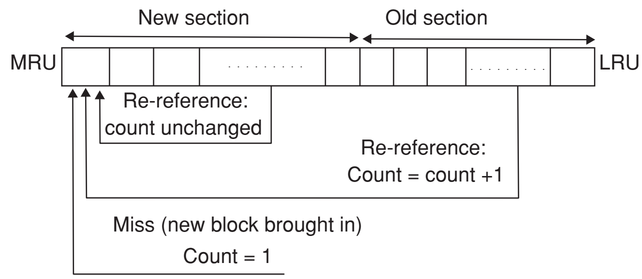



1. It has dual redundancy.
2. It provides independent access.
3. It has $N + 2$ disks.

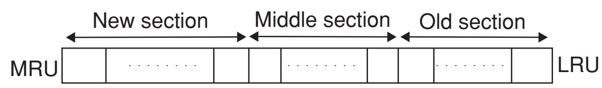
**Disk Cache:**

1. It is a buffer in main memory for disk sectors.
2. The cache contains a copy of some of the sectors on the disk.
3. Replacement policy:
   - LRU
   - LFU

But these two replacement policies lead to poor performance. So a new technique frequency-based replacement is proposed. Two alternatives of frequency-based technique are shown below:




**(A) FIFO**




**(B) Use of three sections.**

In FIFO, the blocks are logically organized in a stack; where the top part of the stack is the new section. When there is a cache hit, the referenced block is moved to top of the stack. If the block was already in the new section, its reference count is not incremented; otherwise it is incremented by 1.

In another technique, we divide the stack into three sections: New, middle and old. Here only blocks in the old section are eligible for replacement.

## Protection and Security

***System protection*** Protection refers to a mechanism for controlling the access of programs, processes or users to the resources defined by a computer system.

### Principles of Protection

Programs, users and even systems are given just enough privileges to perform their tasks. This is the principle of *least privilege*.

***Domain of protection*** A computer system is a collection of processes and objects. The objects may be software or hardware objects. The operations that are possible may depend on the object.

A process should be allowed to access only those resources for which it has authorization. At any time, a process should be able to access only those resources that it currently requires to complete its task. This requirement is referred as *need to know principle* and is useful in limiting the amount of damage, which a faulty process can cause in the system.

***Domain structure*** Each domain defines a set of objects and the types of operations that may be invoked on each object.

The ability to execute an operation on an object is an *access right*. A domain is a collection of access rights, each of which is an ordered pair <obj-name, rights-set>

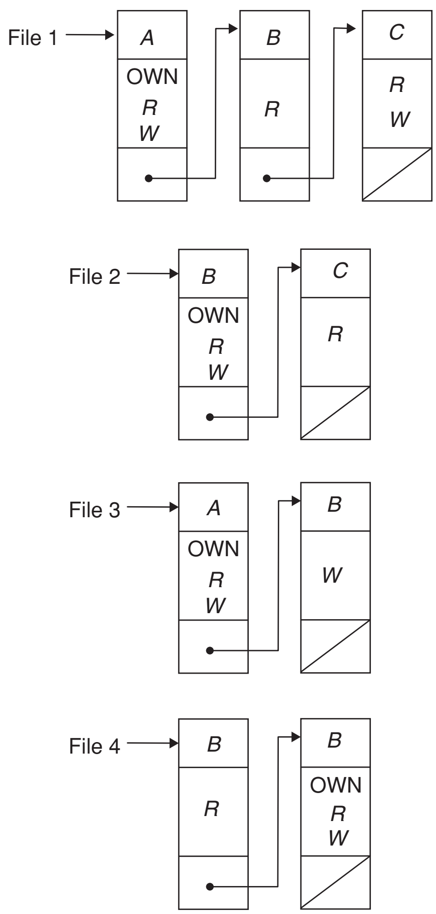
## File System Security

1. A general model of access control for file management is an access matrix.
2. The basic elements of the model are as follows:
   - *Subject*: An entity capable of accessing objects.
   - *Object*: Anything to which access is controlled.
   - *Access right*: The way in which an object is accessed by a subject.

