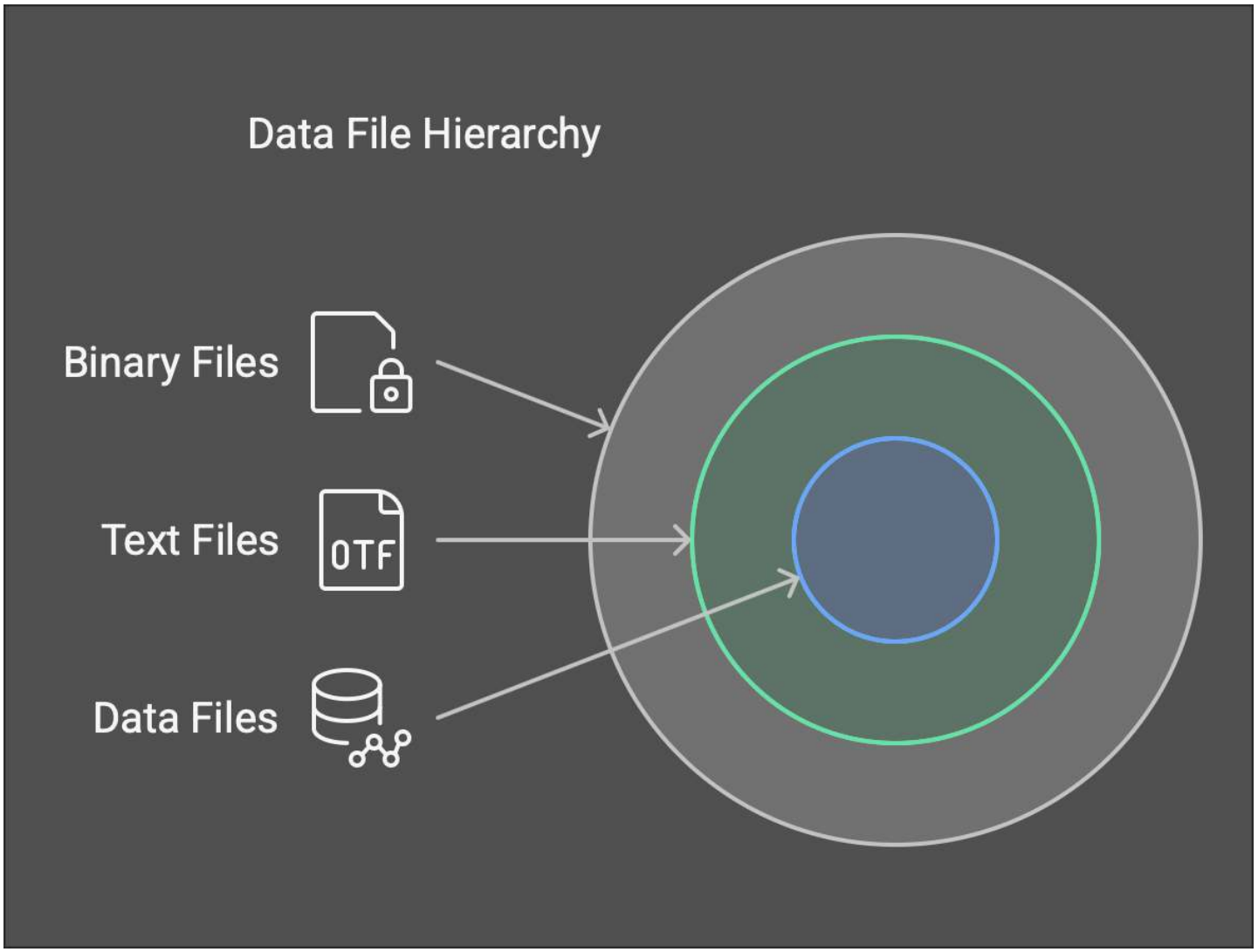
File Handling-in-Python

Phani Rajendra

File Handling-in-Python

A.Introduction: File Operations are an important activity in Python ,before we proceed we should know how many types of files are available or what all types of files can be handled in Python.



Types of Files

Computers store every file as a collection of 0s and1s i.e.,in binary form.Therefore, every file is basically just a series of bytes stored one after the other .There are mainly two types of data files text file and binary file. A text file consists of human readable characters, which can be opened by any text editor. On the other hand, binary files are made up of non-human readable characters and symbols, which require specific programs to access its contents.

File Handling-in-Python

Text file

A text file can be understood as a sequence of characters consisting of alphabets, numbers and other special symbols. Files with extensions like .txt, .py, .csv, etc. are some examples of text files. When we open a text file using a text editor(e.g.,Notepad), we see several lines of text.

