LIVE CLASS

atsheet

E HANDLING

PYTHON

NTCC

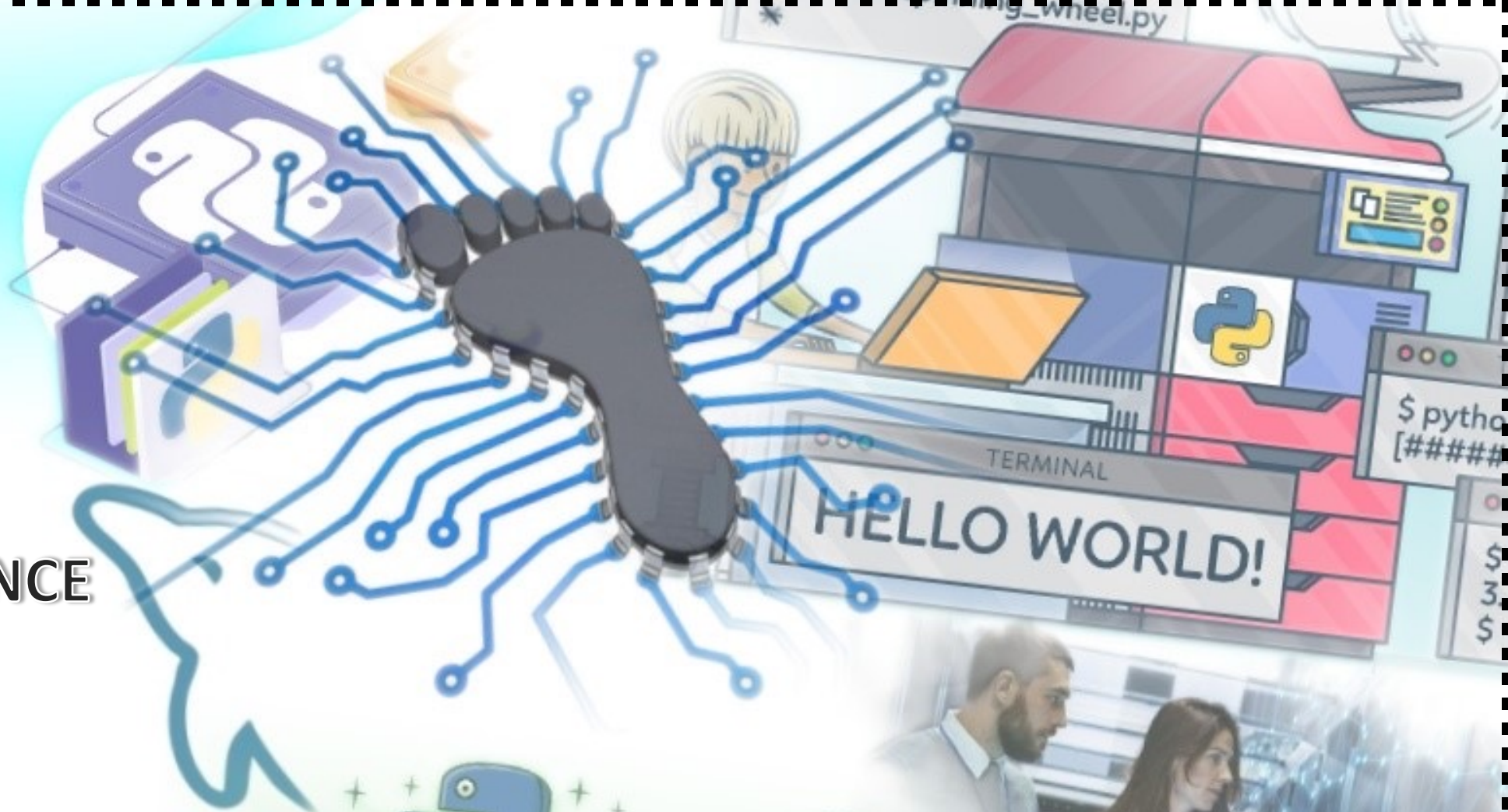
NIK TECH COACHING CENTRE

COMPUTER SCIENCE

CBSE CLASS 12TH

2022

BY ER NIKHIL SIR



Txt → Text Document

↳ Python

f = open('file ka naam', 'mode')

↙

Variable / file Handle

Mode → (3) → (3)

