Advanced Programming-Python (Files and Exceptions)

Presented by

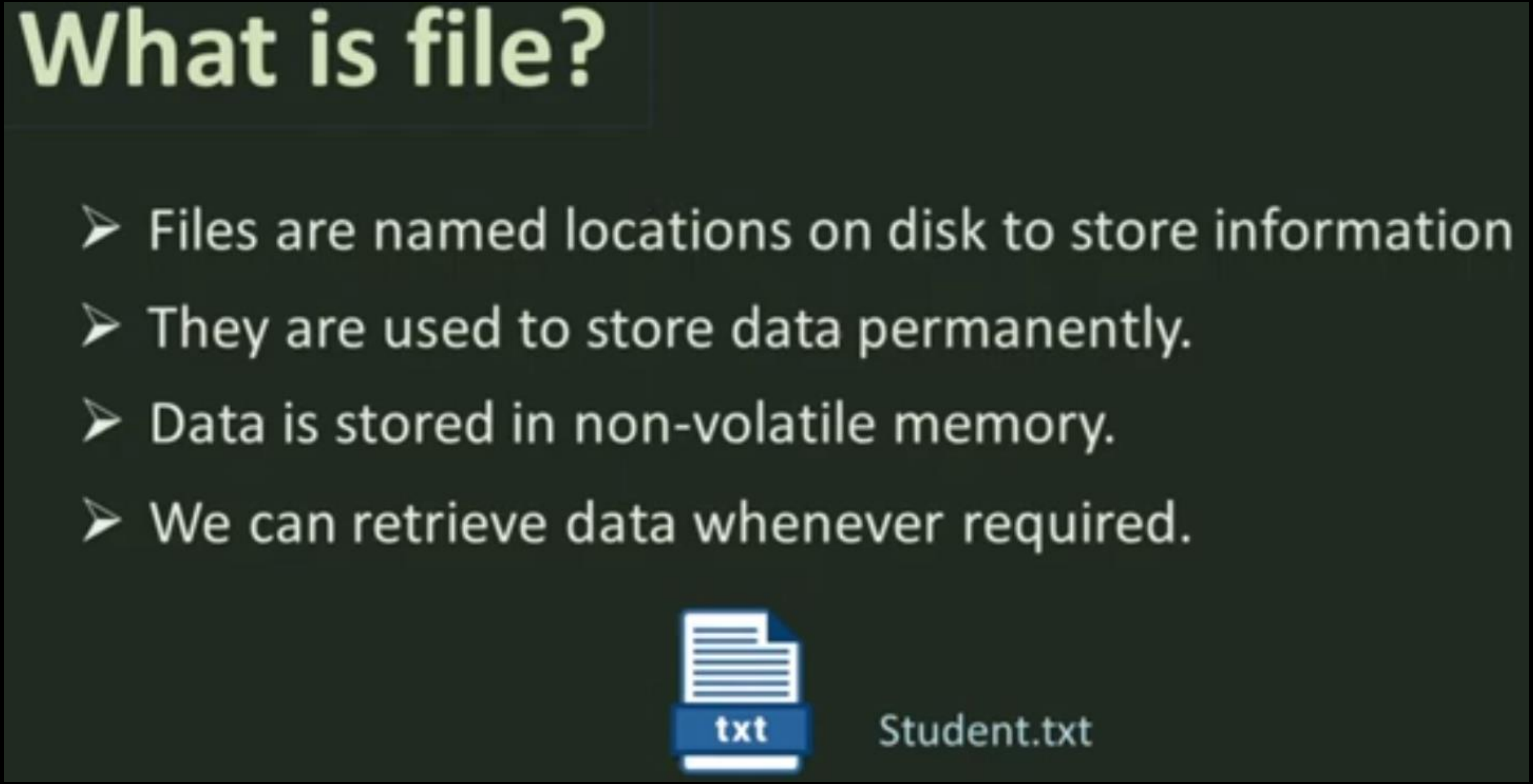
Siddhanta Borah

INTRODUCTION

Files are identified locations on a disk where associated data is stored. Working with files will make your programs fast when analysing masses of data.

Exceptions are special objects that any programming language uses to manage errors that occur when a program is running.

INTRODUCTION



What is file?

- ➢ Files are named locations on disk to store information
- ➢ They are used to store data permanently.
- ➢ Data is stored in non-volatile memory.
- ➢ We can retrieve data whenever required.

txt Student.txt

Types of files?