Access matrix format is shown below:

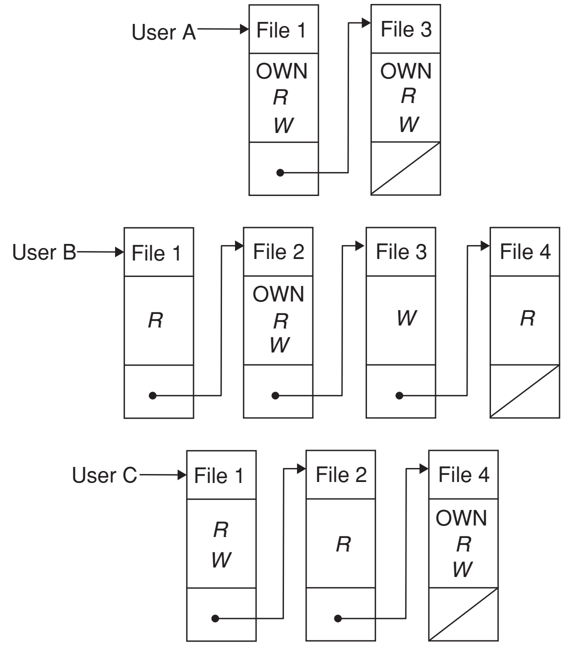
| | File 1 | File 2 | File 3 | File 4 | Account 1 | Account 2 |
|---|---|---|---|---|---|---|
| User A | OWN R W | | OWN R W | | Inquiry credit | |
| User B | R | OWN R W | W | R | Inquiry debit | Inquiry credit |
| User C | R W | R | | OWN R W | | Inquiry Debit |

The access matrix may be decomposed by columns, yielding access control lists. The access control list for above matrix is




**Note:** This allows users that are not explicitly listed as having special rights to have a default set of rights.

Decomposition by row yields. Capability tickets.




**Note:** These tickets would have to be held in a region of memory inaccessible to users.

***System security*** This is the protection afforded to an automated information system in order to attain objectives of preserving the integrity, availability and confidentiality of information system resources.

## Objectives of Computer Security

***Confidentiality*** Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information.

***Integrity*** Guarding against improper information modification or destruction, including ensuring information nonrepudiation and authenticity.

***Availability*** Ensuring timely and reliable access to and use of information.

***Authenticity*** The property of being genuine and being able to be verified and trusted.

***Accountability*** The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity.

## Threats, Attacks and Assets

***Threats and attacks*** *Unauthorized disclosure* is the threat to confidentiality. The attacks of following this threat are as follows:

1. *Exposure*: Sensitive data are directly released to an unauthorized entity.
2. *Interception*: An unauthorized entity directly accesses sensitive data traveling between authorized sources and destinations.

3. *Inference*: A threat action where an unauthorized entity indirectly accesses sensitive data by reasoning from characteristics or by-products of communications.
4. *Intrusion*: An unauthorized entity gains access to sensitive data by circumventing a system's security protections.

**Deception** Threat to either system integrity or data integrity. Types of attacks that can result are as follows:

1. *Masquerade*: An unauthorized entity gains access to a system or performs a malicious act by posing as an authorized entity.
2. *Falsification*: False data deceive an authorized entity.
3. *Repudiation*: An entity deceives another by falsely denying responsibility for an act.

**Disruption** A circumstance or event that interrupts or prevents the correct operation of system services and functions. Attacks for this threat are as follows:

1. *Incapacitation*: Prevents or interrupts system operation by disabling a system component.
2. *Corruption*: Undesirably alters system operation by adversely modifying system functions or data.
3. *Obstruction*: A threat action that interrupts delivery of system service by hindering system operation.

**Usurpation** A circumstance or event that results in control of system services or functions by an unauthorized entity. Attacks with this threat are as follows:

1. *Misappropriation*: An entity assumes unauthorized logical or physical control of a system resource.
2. *Misuse*: Causes a system component to perform a function or service that is detrimental to system security.