↳ read 'r' ↳ wt → write new
at → append

↳ write 'w' overwrite

↳ append 'a' ↳ old data erase

- read ✓ Pass ✓ => Mode
- readline ✓ line (1)
- readlines list [ ] Pass

Write/append

- write(str)
- writelines()

(2)

→ list/tuple

Text ← Human → Binary

r

w

a

r+

w+

a+

rb

wb

ab

r+b

w+b

a+b

Pickle

dump(write)

load

(read)

LIVE CLASS

Data File Handling

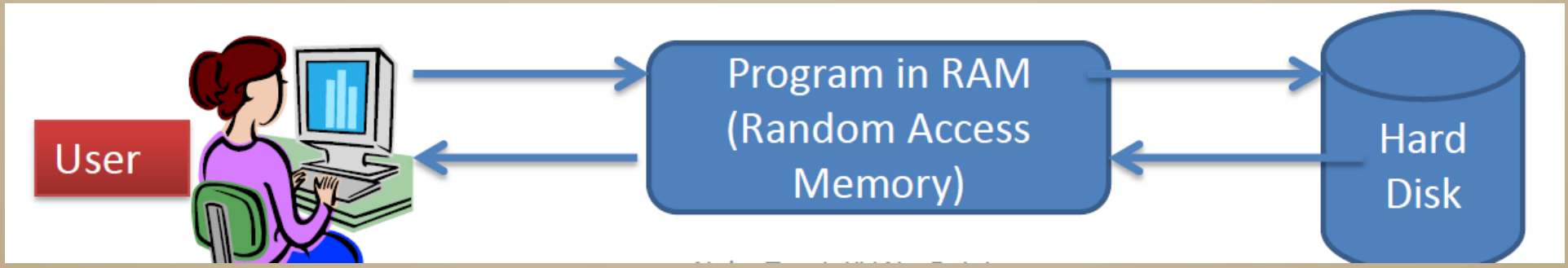
We have seen yet only the transient programs. The programs which run for a short period of time and give some output and after that their data is disappeared. And when we again run those programs then we have to use new data.

This is because the data is entered in primary memory which is temporary memory and its data is volatile.

Those programs which are persistent i.e. they are always in running or run for a long time then their data is stored in permanent storage (e.g. harddisk) . If the program is closed or restarted then the data used will be retrieved.

For this purpose the program should have the capability to read or write the text files or data files. These files can be saved in permanent storage.

The meaning of File I/O (input-output) is to transfer the data from Primary memory to secondary memory and vice-versa.



Why the Files are used?

