Operating System (OS)

Unit 4 **File System**

A file system is a method an operating system uses to store, organize, and manage files and directories on a storage device. Some common types of file systems include:

- **FAT (File Allocation Table):** An older file system used by older versions of Windows and other operating systems.
- **NTFS (New Technology File System):** A modern file system used by Windows. It supports features such as file and folder permissions, compression, and encryption.
- **ext (Extended File System):** A file system commonly used on Linux and Unix based operating system.
- **HFS (Hierarchical File System):** A file system used by macOS.
- **APFS (Apple File System):** A new file system introduced by Apple for their Macs and iOS devices.

Following Issues Are Handled By The File Syst4em

We’ve seen a variety of data structures where the file could be kept. The file system‘s job is to keep the files organized in the best way possible.

A free space is created on the hard drive whenever a file is deleted from it. To reallocate them to other files, many of these spaces may need to be recovered. Choosing where to store the files on the hard disc is the main issue with files one block may or may not be used to store a file. It may be kept in the disk’s non-contiguous blocks. We must keep track of all the blocks where the files are partially located.

File type	Usual extension	Function
Executable	exe, com, bin	Read to run machine language program
Object	obj, o	Compiled, machine language not linked
Source Code	C, java, pas, asm, a	Source code in various languages
Batch	bat, sh	Commands to the command interpreter
Text	txt, doc	Textual data, documents
Word Processor	wp, tex, rrf, doc	Various word processor formats
Archive	arc, zip, tar	Related files grouped into one compressed file
Multimedia	mpeg, mov, rm	For containing audio/video information

- **FILE DIRECTORIES**
- The collection of files is a file directory. The directory contains information about the files, including attributes, location, and ownership. Much of this information, especially that is concerned with storage, is managed by the operating system. The directory is itself a file, accessible by various file management routines.

Information contained in a device directory is:

- Name
- Type
- Address
- Current length
- Maximum length
- Date last accessed
- Date last updated
- Owner id
- Protection information

The operation performed on the directory are:

- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

Physical and Logical File Systems

- **1. Physical files:** Physical files contain the actual data that is stored on an iSeries system, and a description of how data is to be presented to or received from a program. They contain only one record format and one or more members. Records in database files can be described using either a field-level description or a record-level description. A field-level description describes the fields in the record to the system. Database files that are created with field-level descriptions are referred to as externally described files. A record-level description describes only the length of the record, and not the contents of the record. Database files that are created with record-level descriptions are referred to as program-described files.

- **2. Logical files:** Logical files do not contain data. They contain a description of records that are found in one or more physical files. A logical file is a view or representation of one or more physical files. Logical files that contain more than one format are referred to as multi-format logical files. If your program processes a logical file that contains more than one record format, you can use the _Rformat() function to set the format you wish to use. Some operations cannot be performed on logical files.

- If you open a logical file for stream file processing with open modes W, W+, WB, or WB+, the file is opened but not cleared. If you open a logical file for record file processing with open modes WR or WR+, the file is opened but not cleared. Records in iSeries database files can be described using either a field-level description or a record-level description. The field-level description of the record includes a description of all fields and their arrangement in this record. Since the description of the fields and their arrangement is kept within a database file and not in your ILE C/C++ program, database files created with a field-level description are referred to as externally described files

- **Physical File:** A collection of bytes stored on a disk or tape.
- **Logical File:** A “Channel” (like a telephone line) that hides the details of the file’s location and physical format to the program.

Physical File	Logical File
It occupies the portion of memory. It contains the original data.	It does not occupy memory space. It does not contain data.
A physical file contains one record format.	It can contain up to 32 record formats.
It can exist without a logical file.	It cannot exist without a physical file.
If there is a logical file for the physical file, the physical file cannot be deleted until and unless we delete the logical file.	If there is a logical file for a physical file, the logical file can be deleted without deleting the physical file.
CRTPF command is used to make such an object.	CRTLF command is used to make such an object.