**Threats and assets** The assets of a computer are as follows:
1. Hardware
2. Software
3. Data
4. Communication lines

**Hardware** A major threat to computer system hardware is the threat to availability (e.g., theft of CD-ROMS).

**Software**
1. A key threat to software is an attack on availability (e.g., deletion of software).
2. A threat to integrity.
3. A threat to confidentiality.

**Data** Threats to data are an attack on
1. Availability
2. Confidentiality
3. Integrity

**Communication lines and networks** Two types of attacks:
1. Passive attacks
2. Active attacks

**Passive attacks**
1. These are in the nature of monitoring of the transmissions.
2. Attackers obtain information that is being transmitted.
3. Two types of passive attacks:
   - Release of message contents.
   - Traffic analysis
4. These are very difficult to detect because they do not involve any alteration of the data.

**Active attacks**
1. These attacks involves some modification of the date stream or the creation of a false stream and can be subdivided into four categories:
   - Replay
   - Masquerade
   - Modification of messages
   - Denial of service
2. It is difficult to prevent active attacks absolutely.

**Intruders** Three types of intruders:

**Masquerader** An individual who is not authorized to use the computer and who penetrates a systems access controls to exploit a legitimate user's account. He is likely to be an outsider.

**Misfeasor** A legitimate user who accesses data, programs or resources for which such access is not authorized or who is authorized for such access but misuses his or her privileges (generally an insider).

**Clandestine user** An individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection (either outsider or insider).

**Hackers** Those who hack into computers do so for the thrill of it or for status. Attackers often took for targets of opportunity and then share the information with others.

**Criminals** Organized group of hackers have become a widespread and common threat to internet based systems.

**Malicious software overview** The most sophisticated types of threats to computer systems are presented by programs that exploit vulnerabilities in computing systems. These threats are referred as malicious software (or) malware.

1. It is designed to cause damage to or use up the resources of a target computer.
2. There are two types of malicious software:
   - Those that need a host program.
     **Example:** Viruses, logic bombs.
   - Those that are independent.
     **Example:** Worms, bot programs.
3. We can also differentiate between two types of software threats:
   - That do not replicate. These programs are activated by a trigger.
     **Example:** Logic bombs, backdoors.

- Those that replicate.
  **Example:** Viruses, worms

***Backdoor (trapdoor)*** It is a secret entry point into a program that allows someone who is aware of the backdoor to gain access without going through usual security access procedures.

***Logic bomb*** It is a program inserted into software by an intruder. It lies dormant until a predefined condition is met; the program then triggers an unauthorized act.

***Trojan Horse*** It is an apparently useful program, containing hidden code that, when invoked, performs some unwanted or harmful function.

***Mobile code*** It is a software that can be shipped unchanged to a heterogeneous collection of platforms and execute with identical semantics.

***Viruses*** A computer virus is a piece of software that can infect other programs by modifying them.

## Nature of Viruses

A virus can do anything that other programs do. It attaches itself to another program and executes secretly when the host program is run. Three parts of computer virus are as follows:

1. Infection mechanism
2. Trigger
3. Payload

Phases of computer virus are

1. Dormant phase
2. Propagation phase
3. Triggering phase
4. Execution phase

## Types of Virus

1. Encrypted virus
2. Stealth virus
3. Polymorphic virus
4. Metamorphic virus

***Worms*** A worm is a program that can replicate itself and send copies from computer to computer across network connections.

***Bots*** A bot is a program that secretly takes over another internet attached computer and then uses that computer to launch attacks that are difficult to trace to the bot's creator.