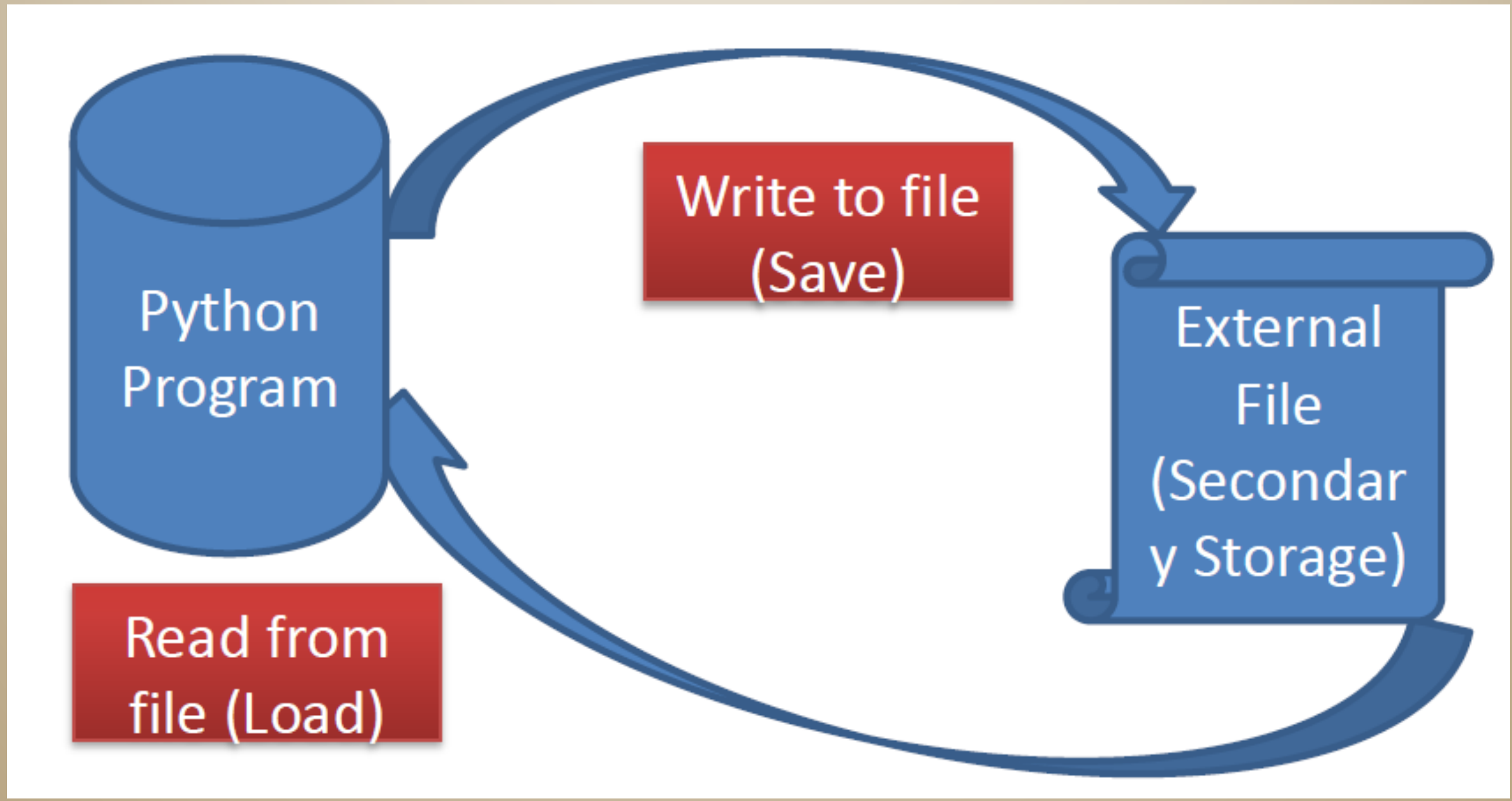
The data stored with in a file is known as persistent data because this data is permanently stored in the system.

Python provides reading and writing capability of data files.

We save the data in the files for further use.

As you save your data in files using word, excel etc. same thing we can do with python.

A File is a collection of characters in which we can perform read and write functions. And also we can save it in secondary storage.”


Python Program
Write to file (Save)
External File (Secondary Storage)
Read from file (Load)

File handling

Working with data files in pyton programming is called File handling (Data File Handling)

Advantage's

Data is stored permanently

Already existing data can be used.

Large amount of data can be processed easily.

Enter=EOL(End of line)

Data File Operations

Following main operations can be done on files -

1.Opening a file

2.Performing operations

1.READ

2.WRITE etc.

3.Closing The File

Beside above operations there are some more operations can be done on files.-

- Creating of Files
- Traversing of Data
- Appending Data into file.
- Inserting Data into File.
- Deleting Data from File.
- Copying of File.
- Updating Data into File.

Text Files

Data Stored in ASCII or Unicode

Data in human readable form

Execution is slow

Binary Files

Data is stored as it is in memory location

Data is not in human readable form

Execution is fast

UNICODE (For Multilingual Computing)

Unicode is a new universal coding standard adopted by all new platforms. It is promoted by Unicode Consortium which is a non-profit organization. Unicode provides a unique number for every character irrespective of the platform, program and the language. It is a character coding system designed to support the worldwide interchange, processing, and display of the written texts of the diverse languages. A unique number for every character, no matter what the platform, no matter what the program, no matter what the language.

Unicode is the preferred encoding scheme used by XML-based tools and applications.

Significance of UNICODE

- Unicode enables a single software product or a single website to be designed for multiple platforms, languages and countries (no need for re-engineering) which can lead to a significant reduction in cost over the use of legacy character sets.
- Unicode data can be used through many different systems without data corruption.
- Unicode represents a single encoding scheme for all languages and characters.
- Unicode is a common point in the conversion between other character encoding schemes. Since it is a superset of all of the other common character encoding systems, you can convert from one encoding scheme to Unicode, and then from Unicode to the other encoding scheme.
- Unicode is the preferred encoding scheme used by XML-based tools and applications.