Text files:-

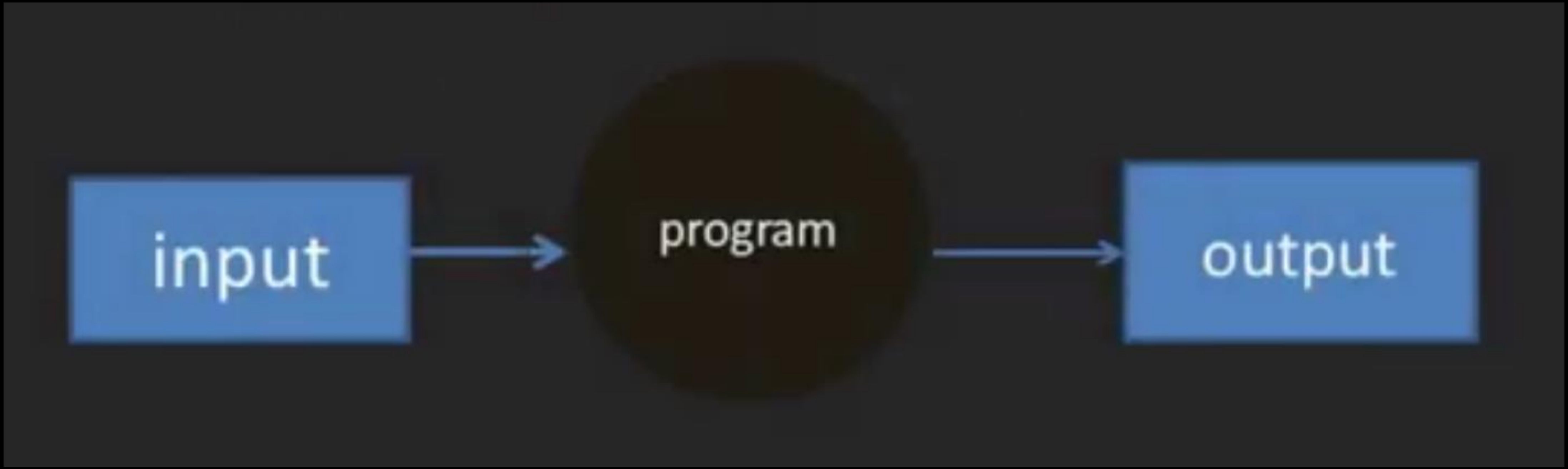
- Stores data in the form of characters. It is used to store data and strings.

Binary files:-

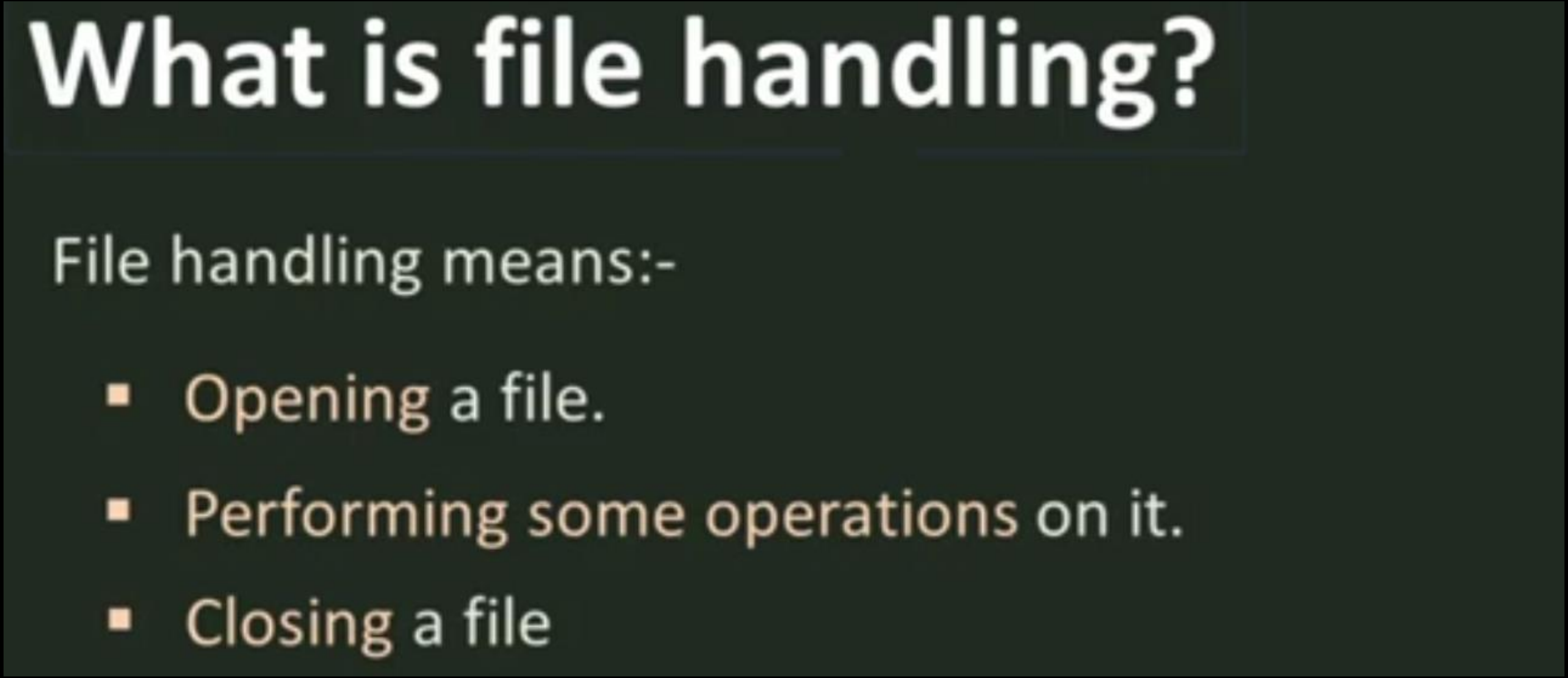
- Stores data in the form of bytes(group of 8 bits)

