🚀 21-Day Python Challenge

Day 12 – File Handling

🎯 Day 12 Learning Objectives

By the end of today's class, you will be able to:

- Understand what a **File** is.
- Create and open a file.
- Write data into a file.
- Read data from a file.
- Append (add) new data to an existing file.
- Understand why file handling is important.
- Build simple real-world file handling programs.

📓 What is File Handling?

A **File** is used to store data permanently on a computer.

Normally, when a Python program ends, all the data stored in variables is **lost**.

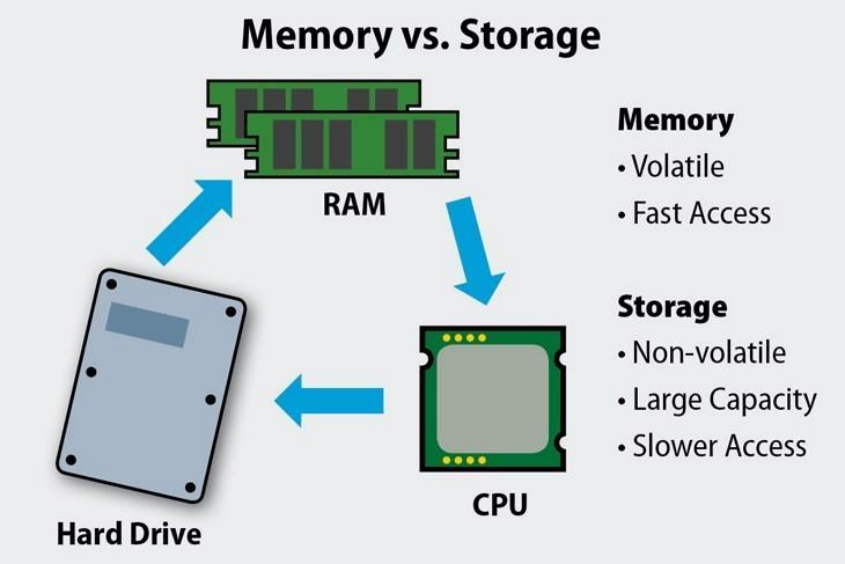
Example:

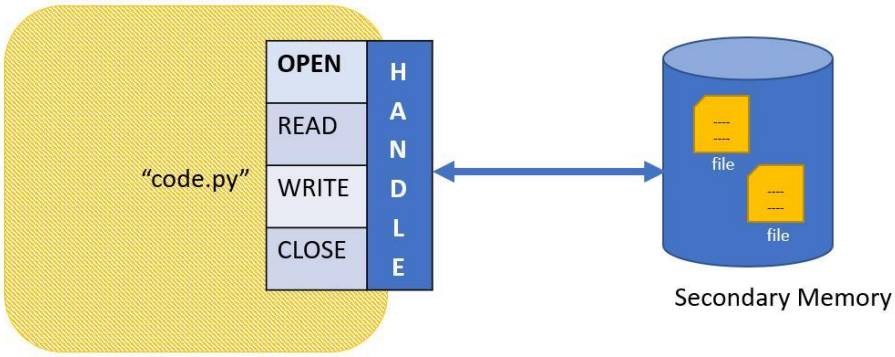
```
name = "Rahul"
marks = 95
```

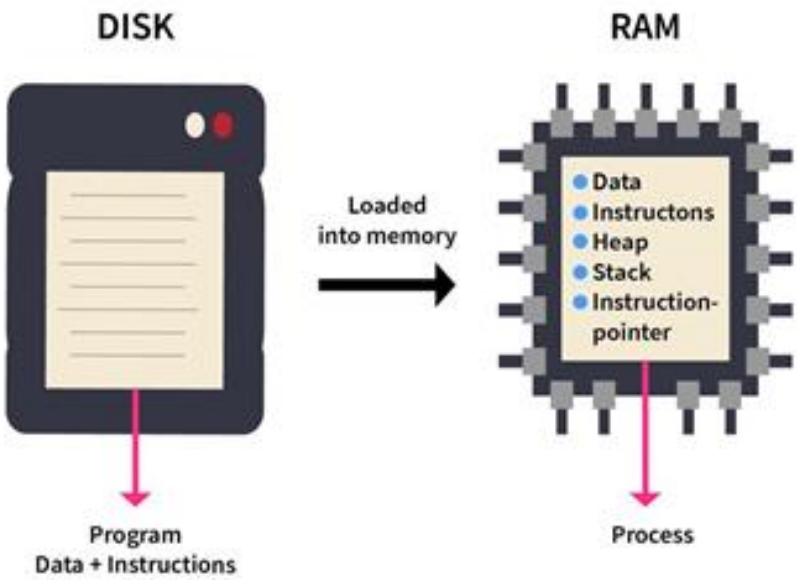
When the program closes, these values disappear from memory.