Operation on Text File

We can perform the following operations on files in Python,

1.Open()

2.Writing()

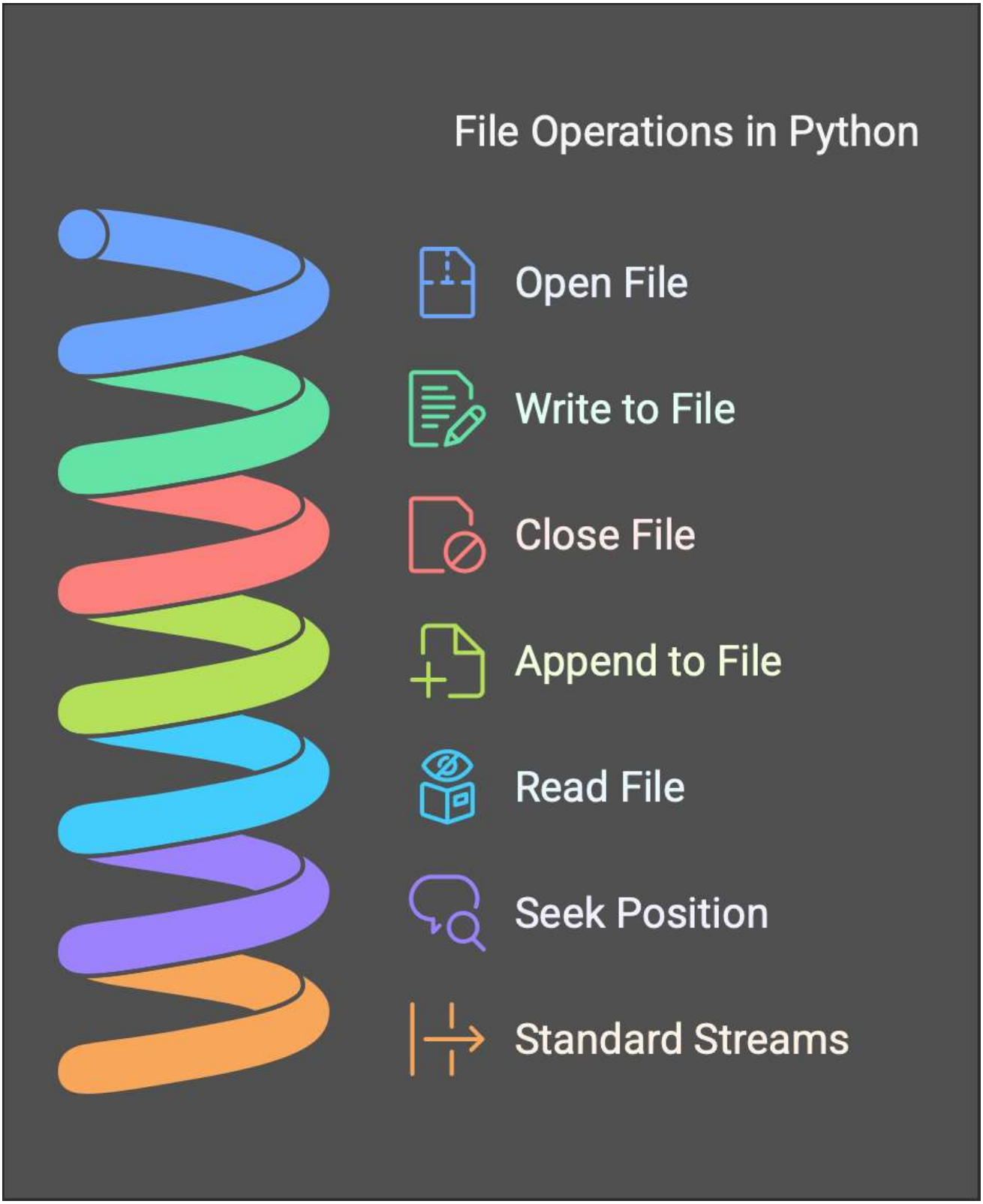
3.Close()

4.Append

5.Reading()

6.Seek()

7.StandardInput / OutPut and Error streams



Opening File:

To open a file, we need to use the built-in open function. The open function returns a file object

that contains methods and attributes to perform various operations on the file.

Syntax:

file_object =open("filename", 'mode')

filename: gives name of the file that the file object has opened.

mode: attribute of a file object tells you which mode a file was opened in.