File Types -- Full Chatai ke sath

1.Text File: A text file is sequence of line and line is the sequence of characters and this file is saved in a permanent storage device. Although in python default character coding is ASCII but by using constant ‘U’ this can be converted into UNICODE. In Text File each line terminates with a special character which is EOL (End Of Line). These are in human readable form and these can be created using any text editor.

2.Binary File: Binary files are used to store binary data such as images, videos audio etc. Generally numbers are stored in binary files. In binary file, there is no delimiter to end a line. Since they are directly in the form of binary hence there is no need to translate them. That’s why these files are easy and fast in working.

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.

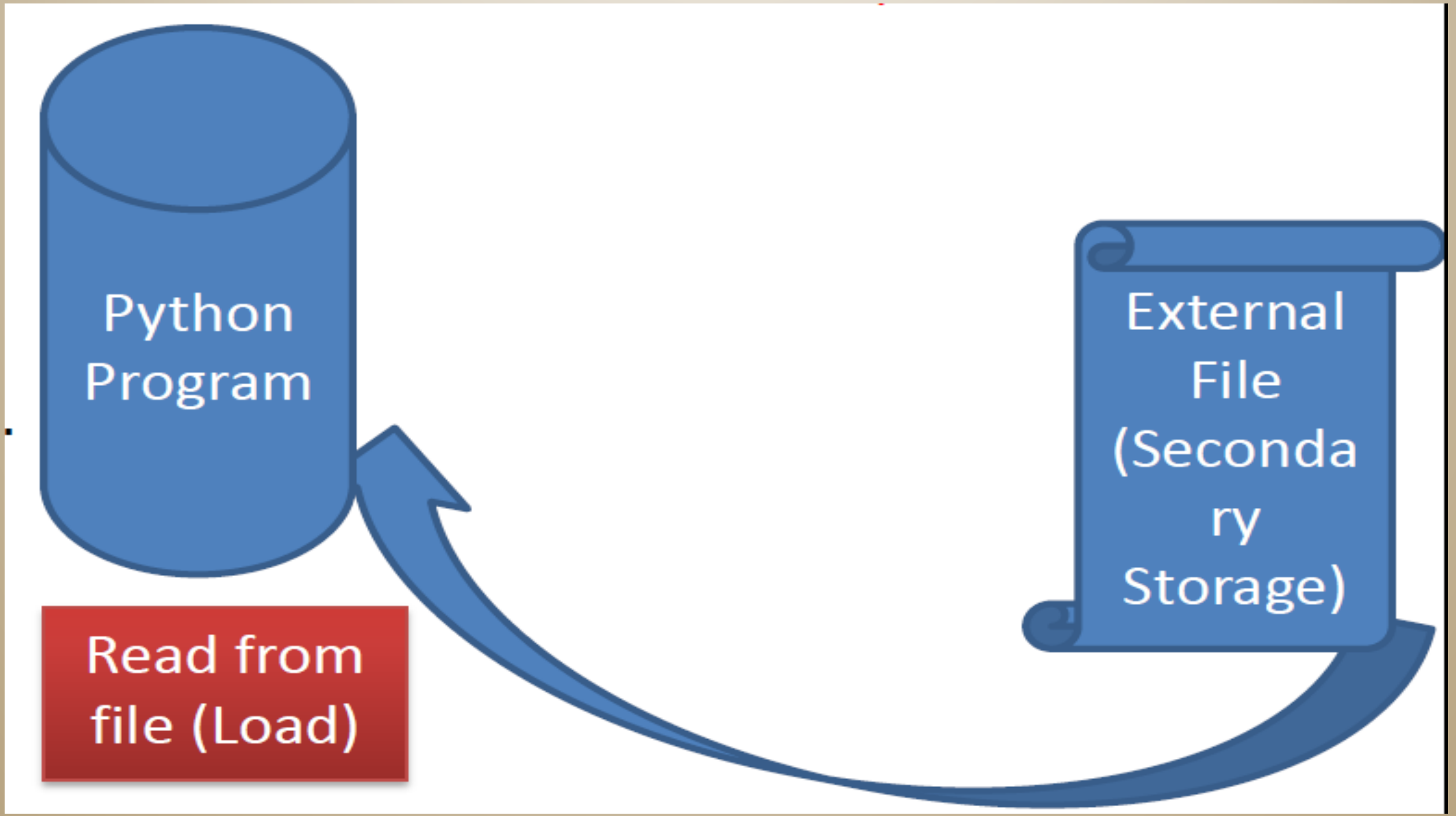
Open( ) function creates a file object, which is used later to access the file using the functions related to file manipulation.