If we save them in a file, they remain even after the computer is turned off.

📷 Picture 1 – Memory vs File Storage







Explanation

```
Program Starts

↓

Variables stored in RAM

↓

Program Ends

↓

Data Lost ✕

-------------------------

Write to File

↓

Hard Disk

↓

Data Saved Permanently ☑
```

Why Do We Need Files?

Files are used to store:

- Student Records
- Employee Details
- Attendance
- Marks
- Bills
- Reports
- Login Information

Almost every software application stores data in files or databases.

Opening a File

Python uses the **open()** function.

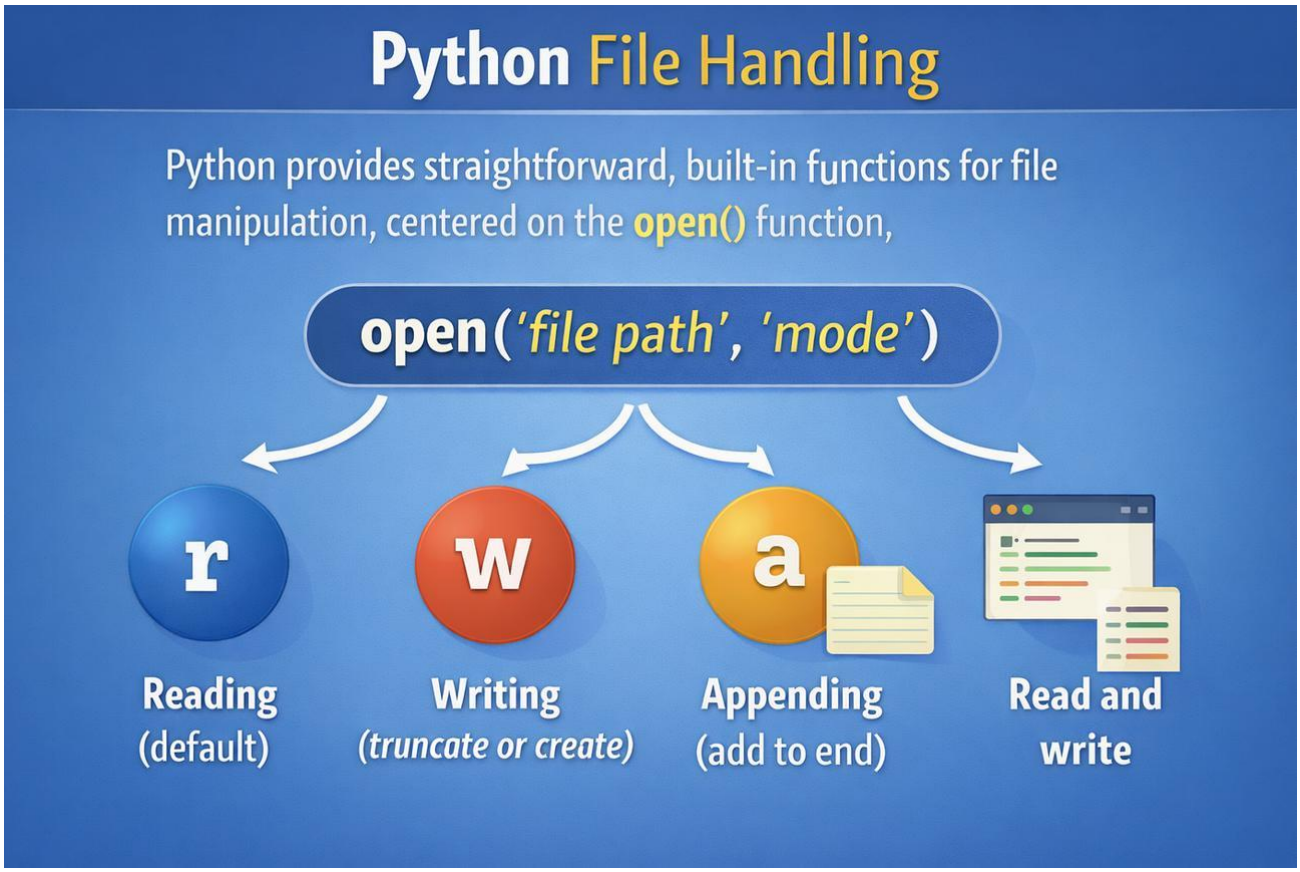
Syntax

```
file = open("student.txt", "mode")
```