File Allocation Methods

- The allocation methods define how the files are stored in the disk blocks. There are three main disk space or file allocation methods.
- Contiguous Allocation
- Linked Allocation
- Indexed Allocation
- The main idea behind these methods is to provide:
- Efficient disk space utilization.
- Fast access to the file blocks.
- All the three methods have their own advantages and disadvantages as discussed below:

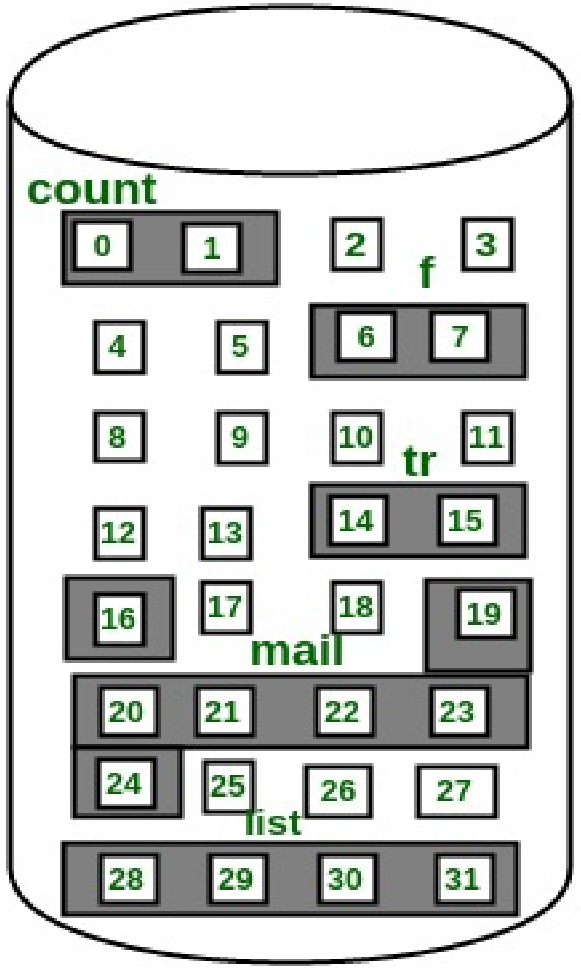
1. Contiguous Allocation

- 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

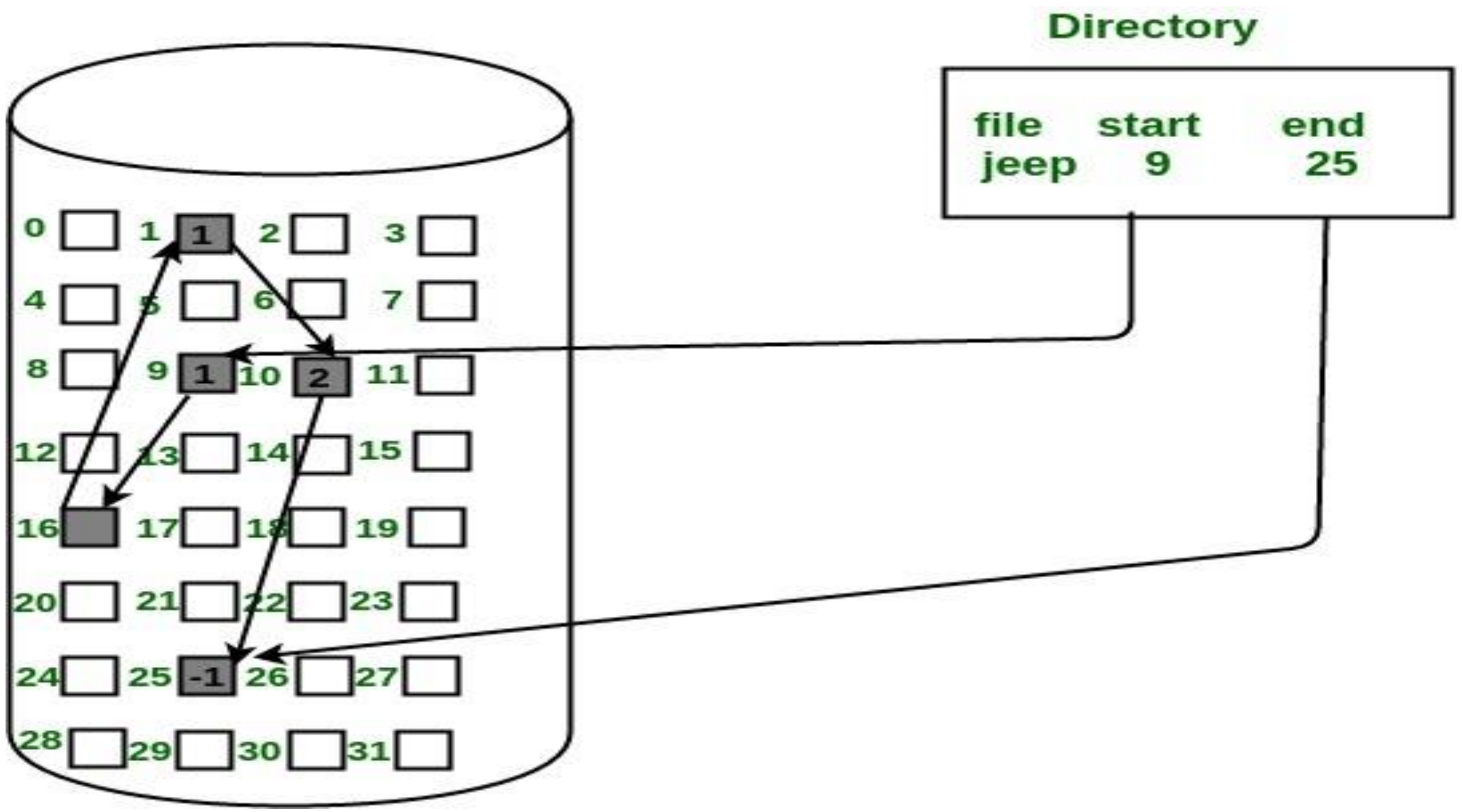
- Length of the allocated portion.
- The *file ‘mail’* in the following figure starts from the block 19 with length = 6 blocks. Therefore, it occupies *19*, *20*, *21*, *22*, *23*, *24* blocks.