# Exercises

## Practice Problems I

***Directions for questions 1 to 15:*** Select the correct alternative from the given choices.

**1.** Given a system using unspanned blocking and 100 byte blocks. A file contains records 30, 40, 55, 80, 30, 40. What percentage of space will be wasted in the blocks allocated for the file?
(A) 31.25% (B) 41.25%
(C) 51.25% (D) 62.15%

**2.** Disk requests come into the disk driver for cylinders 15, 25, 10, 2, 35, 9, 42 in that order. The disk head is currently positioned over cylinder 15. A seek takes 6 msec per cylinder moved. What is the total seek time using First Come First Served Algorithm?
(A) 750 msec (B) 650 msec
(C) 550 msec (D) 450 msec

**3.** A Java application needs to load 50 libraries. To load each library, one disk access is required. Seek time to access the location is 10 ms. Rotational speed is 6000 rpm. The total time needed to load all libraries is
(A) 0.65 sec (B) 0.75 sec
(C) 0.85 sec (D) 1 sec

**4.** A program has just read the 13th record in a sequential access file. If it wants to read the 10th record next, how many records must the program read to input the tenth record?
(A) 5 (B) 0
(C) 10 (D) 13

**5.** A disk is formatted into 40 sectors and 20 tracks. The disk rotates at 200 ms in one revolution. The time taken by the head to move from the centre to the rim is 10 ms. There are three different files stored on the disk:
File $P$ : Sector 2, track 4
File $Q$ : Sector 5, track 1
File $R$ : Sector 6, track 2
Calculate the average latency time required for the three files.
(A) 22.55 ms (B) 32.22 ms
(C) 21.66 ms (D) 30.22 ms

**6.** Match the following

| | |
|---|---|
| (a) RAID0 | (1) Parallel access |
| (b) RAID1 | (2) Striping |
| (c) RAID2 | (3) Use hamming code |
| (d) RAID3 | (4) Mirrored |

(A) a – 2, b – 4, c – 3, d – 1
(B) a – 1, b – 2, c – 3, d – 4
(C) a – 3, b – 2, c – 4, d – 1
(D) a – 4, b – 1, c – 2, d – 3

**7.** The correct matching for the following pairs is

| | |
|---|---|
| (A) Disk scheduling | (1) Round Robin |
| (B) Batch processing | (2) SCAN |
| (C) Time sharing | (3) LIFO |
| (D) Interrupt processing | (4) FIFO |

(A) A – 3, B – 4, C – 2, D – 1
(B) A – 4, B – 3, C – 2, D – 1
(C) A – 2, B – 4, C – 1, D – 3
(D) A – 2, B – 1, C – 4, D – 3

**8.** A program *P* reads and processes 2000 consecutive records from a sequential file stored on device *R* without using any file system facilities. Given the following:
Size of each record = 3500 bytes
Access time of *R* = 20 ms
Data transfer rate of *R* = $500 \times 10^3$ bytes/sec
CPU time to process each record = 5 m sec
What is the elapsed time of *P* if *R* contains unblocked records and *P* does not use buffering?
(A) 64 sec (B) 46 sec
(C) 34 sec (D) 17 sec

**9.** A disk has 19456 cylinders, 16 heads and 63 sectors per track. The disk spins at 5400 rpm. Seek time between adjacent tracks is 2 ms. Assuming the read/write head is already positioned at track 0, how long does it take to read the entire disk?
(A) 35 min (B) 68 min
(C) 58 min (D) 53 min

**10.** On a disk with 1000 cylinders, numbered 0 to 999, compute the number of tracks the disk arm must move to satisfy all the requests in the disk queue using SCAN algorithm. Assume the last request serviced was at track 345 and the head is moving towards 0. The queue in FIFO order contains requests for the following tracks:
123, 874, 692, 475, 105, 376
(A) 219 (B) 635
(C) 845 (D) 1219

**Common data for questions 11 to 14:** In the operation of a certain disk drive mechanism, a disk is formatted into 20 sectors and 10 tracks. The disk can be rotated either clockwise or anti-clockwise. The times required to perform certain operations are as follows:

I. Rotate the disk through one revolution = 200 ms
II. Move the disk head from the centre to the rim = 20 ms
III. Read and transmit one block of data = 0.3 ms

Three files are stored on the disk:
File *A*: 2 blocks at track 6
File *B*: 5 blocks at track 2
File *C*: 1 block at track 5