NEED OF FILE HANDLING

Need of
file handling



Example 1-2



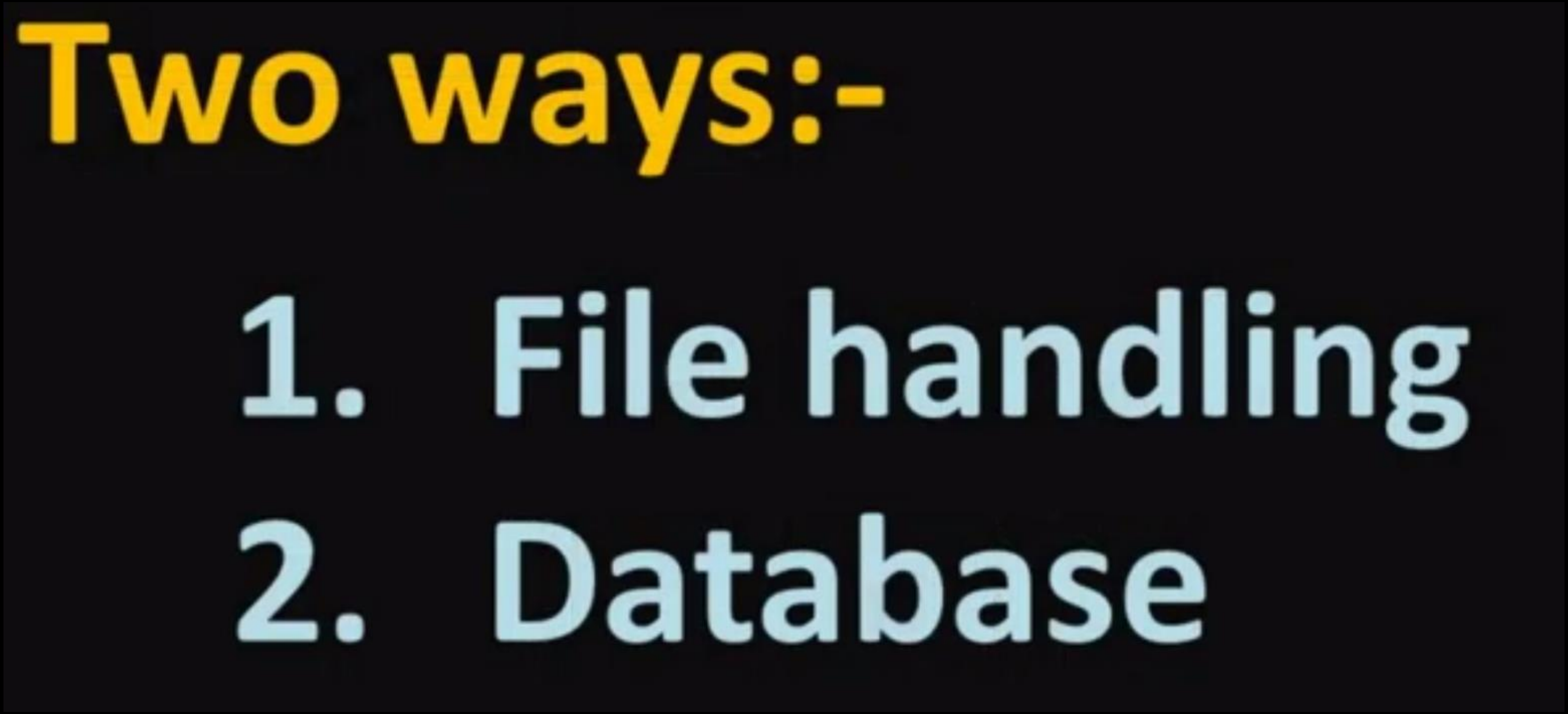
What is file handling?