File Modes

Mode	Meaning
`"r"`	Read
`"w"`	Write
`"a"`	Append

📷 Picture 2 – File Modes

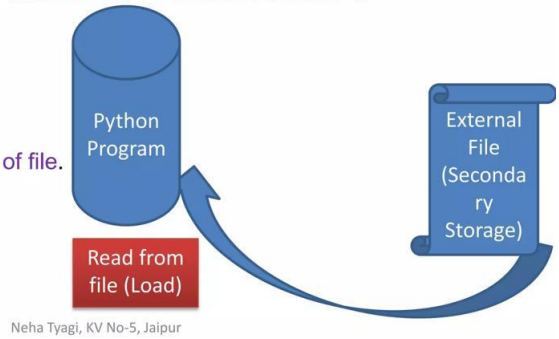


Mode	Description
r	Opens a file for reading (default mode).
w	Opens a file for writing. Creates a new file if it does not exist, or truncates the file if it exists.
a	Opens a file for appending. Data is added to the end of the file without overwriting it
r+	Opens a file for both reading and writing.
w+	Opens a file for both writing and reading. Overwrites the file if it exists.
a+	Opens a file for both appending and reading.
b	Binary mode. Add this to other modes (e.g., rb, wb) to work with binary files.

Opening & Closing Files

- We need a *file variable* or *file handle* to work with files in Python.
- This file object can be created by using open( ) function or file( ) function.
- Open( ) function creates a file object, which is used later to access the file using the functions related to file manipulation.
- Its syntax is following -

  <file_object>=open(<file_name>,<access_mode>)
- File accessing modes -
  - read(r): To read the file
  - write(w): to write to the file
  - append(a): to Write at the end of file.



Neha Tyagi, KV No-5, Jaipur

```
File

↓

Read (r)

↓

Write (w)

↓

Append (a)
```