**11.** The disk head is initially at sector 0, track 0. If all three files *A*, *B* and *C* are to be read in the minimum amount of time, they should be read in the following order:
(A) *A*, *B*, *C* (B) *A*, *C*, *B*
(C) *B*, *C*, *A* (D) *C*, *A*, *B*

**12.** The disk head is initially at sector 0, track 0. The files are read in the order *C*, *B*, *A*. The total time to read the files is
(A) 143.9 ms (B) 100.4 ms
(C) 114.0 ms (D) 102.6 ms

**13.** The most nearly average latency time for the sequence *CBA* is:
(A) 1 ms (B) 7 ms
(C) 27 ms (D) 50 ms

**14.** The most nearly average seek time for *CBA* is
(A) 1 ms (B) 8 ms
(C) 30 ms (D) 50 ms

**15.** A CD has 150 tracks rotating at 3500 rpm. Average seek time for consecutive tracks is 0.1 ms, the disk is subjected to read data from the track numbers 89, 75, 112, 5. What is the total seek time if the requests are served unidirectionally (C-Scan) and the first request determines initial direction? Assume that the current position of the head is at track 100.
(A) 20.2 ms (B) 16.9 ms
(C) 21.3 ms (D) 14 ms

## Practice Problems 2

***Directions for questions 1 to 15:*** Select the correct alternative from the given choices.

**1.** The strategy that allocates the smallest possible chunk of disk space that is sufficient to the file is
(A) Nearest fit (B) Best fit
(C) Worst fit (D) First fit

**2.** If a process of 200 kB is transferred from backing store to memory and average disk latency is 10 ms, then what would be the total swap time, if transfer ratio is 2 Mbps?
(A) 10 ms
(B) 20 ms
(C) 30 ms
(D) 40 ms

**3.** Let us assume that the user process is 10 MB in size and backing store is a standard hard disk with a transfer rate of 40 MB per second. Let the average latency is 8 millisecond. Find actual time transfer of the 10 MB process to or from main memory?
(A) 8 ms (B) 250 ms
(C) 258 ms (D) 516 ms

**4.** Disk scheduling involves deciding
(A) which disk should be accessed next
(B) the order in which disk access requests must be serviced.
(C) the physical location when files should be accessed in the disk
(D) disk access time and an unused space.

5. The root directory of a disk should be placed
   (A) at a fixed address in the memory
   (B) anywhere on the disk
   (C) at a fixed location on the system disk
   (D) at a location on floppy.
6. Direct access methods are not effectively supported by
   (A) Contiguous allocation
   (B) Linked allocation
   (C) Indexed allocation
   (D) Sequential allocation
7. In which of the following directory systems, it is possible to have multiple paths for a file, starting from the root directory?
   (A) Single-level directory
   (B) Two-level directory
   (C) Tree-structured directory
   (D) A cyclic graph directory
8. The most common system's security method is:
   (A) Passwords
   (B) Key card systems
   (C) Surveillance system
   (D) Lock system
9. Trojan Horse programs
   (A) are legitimate programs that allow unauthorized access.
   (B) are hacker programs that do not show up on the system
   (C) really do not work
   (D) are immediately discovered
10. Which of the following is a program that spreads throughout the network?
    (A) Trojan Horse (B) Virus
    (C) TSR (D) Worm
11. A program has just read the 15th record in a sequential access file. If it wants to read the 10th record next, how many records must the program read to input the tenth record?
    (A) 0 (B) 5
    (C) 4 (D) 10
12. Formatting of a floppy disk refers to
    (A) Arranging the data on the disk in contiguous fashion
    (B) Writing the directory
    (C) Erasing the system area
    (D) Writing identification information on all tracks
13. Sector interleaving in disks is done by
    (A) the disk manufacturer
    (B) the disk controller card
    (C) the operating system
    (D) the user
14. Disk I/O is done in terms of
    (A) Tracks
    (B) Blocks
    (C) Bits
    (D) Bytes
15. How many six-letter passwords can be constructed using lowercase letters and digits?
    (A) $26^6$ (B) $10^6$
    (C) $36^6$ (D) $35^6$