File handling means:-

- Opening a file.
- Performing some operations on it.
- Closing a file

Example 3-4

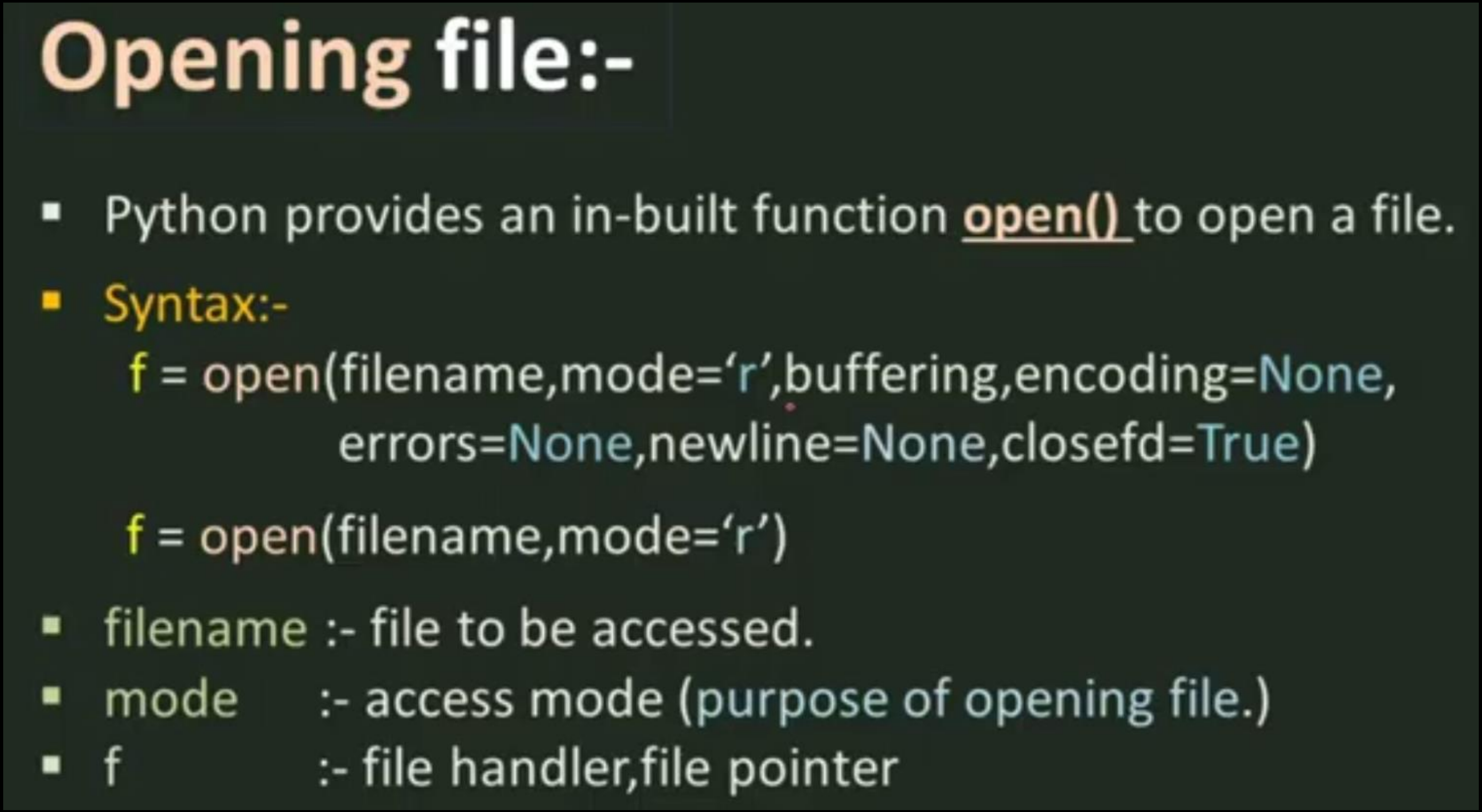
NEED OF FILE HANDLING



Two ways:-

1. File handling
2. Database

OPEN FUNCTION ARGUMENTS



Opening file:-

- Python provides an in-built function <u>**open()**</u> to open a file.
- Syntax:-

```
f = open(filename,mode='r',buffering,encoding=None,
            errors=None,newline=None,closefd=True)

f = open(filename,mode='r')
```

- filename :- file to be accessed.
- mode   :- access mode (purpose of opening file.)
- f   :- file handler,file pointer

OPEN FUNCTION ARGUMENTS

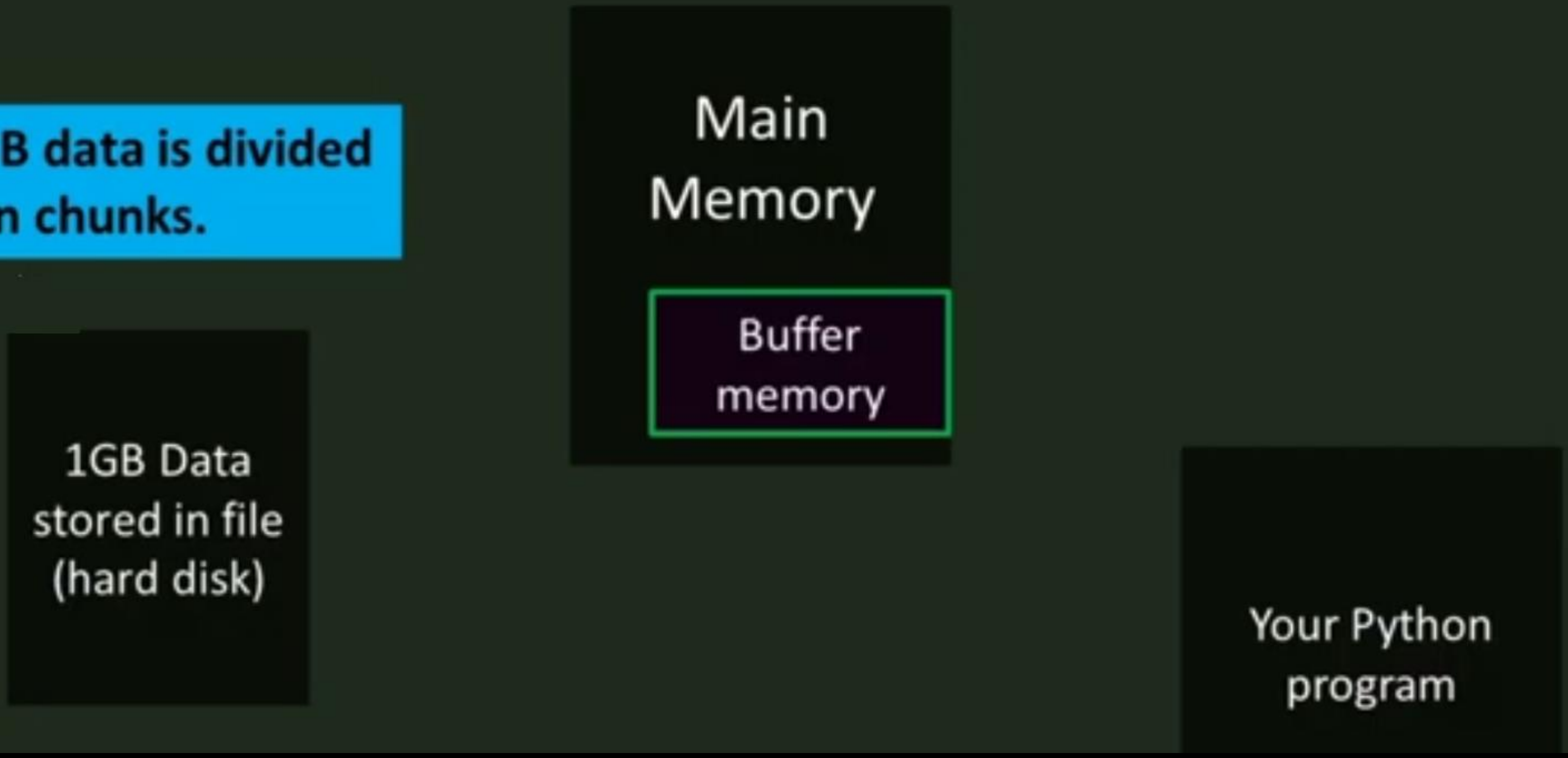
```
f = open(filename,mode='r',buffering,encoding=None,
              errors=None,newline=None,closefd=True)
```