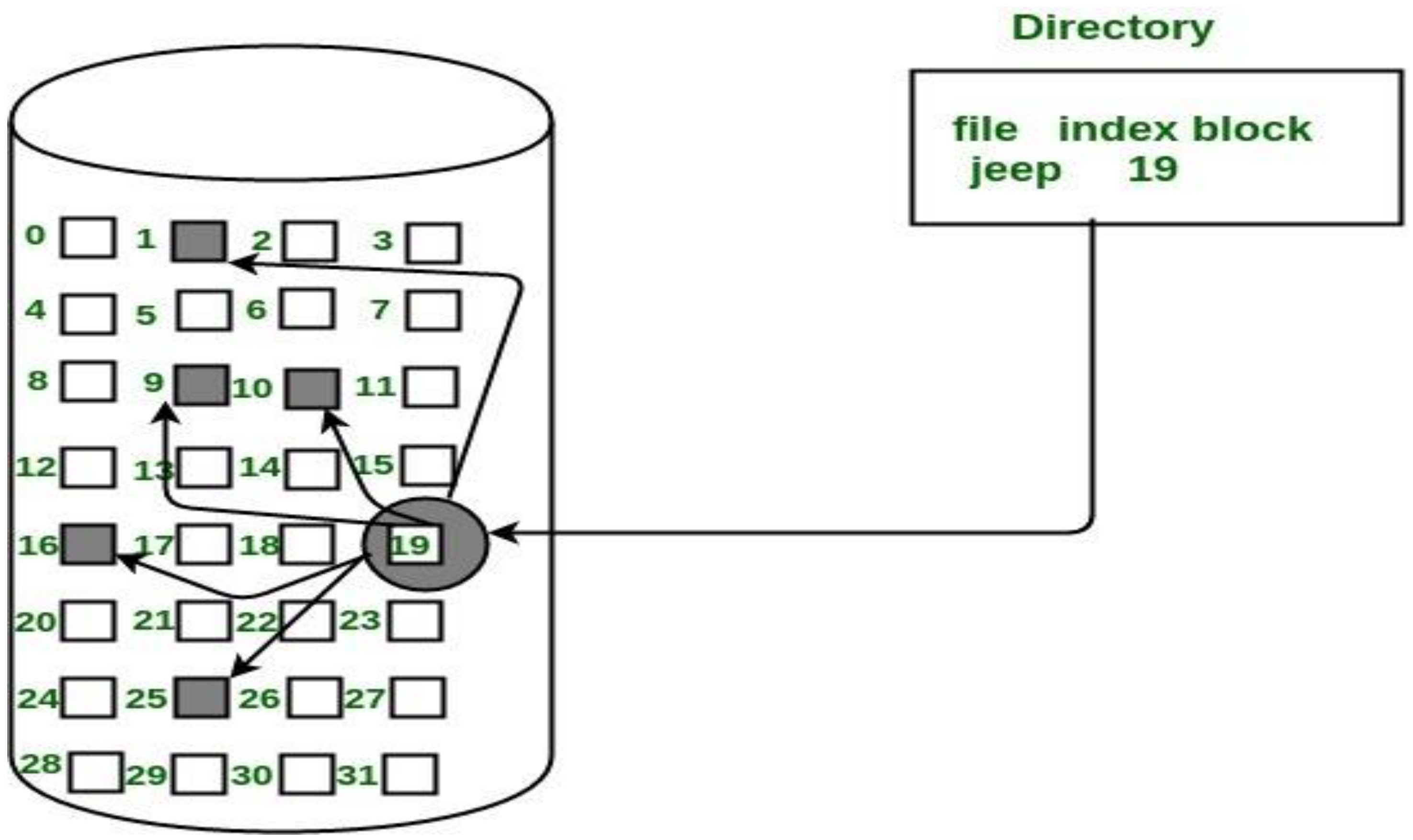
Directory

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

- **2. Linked List Allocation**
- In this scheme, each file is a linked list of disk blocks which **need not be** contiguous. The disk blocks can be scattered anywhere on the disk.
  The directory entry contains a pointer to the starting and the ending file block. Each block contains a pointer to the next block occupied by the file.
- *The file 'jeep' in following image shows how the blocks are randomly distributed. The last block (25) contains -1 indicating a null pointer and does not point to any other block.*


Directory
file start end
jeep 9 25
0
1
1
2
3
4
5
6
7
8
9
1
10
2
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
-1
26
27
28
29
30
31

- **3. Indexed Allocation**
- In this scheme, a special block known as the **Index block** contains the pointers to all the blocks occupied by a file. Each file has its own index block. The ith entry in the index block contains the disk address of the ith file block. The directory entry contains the address of the index block as shown in the image:



Directory
file index block
jeep 19
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Free Space Management

- Free space management is a critical aspect of operating systems as it involves managing the available storage space on the hard disk or other secondary storage devices. The operating system uses various techniques to manage free space and optimize the use of storage devices. Here are some of the commonly used free space management techniques:
- Linked Allocation: In this technique, each file is represented by a linked list of disk blocks. When a file is created, the operating system finds enough free space on the disk and links the blocks of the file to form a chain. This method is simple to implement but can lead to fragmentation and wastage of space.
- Contiguous Allocation: In this technique, each file is stored as a contiguous block of disk space. When a file is created, the operating system finds a contiguous block of free space and assigns it to the file. This method is efficient as it minimizes fragmentation but suffers from the problem of external fragmentation.

- Indexed Allocation: In this technique, a separate index block is used to store the addresses of all the disk blocks that make up a file. When a file is created, the operating system creates an index block and stores the addresses of all the blocks in the file. This method is efficient in terms of storage space and minimizes fragmentation.