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.


Python Program
External File (Secondary Storage)
Read from file (Load)

Opening file in python :-

Reading mode

File_object=open(Filename\address,mode)

Variable or file handle

Nikhil=open("hello_bhai.txt","r")

~~Nikhil=open(("d:\today\hello_bhai.txt")~~

Ye bhi chalta h but kabhi kabhi /n ya /t bhi kaam krne lag jate h

```
Nikhil=open("d:\\today\\hello_bhai.txt")
```

Yaha double h to koi dikkat nhi aati h

Kisi kisi me khujli hoti h ,me single hi use karunga

Raw string (r) kaam krta h stating me laga dete h to vo batata h ki ab string ka koi bhi special meaning nhi h

```
Nikhil=open( r "d:\today\hello_bhai.txt")
```

Opening & Closing Files. . .

```
>>> f=open("Hello.txt")
>>> print(f)
<_io.TextIOWrapper name='Hello.txt' mode='r' encoding='cp1252'>
```

Opened the File

Here the point is that the file "Hello.txt" which is used here is pre built and stored in the same folder where Python is installed.

```
>>> f=open("Hello.txt")
>>> print(f)
<_io.TextIOWrapper name='Hello.txt' mode='r' encoding='cp1252'>
>>> f.close()
```

The file is closed.

A program describing the functions of file handling.

```
f=open("Hello.txt",'r')
print("File Name : ", f.name)
print("File mode : ", f.mode)
print("Is File Readable? : ", f.readable())
print("Is File Closed? : ", f.closed)
f.close()
print("Is File Closed? : ", f.closed)
```

Output

```
= RESTART: C:/Users/KVBBKS
File Name :  Hello.txt
File mode :  r
Is File Readable? :  True
Is File Closed? :  False
Is File Closed? :  True
```

File Modes

Table 5.1 *File-modes*

Text File Mode	Binary File Mode	Description	Notes
'r'	'rb'	read only	❖ File must exist already, otherwise Python raises I/O error.
'w'	'wb'	write only	❖ If the file does not exist, file is created. ❖ If the file exists, Python will truncate existing data and over-write in the file. So this mode must be used with caution.
'a'	'ab'	append	❖ File is in **write only** mode. ❖ If the file exists, the data in the file is retained and new data being written will be appended to the end. ❖ If the file does not exist, Python will create a new file.
'r+'	'r+b' or 'rb+'	read and write	❖ File must exist otherwise error is raised. ❖ Both reading and writing operations can take place.
'w+'	'w+b' or 'wb+'	write and read	❖ File is created if does not exist. ❖ If file exists, file is truncated (past data is lost). ❖ Both reading and writing operations can take place.
'a+'	'a+b' or 'ab+'	write and read	❖ File is created if does not exist. ❖ If file exists, file's existing data is retained ; new data is appended. ❖ Both reading and writing operations can take place.

Adding a ***'b'*** to *text-file mode* makes it *binary-file mode*.

Practical Karte hai

Reading data from file:-

→Hello bhai \n
→Python sikh lo \n
→Hello chai pilo frds \n
→Garmi h frds coca coca pilo

Hello_bhai.txt

Case-I

```
p=open("Hello_bhai.txt","r")
k=p.read()
print(k)
```

Case -II

```
p=open("hello.txt","r").read()
print(p)
```

BOOK – CODE SNIPPER 1

Reading a file's first 30 bytes and printing it

```
Myfile=open(r"hello_bhai.txt","r")
Str=myfile.read(30)
Print(str)
```

BOOK- CODE SNIPPER 2

Reading n bytes and then reading some more bytes from the last position read

```
myfile=open(r"hello_bhai.txt","r")
str=myfile.read(30)
print(str)
str2=myfile.read(50)
print(str2)
myfiles.close()
```

BOOK- CODE SNIPPER 3

Reading a file's entire content

```
myfile=open(r"hello_bhai.txt","r")
str=myfiles.read()
print(str)
myfile.close()
```

BOOK- CODE SNIPPER 4

Reading a file's first three lines- lines by lines

```
myfile=open(r"hello_bhai.txt","r")
str=myfile.readline()
print=(str,end=' ')
str=myfile.readline()
print(str,end=' ')
str=myfile.readline()
print(str,end=' ')
myfile.close()
```

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS

LIVE CLASS