buffering:-

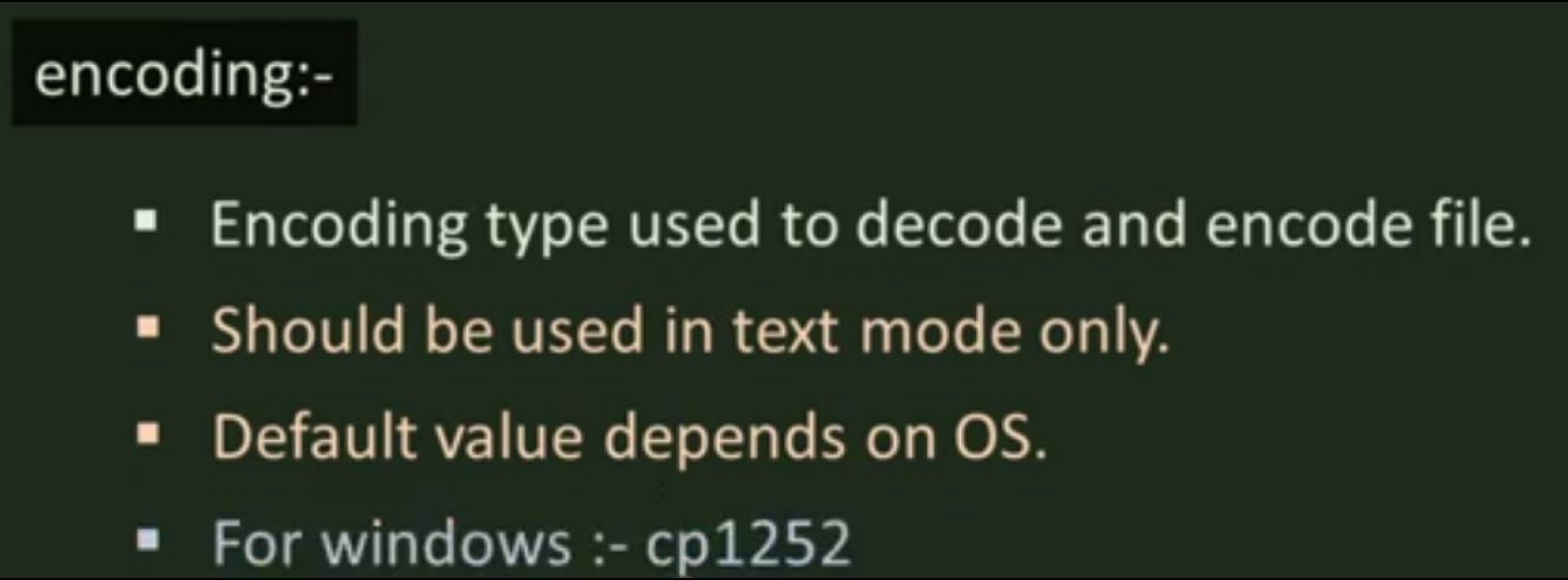
- Positive Integer value used to set buffer size for file.
- In text mode, buffer size should be 1 or more than 1.
- In binary mode, buffer size can be 0.
- Default size:- 4096-8192 bytes

OPEN FUNCTION ARGUMENTS

1GB data is divided in n chunks.