Writing Data to a File

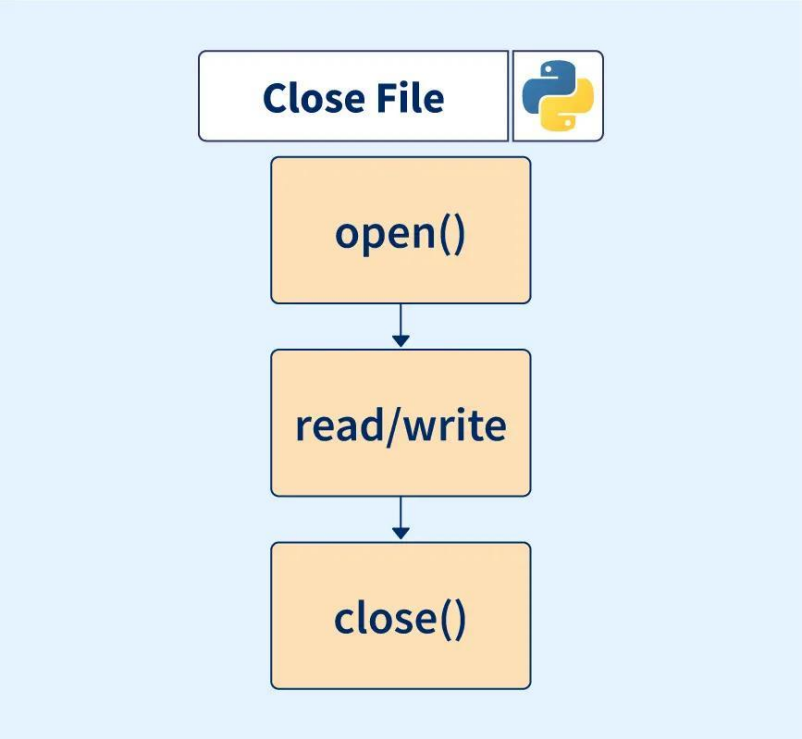
Example 1

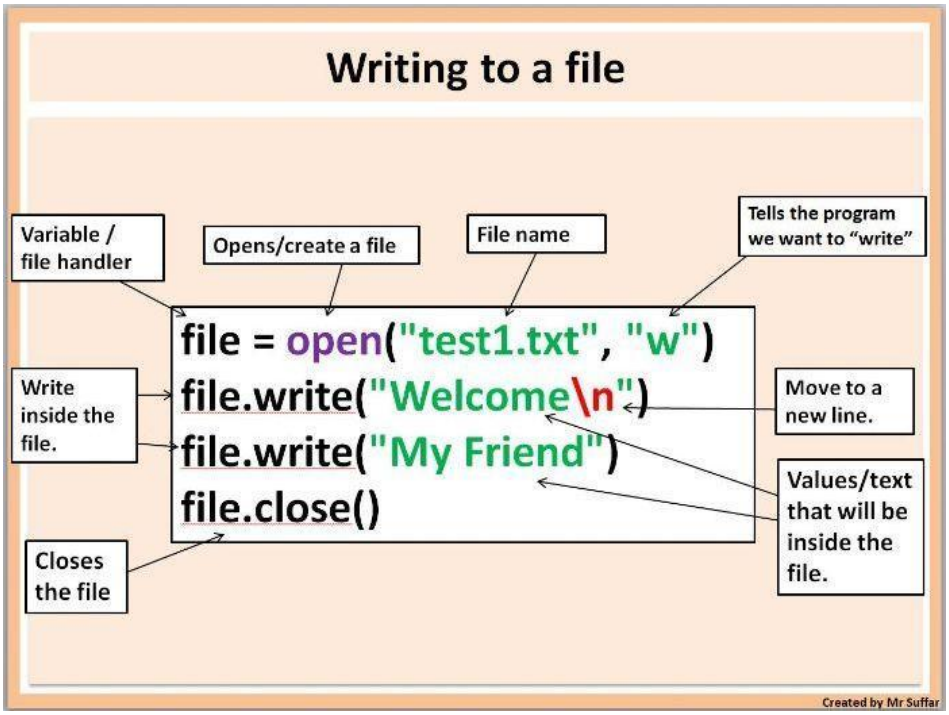
```
file = open("student.txt", "w")

file.write("Welcome to Python")

file.close()

print("Data Written Successfully")
```

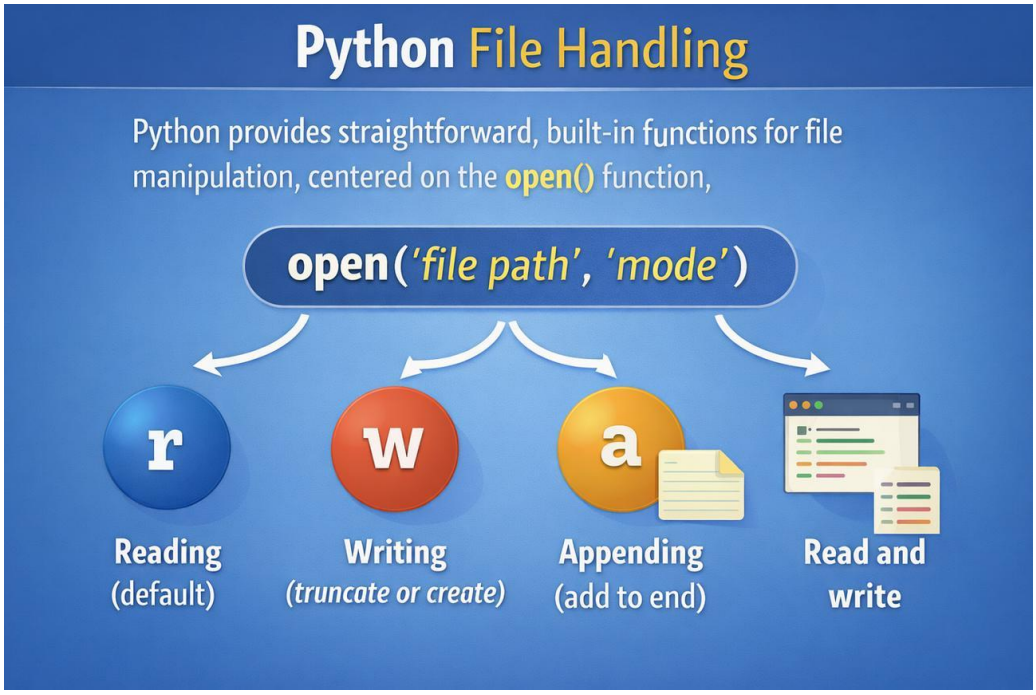
Explanation

- `"w"` → Creates a new file (or overwrites an existing one).
- `write()` → Stores the text.
- `close()` → Closes the file safely.