- File Allocation Table (FAT): In this technique, the operating system uses a file allocation table to keep track of the location of each file on the disk. When a file is created, the operating system updates the file allocation table with the address of the disk blocks that make up the file. This method is widely used in Microsoft Windows operating systems.
- Volume Shadow Copy: This is a technology used in Microsoft Windows operating systems to create backup copies of files or entire volumes. When a file is modified, the operating system creates a shadow copy of the file and stores it in a separate location. This method is useful for data recovery and protection against accidental file deletion.

- Contiguous Allocation: In this technique, each file is stored as a contiguous block of disk space. When a file is created, the operating system finds a contiguous block of free space and assigns it to the file. This method is efficient as it minimizes fragmentation but suffers from the problem of external fragmentation.
- Indexed Allocation: In this technique, a separate index block is used to store the addresses of all the disk blocks that make up a file. When a file is created, the operating system creates an index block and stores the addresses of all the blocks in the file. This method is efficient in terms of storage space and minimizes fragmentation.

- File Allocation Table (FAT): In this technique, the operating system uses a file allocation table to keep track of the location of each file on the disk. When a file is created, the operating system updates the file allocation table with the address of the disk blocks that make up the file. This method is widely used in Microsoft Windows operating systems.
- Volume Shadow Copy: This is a technology used in Microsoft Windows operating systems to create backup copies of files or entire volumes. When a file is modified, the operating system creates a shadow copy of the file and stores it in a separate location. This method is useful for data recovery and protection against accidental file deletion.

File Access Control

File access control is the technique of assigning or restricting user access to certain files. It ensures that sufficient information is provided for authorized users, but is kept safe from malicious intruders attempting to launch file-based attacks or instigate data breach incidents.