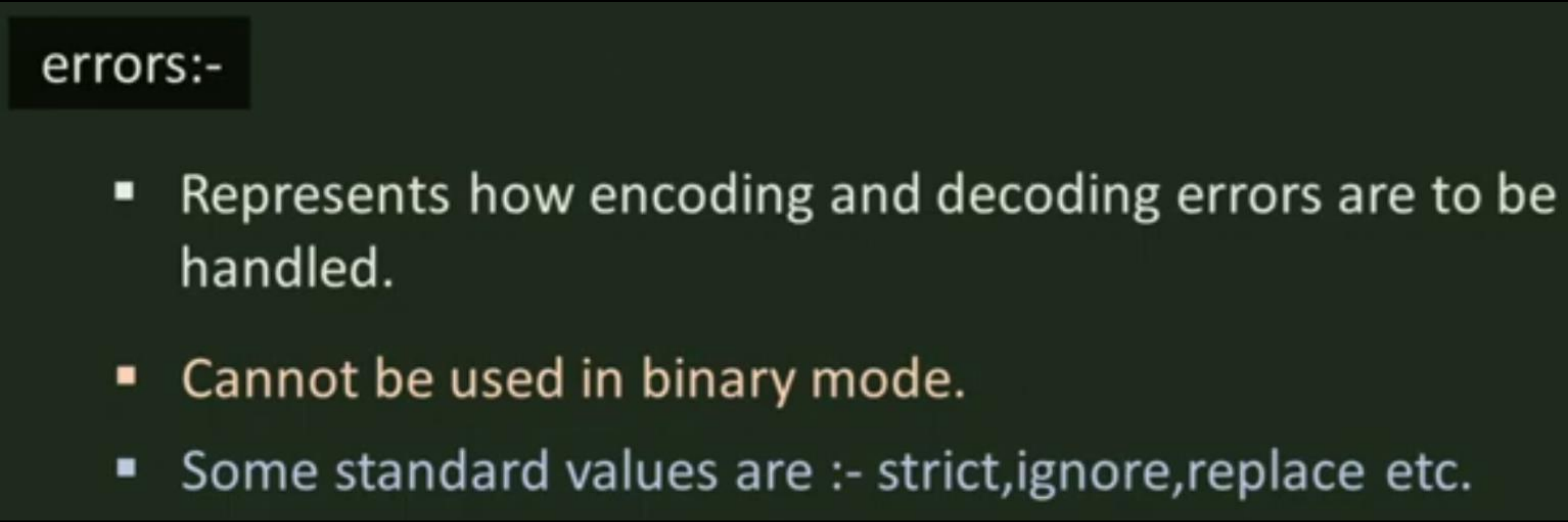
OPEN FUNCTION ARGUMENTS



encoding:-

- Encoding type used to decode and encode file.
- Should be used in text mode only.
- Default value depends on OS.
- For windows :- cp1252

OPEN FUNCTION ARGUMENTS



errors:-

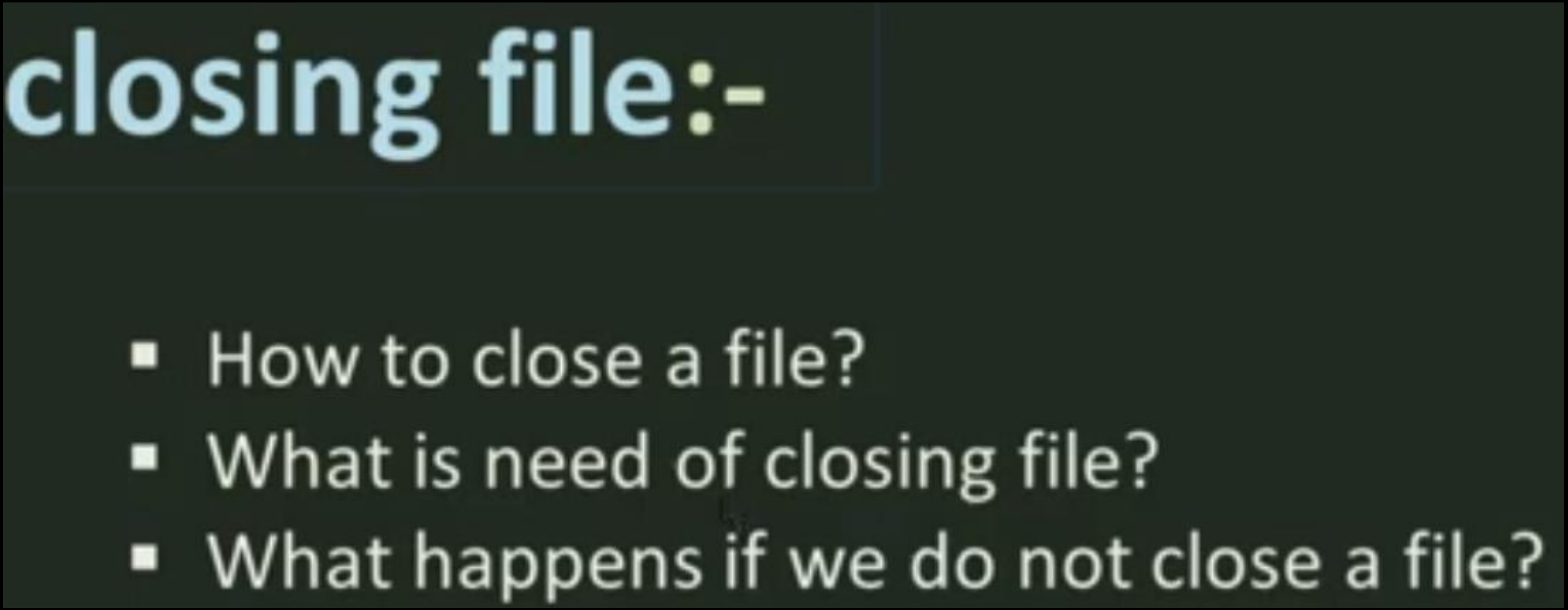
- Represents how encoding and decoding errors are to be handled.
- Cannot be used in binary mode.
- Some standard values are :- strict,ignore,replace etc.