## Previous Years' Questions

1. Consider a disk drive with the following specifications: **[2005]**

   16 surfaces, 512 tracks/surfaces, 512 sectors/track, 1 KB/sector, rotation speed 3000 rpm. The disk is operated in cycle stealing mode whereby whenever one 4 byte word is ready it is sent to memory; similarly, for writing, the disk interface reads a 4 byte word from the memory in each DMA cycle. Memory cycle time is 40 nsec. The maximum percentage of time that the CPU gets blocked during DMA operation is:
   (A) 10 (B) 25
   (C) 40 (D) 50
2. Consider a disk pack with 16 surfaces, 128 tracks per surface and 256 sectors per track. 512 bytes of data are stored in a bit serial manner in a sector. The capacity of the disk pack and the number of bits required to specify a particular sector in the disk are, respectively: **[2007]**
   (A) 256 Mbyte, 19 bits
   (B) 256 Mbyte, 28 bits
   (C) 512 Mbyte, 20 bits
   (D) 64 Gbyte, 28 bits
3. For a magnetic disk with concentric circular tracks, the seek latency is not linearly proportional to the seek distance due to **[2008]**
   (A) non-uniform distribution of requests
   (B) arm starting and stopping inertia
   (C) higher capacity of tracks on the periphery of the platter
   (D) use of unfair arm scheduling policies
4. Consider a file of 16384 records. Each record is 32 bytes long and its key field is of size 6 bytes. The file is ordered on a non-key field, and the file organization is unspanned. The file is stored in a file system with block size 1024 bytes, and the size of a block pointer is 10 bytes. If the secondary index is built on the key field of the file, and a multi-level index scheme is used to store the secondary index, the number of first-level and second-level blocks in the multi-level index are, respectively **[2008]**

(A) 8 and 0 (B) 128 and 6
(C) 256 and 4 (D) 512 and 5

**5.** Consider a disk system with 100 cylinders. The requests to access the cylinders occur in following sequence: **[2009]**

4, 34, 10, 7, 19, 73, 2, 15, 6, 20

Assuming that the head is currently at cylinder 50, what is the time taken to satisfy all requests if it takes 1ms to move from one cylinder to adjacent one and shortest seek time first policy is used?

(A) 95 ms (B) 119 ms
(C) 233 ms (D) 276 ms

**Common data for questions 6 and 7:** A hard disk has 63 sectors per track, 10 platters each with 2 recording surfaces and 1000 cylinders. The address of a sector is given as a triple $\langle c, h, s \rangle$, where c is the cylinder number, h is the surface number and s is the sector number. Thus, the 0th sector is addressed as $\langle 0, 0, 0 \rangle$, the 1st sector as $\langle 0, 0, 1 \rangle$, and so on

**6.** The address <400, 16, 29> corresponds to the sector number: **[2009]**

(A) 505035 (B) 505036
(C) 505037 (D) 505038

**7.** The address of the 1039th sector is **[2009]**

(A) $\langle 0, 15, 31 \rangle$ (B) $\langle 0, 16, 30 \rangle$
(C) $\langle 0, 16, 31 \rangle$ (D) $\langle 0, 17, 31 \rangle$

**8.** A file system with 300 GByte disk uses a file descriptor with 8 direct block addresses, 1 indirect block address and 1 doubly indirect block address. The size of each disk block is 128 bytes and the size of each disk block address is 8 bytes. The maximum possible file size in this file system is **[2012]**

(A) 3 Kbytes
(B) 35 Kbytes
(C) 280 Kbytes
(D) dependent on the size of the disk

**9.** Consider a hard disk with 16 recording surfaces (0–15) having 16384 cylinders (0–16383) and each cylinder contains 64 sectors (0–63). Data storage capacity in each sector is 512 bytes. Data are organized cylinder-wise and the addressing format is <cylinder no., surface no., sector no.>. A file of size 42797 KB is stored in the disk and the starting disk location of the file is <1200, 9, 40>. What is the cylinder number of the last sector of the file, if it is stored in a contiguous manner? **[2013]**

