📘 MODULE 8: FILE HANDLING (PYTHON)

---

1. Introduction to File Handling

File handling in Python allows us to **create, read, write, update, and delete files**.
Files are used to **store data permanently**, unlike variables which store data temporarily in memory.

Python provides **built-in functions and modules** to perform file operations easily and efficiently.

---

2. File I/O Operations

File I/O (Input / Output) operations include:

- Opening a file
- Reading data
- Writing data
- Appending data
- Closing a file

A file must be **opened before performing any operation**.

---

3. Opening a File

Syntax:

open(filename, mode)

- Default mode is **read (r)**
- If file does not exist in read mode → error

---

4. File Opening Modes

Mode	Description
r	Read (default mode)
w	Write (overwrites existing content)
a	Append (adds data without deleting old data)
x	Create new file
r+	Read and write
t	Text mode (default)

Mode	Description
b	Binary mode

---

5. Writing to a File

Write Mode (w)

- Deletes existing content
- Creates file if not present

```
with open("data.txt", "w") as f:
    f.write("Hello Python\n")
```

Append Mode (a)

- Adds data at the end
- Keeps old data safe

```
with open("data.txt", "a") as f:
    f.write("New line added\n")
```

---

6. Reading from a File

Methods for Reading

Method	Description
read()	Reads entire file
readline()	Reads one line
readlines()	Reads all lines into a list

Example:

```
with open("data.txt", "r") as f:
    print(f.read())
```

---

7. Closing a File

- Files must be closed to **free system resources**
- .close() closes the file manually
- with statement closes file **automatically**

file.close()

✓ Recommended: **use with statement**

---

8. File Handling Using with Statement

The with statement:

- Automatically closes file
- Prevents data loss
- Makes code clean and safe

```
with open("data.txt", "r") as f:
    print(f.read())
```

---

9. Cursor Positioning & File Control

seek(n)

Moves file cursor to **nth byte**

```
f.seek(5)
```

---

tell()

Returns current cursor position

```
print(f.tell())
```

---

10. Truncating a File

truncate(n)

Cuts file to **n bytes**

```
with open("data.txt", "w") as f:
    f.write("Hello World")
    f.truncate(5)
```

Result:

Hello

---

11. Binary File Handling

Binary files store **non-text data** like images, audio, video.

Modes:

- rb → Read binary
- wb → Write binary

Example (Image Copy):

```
with open("image.jpg", "rb") as f:
    data = f.read()


with open("copy.jpg", "wb") as f:
    f.write(data)
```

---

12. Text Files vs CSV Files

Text Files

- Store plain text
- Extension: .txt

CSV Files (Comma Separated Values)

- Store data in rows and columns
- Used in Excel, reports, databases

---

13. CSV File Handling

Writing CSV File

```
import csv


with open("data.csv", "w", newline="") as f:
    writer = csv.writer(f)
    writer.writerow(["Name", "Age"])
    writer.writerow(["Mayank", 14])
```

Reading CSV File

```
import csv
```

```
with open("data.csv", "r") as f:
    reader = csv.reader(f)
    for row in reader:
        print(row)
```

---

14. Log File Creation

What is a Log File?

A **log file** records:

- Program activity
- Errors
- User actions

Extension: .log

Creating a Log File

```
with open("app.log", "a") as log:
    log.write("Application started\n")
```

Logging Errors

```
try:
    x = 10 / 0
except:
    with open("error.log", "a") as log:
        log.write("Division by zero error\n")
```

---

15. File Checking, Copying & Deleting

Checking File Existence

```
import os

if os.path.exists("data.txt"):
    print("File exists")
```

Copying a File

```
import shutil

shutil.copy("data.txt", "backup.txt")
```

Deleting a File

```
os.remove("backup.txt")
```

---

16. Exception Handling in File Operations

Exception handling prevents program crashes.

```
try:
    with open("missing.txt", "r") as f:
        print(f.read())
except FileNotFoundError:
    print("File not found")
```

---

17. Real-Time File Handling Projects

1 Student Record System

- Store student name, marks in file
- Read records anytime

---

2 Attendance Management System

- Save daily attendance
- Append date and student name

---

3 Login History Logger

- Record login time in .log file

---

4 CSV-Based Report System

- Store sales data
- Generate monthly reports

⭐ One-Line Exam Answers

- **File handling**: Managing files using Python
- **Text file**: Stores plain text
- **CSV file**: Stores tabular data
- **with statement**: Auto file closing
- **Log file**: Program activity record
- **Binary file**: Stores non-text data

⭐ Quick Revision Table

Topic	Key Point
File I/O	Read & write files
Modes	r, w, a, b
with	Safe file handling
CSV	Table data
Logs	Error & activity
seek/tell	Cursor control
truncate	Cut file size