OPEN FUNCTION ARGUMENTS

newline:-

- It can be \n, \r, \r\n.

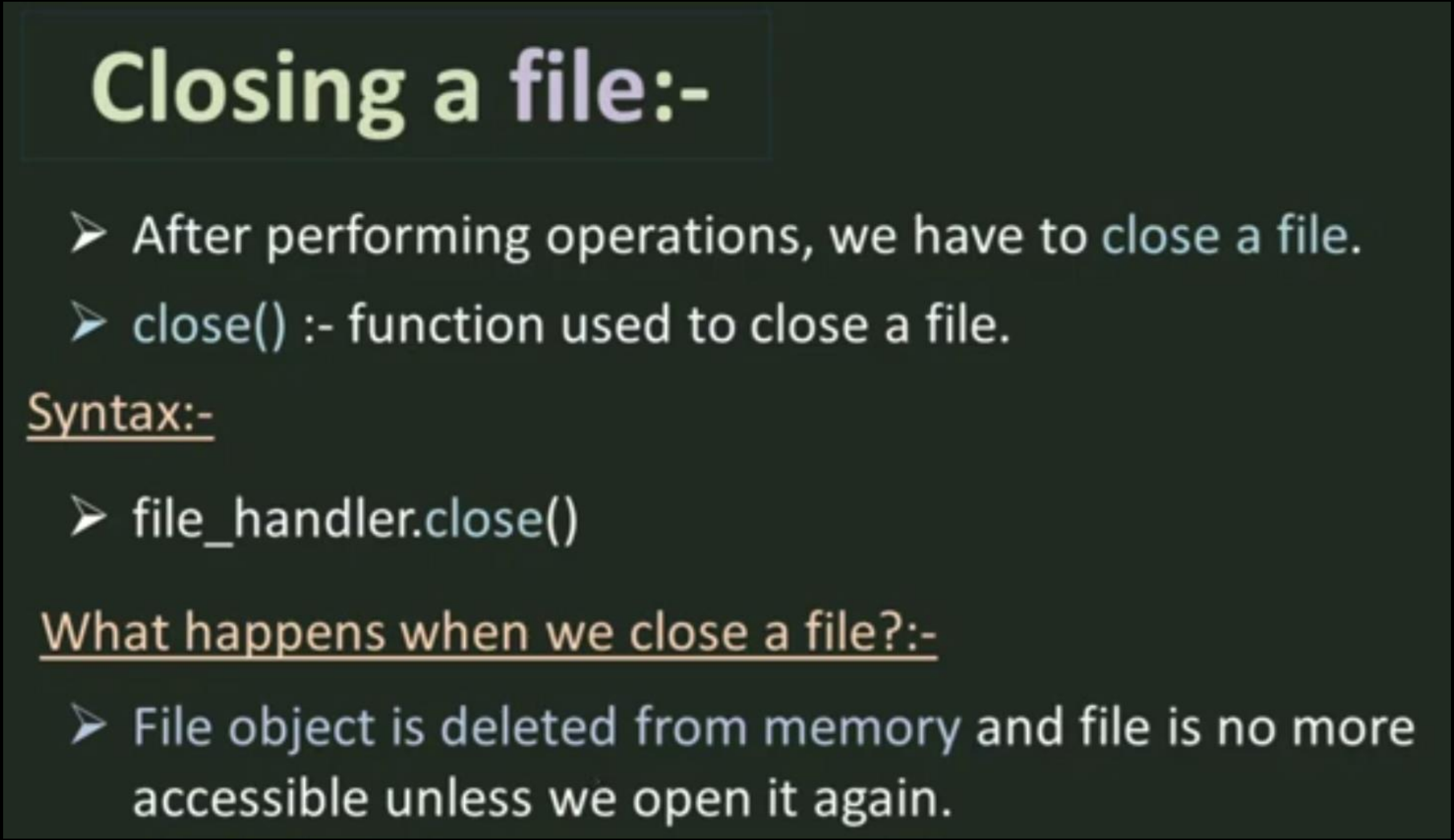
CLOSING FILE IN PYTHON


closing file:-

- How to close a file?
- What is need of closing file?
- What happens if we do not close a file?

Example 5

CLOSING FILE IN PYTHON



Closing a file:-

- After performing operations, we have to close a file.
- close() :- function used to close a file.

Syntax:-

- file_handler.close()

What happens when we close a file?:-

- File object is deleted from memory and file is no more accessible unless we open it again.

CLOSING FILE IN PYTHON

What happens when we do not close a file?:-

➢ After program execution, python garbage collector will destroy file object and closes file automatically.