(A) 1281 (B) 1282
(C) 1283 (D) 1284

**10.** A FAT (File allocation table)-based file system is being used, and the total over head of each entry in the FAT is 4 bytes in size. Given a $100 \times 10^6$ bytes disk on which the file system is stored and data block size is $10^3$ bytes, the maximum size of a file that can be stored on this disk in units of $10^6$ bytes is ______? **[2014]**

**11.** Consider a disk pack with a seek time of 4 milliseconds and rotational speed of 10000 rotations per minute (RPM). It has 600 sectors per track and each sector can store 512 bytes of data. Consider a file stored in the disk. The file contains 2000 sectors. Assume that every sector access necessitates a seek, and the average rotational latency for accessing each sector is half of the time for one complete rotation. The total time (in milliseconds) needed to read the entire file is ________. **[2015]**

**12.** Suppose the following disk request sequence (track numbers) for a disk with 100 tracks is given: 45, 20, 90, 10, 50, 60, 80, 25, 70. Assume that the initial position of the R/W head is on track 50. The additional distance that will be traversed by the R/W head when the Shortest Seek Time First (SSTF) algorithm is used compared to the SCAN (Elevator) algorithm (assuming that SCAN algorithm moves towards 100 when it starts execution) is ______ tracks. **[2015]**

**13.** Consider a typical disk that rotates at 15000 rotations per minute (RPM) and has a transfer rate of $50 \times 10^6$ bytes/sec. If the average seek time of the disk is twice the average rotational delay and the controller's transfer time is 10 times the disk transfer time, the average time (in milliseconds) to read or write a 512-byte sector of the disk is ________ **[2015]**

**14.** Consider a disk queue with requests for I/O to blocks on cylinders 47, 38, 121, 191, 87, 11, 92, 10. The C-LOOK scheduling algorithm is used. The head is initially at cylinder number 63, moving towards larger cylinder numbers on its servicing pass. The cylinders are numbered from 0 to 199. The total head movement (in number of cylinders) incurred while servicing these requests is _____. **[2016]**

**15.** In a file allocation system, which of the following allocation scheme(s) can be used if no external fragmentation is allowed? **[2017]**

I. Contiguous
II. Linked
III. Indexed

(A) I and III only (B) II only
(C) III only (D) II and III only

**16.** Consider a storage disk with 4 platters (numbered as 0, 1, 2 and 3), 200 cylinders (numbered as 0, 1, ..., 199), and 256 sectors per track (numbered as 0, 1, ..., 255). The following 6 disk requests of the form [sector number, cylinder number, platter number] are received by the disk controller at the same time:

[120, 72, 2], [180, 134, 1], [60, 20, 0], [212, 86, 3], [56, 116, 2], [118, 16, 1]

Currently the head is positioned at sector number 100 of cylinder 80, and is moving towards higher cylinder numbers. The average power dissipation in moving the head over 100 cylinders is 20 milliwatts and for reversing the direction of the head movement once is 15 milliwatts. Power dissipation associated with rotational latency and switching of head between different platters is negligible.

The total power consumption in milliwatts to satisfy all of the above disk requests using the Shortest Seek Time First disk scheduling algorithm is ________. **[2018]**

## Answer Keys

## Exercises

### Practice Problems 1

**1.** A **2.** A **3.** B **4.** C **5.** C **6.** A **7.** C **8.** A **9.** C **10.** D
**11.** C **12.** B **13.** C **14.** B **15.** A

### Practice Problems 2

**1.** B **2.** C **3.** D **4.** B **5.** C **6.** D **7.** C **8.** A **9.** A **10.** D
**11.** D **12.** D **13.** C **14.** B **15.** C

### Previous Years' Questions

**1.** B **2.** A **3.** C **4.** C **5.** B **6.** C **7.** C **8.** B **9.** D **10.** 99.6
**11.** 14020 **12.** 10 **13.** 6.1 to 6.2 **14.** 346 **15.** D **16.** 85