📷 Picture 3 – Writing to a File








```
Open File
↓
Write Data
↓
Save
↓
Close File
```

Reading Data from a File

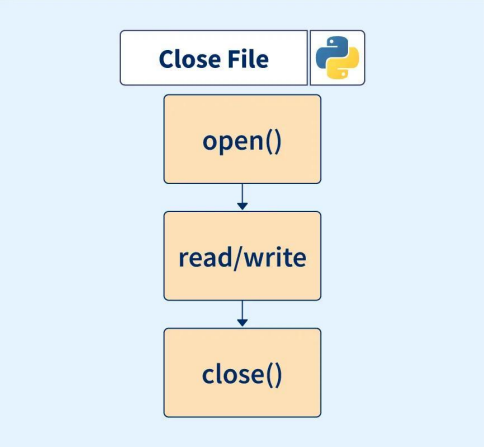
Example 2

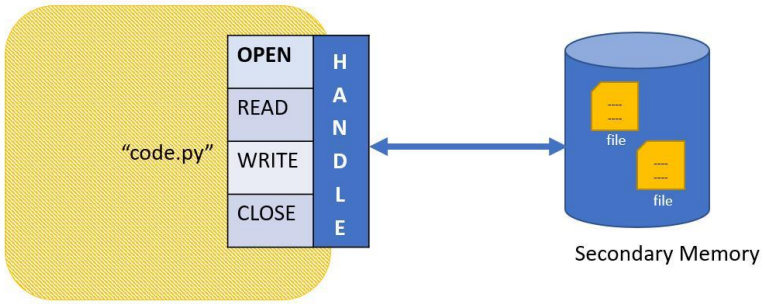
```
file = open("student.txt", "r")
data = file.read()
print(data)
file.close()
```

Output

```
Welcome to Python
```

📷 Picture 4 – Reading a File






```
Open File
↓
Read Data
↓
Display on Screen
↓
Close File
```

📓 Appending Data

Appending means **adding new data without deleting the old data**.