Example:

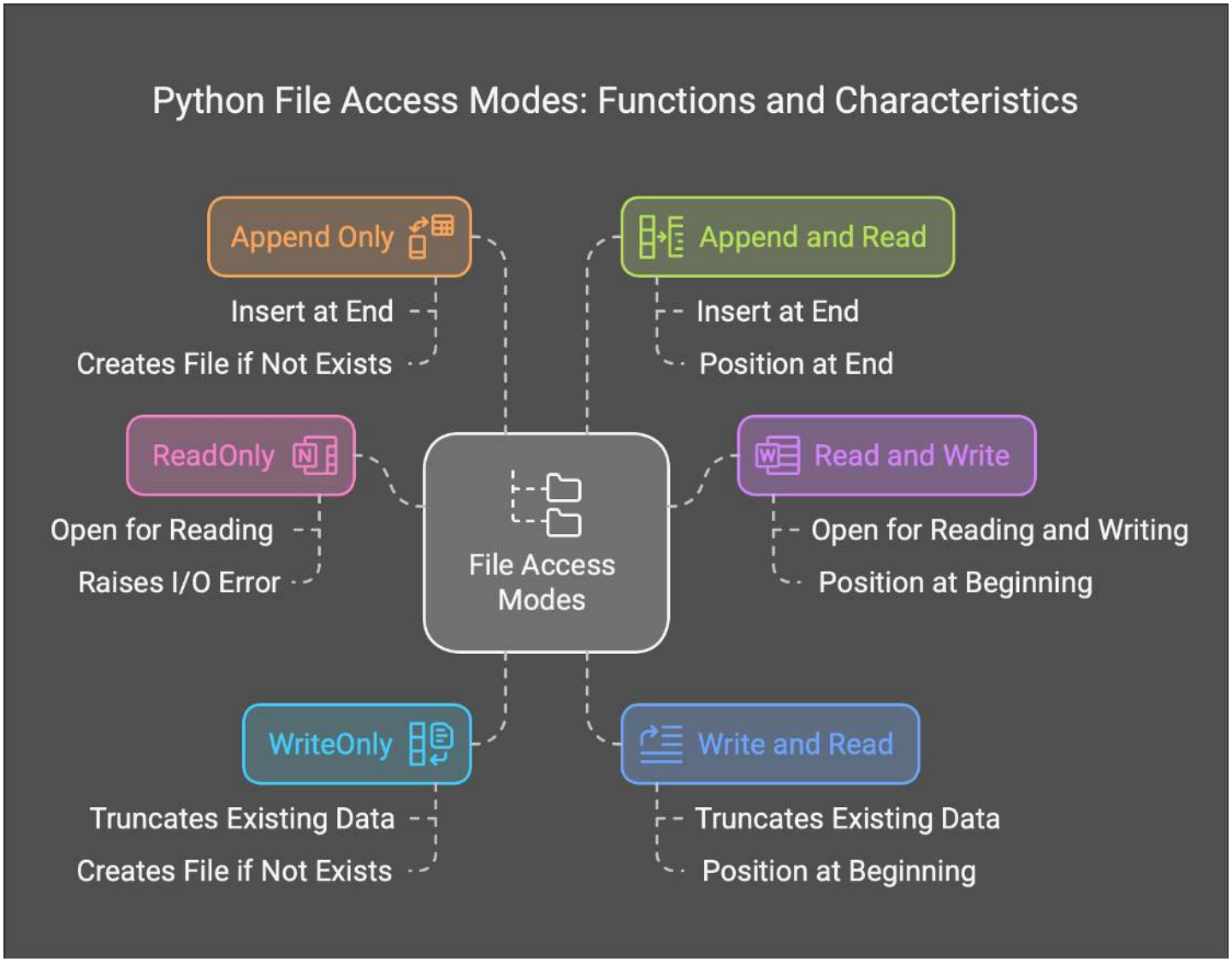
f =open("demofile.txt","w')

There are 6 access modes in python text file, they are as following

1. **ReadOnly('r')**: Open text file for reading. The handle is positioned at the beginning of the file. If

   the file does not exists, it raises I/O error. This is also the default mode in which file is opened.

2. **Read and Write('r+')**: Open the file for reading and writing. The handle is positioned at the beginning of the file.
3. **WriteOnly('w')**: Open the file for writing. For existing file, the data is truncated and over written. The handle is positioned at the beginning of the file. Creates the file if the file does not exists.
4. **Write andRead('w+')**: Open the file for reading and writing. For existing file, data is truncated and overwritten. The handle is positioned at the beginning of the file.
5. **Append Only('a')**: Open the file for writing. The file is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.
6. **Append and Read('a+')**: Open the file for reading and writing. The file is created if it does not exist. The handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data.



Examples:

file=open('demos.txt",'w')

file.write("ShineBlue , welcome"\n")

file.write("Welcome to AI Class")

file.close()

2. Writing onto files

To write strings into text data file we can use write()or writelines() function.For writing we can open a

file in ‘w’or ‘w+’or ‘r+’mode.

Example: ***file.write("welcome to AI Learning")***

3.Closing Files

An open file is closed by calling the close()function of its object. Closing of files is important in python, files are automatically closed at the end of the program but it is good practice to get into the habit of closing files.

Example: **file.close()**

4.Appending a file

If we want to write into the file retaining the old data, then we should open the file in an “a+”or append mode. A file opened in append mode retains its previous data while allowing you to add newer data into it.