- The operating system protects the files associated with each account and allows or disallows other users to view or modify files that belong to other accounts. POSIX-compliant systems (like Unix, Linux, and macOS) define three kinds of file access: by the file owner (called the user), by a group, and by others (everyone else). A group consists of one or more accounts and has an intermediate level of access, which can allow more access than others while at the same time allowing less access than the user. Windows maintains this information in an Access Control List or ACL, but *some* file information is still available using the following technique. POSIX systems compress or encode file type and access information into a 16-bit integer to save disk space:

- **7-bits for the mode**: one of the seven bits distinguishes between a regular file and a directory (1 for a directory and 0 for a regular file). The other six bits indicate other file attributes that we ignore for simplicity.
- **9-bits for file access control**: controlled by three sets of three bits; in each set, 1 grants permission, and 0 disallows it. The three sets are named:
  – User
  – Group
  – Others

File access control encoded with bits.

	Binary	Decimal	Hexadecimal
Read	100	4	0x4
Write	010	2	0x2
Execute	001	1	0x1

Binary	Decimal	Permissions
000	0	none
001	1	execute
010	2	write
011	3	write + execute
100	4	read
101	5	read + execute
110	6	read + write
111	7	read + write + execu

- **File permissions**. For each kind of owner, the operating system encodes file permissions as a 3-bit pattern.
- **File access permission combinations**. With three bits, file systems can represent eight unique permission combinations, including no access.

Data Access Techniques

When a file is used, information is read and accessed into computer memory and there are several ways to access this information of the file. There are three ways to access a file into a computer system: Sequential-Access, Direct Access, Index sequential Method.

Sequential Access –

It is the simplest access method. Information in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editor and compiler usually access the file in this fashion. Read and write make up the bulk of the operation on a file. A read operation -*read next*- read the next position of the file and automatically advance a file pointer, which keeps track I/O location. Similarly, for the -write *next*- append to the end of the file and advance to the newly written material.

Keypoints

- Data is accessed one record right after another record in an order.
- When we use read command, it move ahead pointer by one
- When we use write command, it will allocate memory and move the pointer to the end of the file
- Such a method is reasonable for tape.

2.Direct Access –

Another method is *direct access method* also known as *relative access method*. A fixed-length logical record that allows the program to read and write record rapidly. in no particular order. The direct access is based on the disk model of a file since disk allows random access to any file block. For direct access, the file is viewed as a numbered sequence of block or record. Thus, we may read block 14 then block 59, and then we can write block 17. There is no restriction on the order of reading and writing for a direct access file.

A block number provided by the user to the operating system is normally a *relative block number*, the first relative block of the file is 0 and then 1 and so on.

- **3.Index sequential method –**
  It is the other method of accessing a file that is built on the top of the sequential access method. These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks. To find a record in the file, we first search the index, and then by the help of pointer we access the file directly.
- **Key points:**
- It is built on top of Sequential access.
- It control the pointer by using index

4.Relative Record Access –

Relative record access is a file access method used in operating systems where records are accessed relative to the current position of the file pointer. In this method, records are located based on their position relative to the current record, rather than by a specific address or key value.

Key Points of Relative Record Access:

- Relative record access is a random access method that allows records to be accessed based on their position relative to the current record.
- This method is efficient for accessing individual records but may not be suitable for files that require frequent updates or random access to specific records.
- Relative record access requires fixed-length records and may not be flexible enough for some applications.
- This method is useful for processing records in a specific order or for files that are accessed sequentially.

- **5.Content Addressable Access-**
- Content-addressable access (CAA) is a file access method used in operating systems that allows records or blocks to be accessed based on their content rather than their address. In this method, a hash function is used to calculate a unique key for each record or block, and the system can access any record or block by specifying its key.

Keys in Content-Addressable Access:

- Unique: Each record or block has a unique key that is generated using a hash function.
- Calculated based on content: The key is calculated based on the content of the record or block, rather than its location or address.

Data Integrity Protection

The overall precision, completeness, and continuity of data is known as data integrity. Data integrity also applies to the data's protection and security in terms of regulatory enforcement. It is kept up to date by a set of procedures, guidelines, and specifications that were put in place during the design phase.

- Data protection and data quality are often confused with data integrity, but the two terms have different meanings.
- Data integrity also ensures that the information is protected from outside influences.

Data security involves protecting data from unauthorized access and preventing data from being corrupted or stolen. Data integrity is typically a benefit of data security but only refers to data accuracy and validity rather than data protection.

A physical or cryptographic means of providing assurance that information has not been altered in an unauthorized manner since it was created, transmitted, or stored is data integrity protection.

Assignment 4

1. Explain File organization
2. Define physical file system in detail
3. Discuss different data access techniques
4. Explain data integrity protection
5. Write case study of NTFS file system.

Thank you!