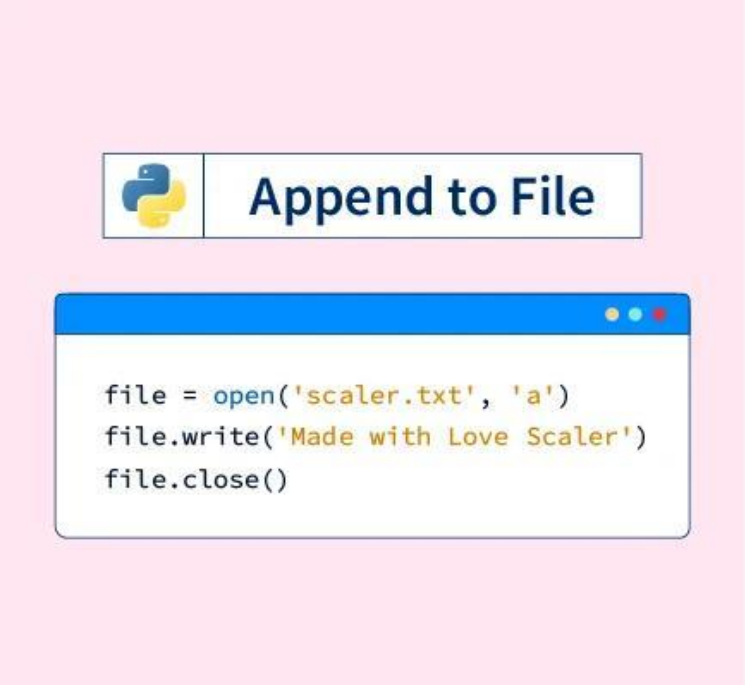
Example 3

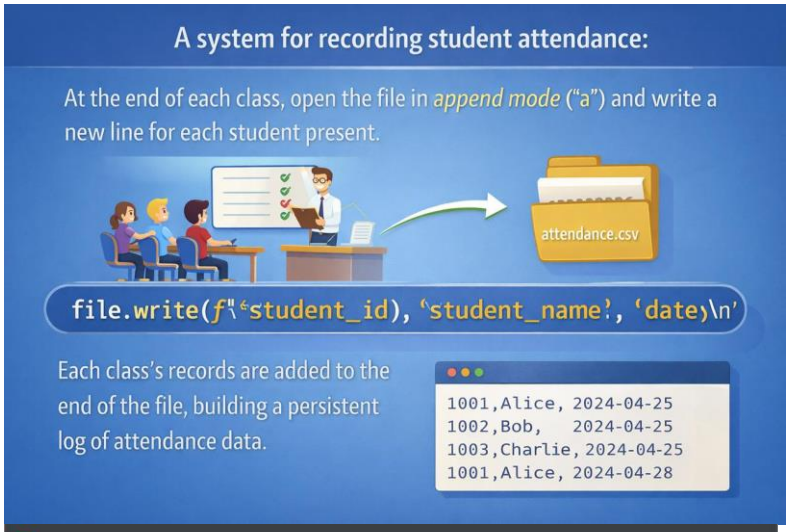
```
file = open("student.txt", "a")

file.write("\nWelcome to GK InfoTech")

file.close()

print("Data Added Successfully")
```

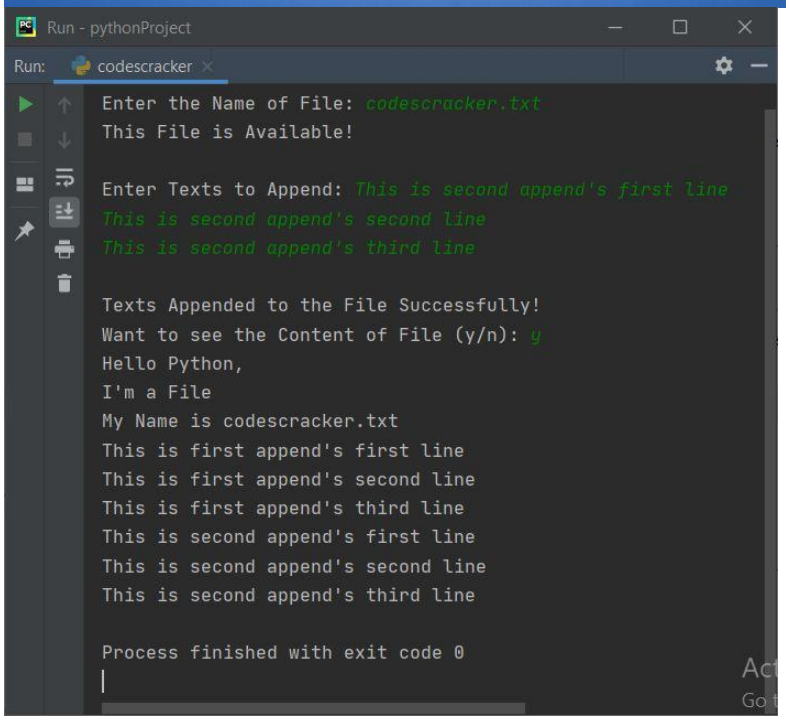
Output inside the file:

```
Welcome to Python
Welcome to GK InfoTech
```

📷 Picture 5 – Append Data







Before

Python

↓

Append

↓

After

Python

GK InfoTech

Real-Time Program 1

Save Student Name

```
file = open("students.txt", "w")

name = input("Enter Student Name : ")

file.write(name)

file.close()

print("Student Saved Successfully")
```

Real-Time Program 2

Read Student Name

```
file = open("students.txt", "r")

print(file.read())

file.close()
```

Real-Time Program 3

Add More Students

```
file = open("students.txt", "a")

name = input("Enter Another Student : ")

file.write("\n" + name)

file.close()

print("Student Added Successfully")
```

Real-Time Program 4

Student Attendance Register

```
file = open("attendance.txt", "a")

name = input("Enter Student Name : ")

file.write(name + "\n")

file.close()

print("Attendance Saved")
```

Real-Time Program 5

Save Marks

```
file = open("marks.txt", "w")

marks = input("Enter Marks : ")

file.write(marks)

file.close()

print("Marks Saved")
```

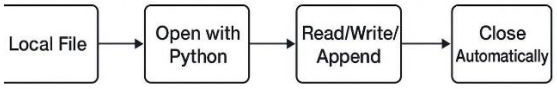
📓 Better Way – Using `with open()`

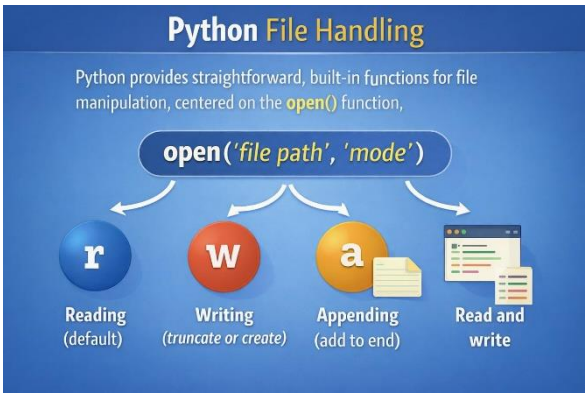
Python provides a safer way to work with files.

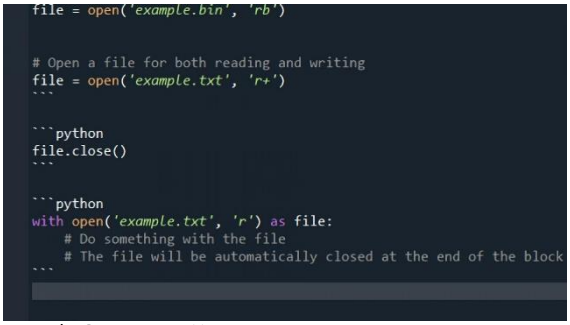
```
with open("student.txt", "w") as file:
    file.write("Welcome to Python")
```

You **do not need to call `close()`** because Python closes the file automatically.

📷 Picture 6 – `with open()`







```
file = open('example.bin', 'rb')


# Open a file for both reading and writing
file = open('example.txt', 'r+')
```

```python
file.close()
```

```python
with open('example.txt', 'r') as file:
    # Do something with the file
    # The file will be automatically closed at the end of the block
```

```
with open()

↓

Open File

↓

Do Work

↓

Automatically Close File
```

Mini Project : Student Record Management

```
while True:

    print("\n1. Add Student")
    print("2. View Students")
    print("3. Exit")

    choice = input("Enter Choice : ")

    if choice == "1":

        with open("students.txt", "a") as file:
            name = input("Enter Student Name : ")
            file.write(name + "\n")

        print("Student Saved Successfully")

    elif choice == "2":

        with open("students.txt", "r") as file:
            print("\nStudent List")
            print(file.read())

    elif choice == "3":
        print("Thank You")
        break

    else:
        print("Invalid Choice")
```

Practice Programs

1. Create a file and write your name.
2. Read data from a file.
3. Append your college name.
4. Store five student names.
5. Read all student names.
6. Save marks in a file.
7. Create an attendance register.

Assignment

Write programs to:

- Create a file named `student.txt`.
- Store your name in the file.
- Read the file.
- Append your city name.
- Store five student names.
- Create a marks file.
- Display all marks.

Interview Questions

1. What is File Handling?
2. Why do we use files?
3. What does `open()` do?
4. What is `"r"` mode?
5. What is `"w"` mode?
6. What is `"a"` mode?
7. What is the difference between Write and Append?
8. Why do we use `close()`?
9. What is the advantage of `with open()`?
10. Name two real-world applications of file handling.

🏆 Day 12 Challenge : Student Information File

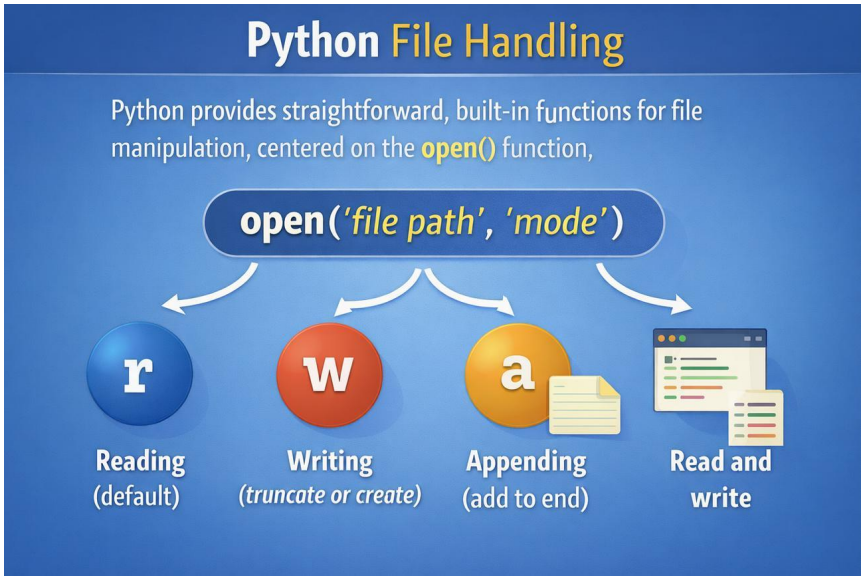
Write a Python program that:

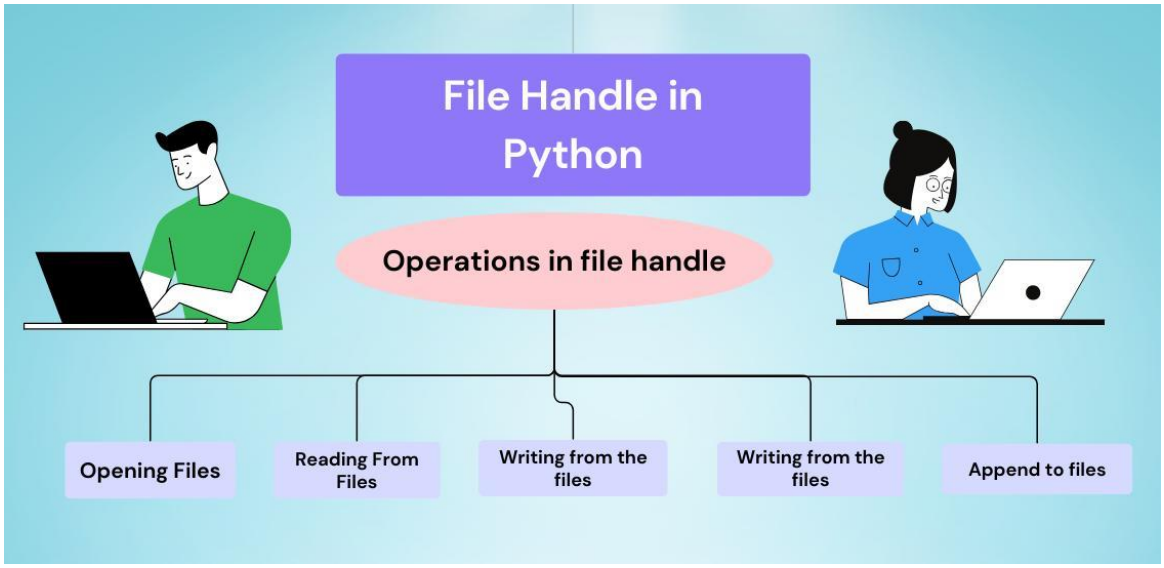
- Accepts:
  - Student Name
  - Roll Number
  - Course
  - Marks
- Saves the information into a file called `student_details.txt`.
- Reads the file and displays all the saved information.

Sample Output

```
===== STUDENT DETAILS =====
Name         : Rahul
Roll Number  : 101
Course       : Python
Marks        : 92

Student information saved successfully.
```

Picture 7 – Complete File Handling Process






```
Program Starts
      |
      ▼
Open File
      |
      ▼
Choose Mode
(r / w / a)
      |
      ▼
Read or Write Data
      |
      ▼
Save Changes
      |
      ▼
Close File
      |
      ▼
Program Ends
```

📊 Quick Revision Table

Function	Purpose	Example
`open()`	Opens a file	`open("data.txt", "r")`
`read()`	Reads file contents	`file.read()`
`write()`	Writes data to a file	`file.write("Hello")`
`close()`	Closes the file	`file.close()`
`with open()`	Opens and closes automatically	`with open("data.txt") as file:`

🌟 Day 12 Success Message

Congratulations! 🎉

Today you learned one of the most practical topics in Python—**File Handling**. You explored:

- ✅ Reading files (`"r"`)
- ✅ Writing files (`"w"`)
- ✅ Appending data (`"a"`)
- ✅ Using `with open()` for safer file operations

These skills are used in **student management systems, attendance software, billing applications, banking systems, inventory management, and almost every real-world application**.

"Variables remember data only while the program is running. Files remember data even after the program closes. Learning file handling is your first step toward building real-world software."

🚀 **Next Lesson (Day 13): Exception Handling – `try`, `except`, `finally`, and handling common runtime errors.**