Note:- Don’t rely on garbage collector.

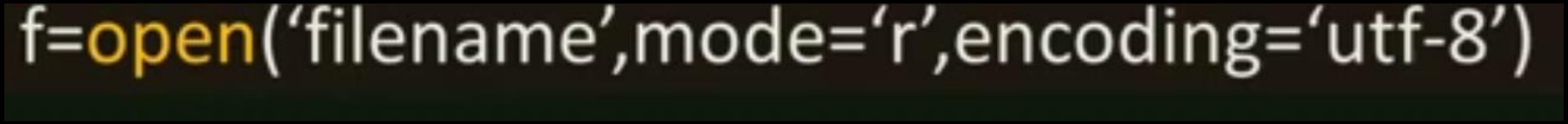
File.py:-

```
open()
...
#operations
...
#statements
#statements
#statements
#statements
```

Possible outcomes:-

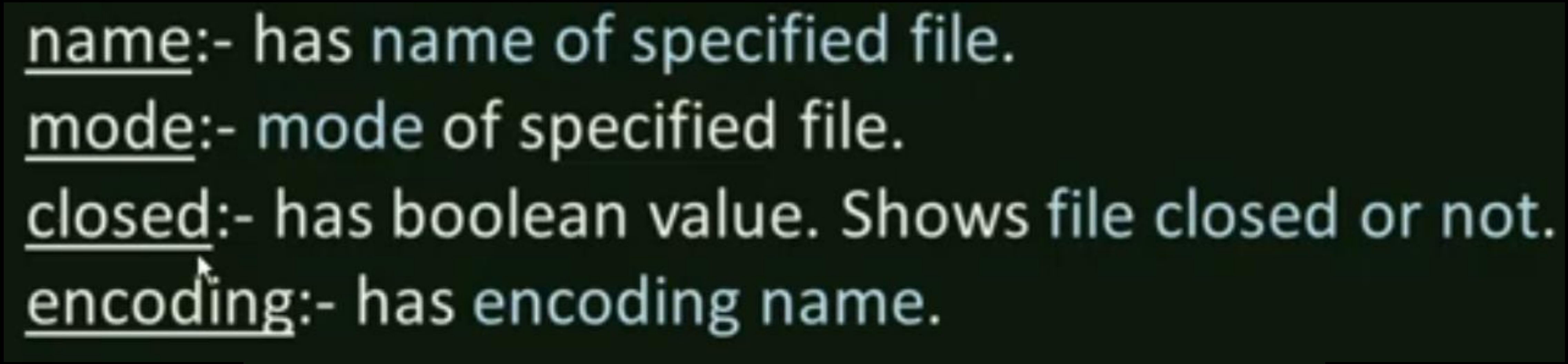
1. Data will corrupt.
2. Memory wastage.

FILE OBJECT VARIABLES IN PYTHON



```
f=open('filename',mode='r',encoding='utf-8')
```

File object variables:

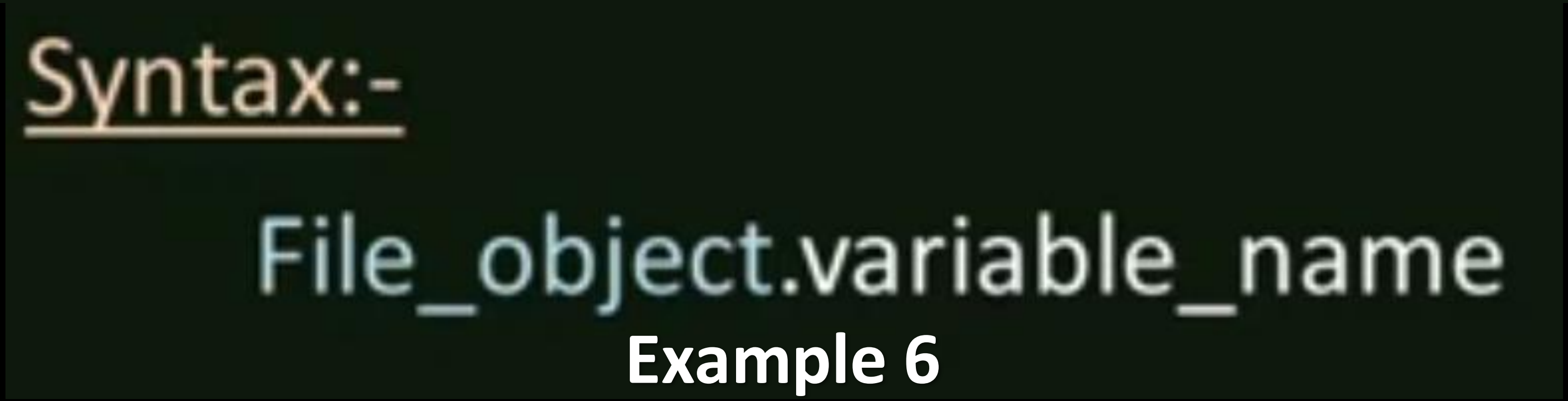


<u>name</u>:- has name of specified file.

<u>mode</u>:- mode of specified file.

<u>closed</u>:- has boolean value. Shows file closed or not.

<u>encoding</u>:- has encoding name.

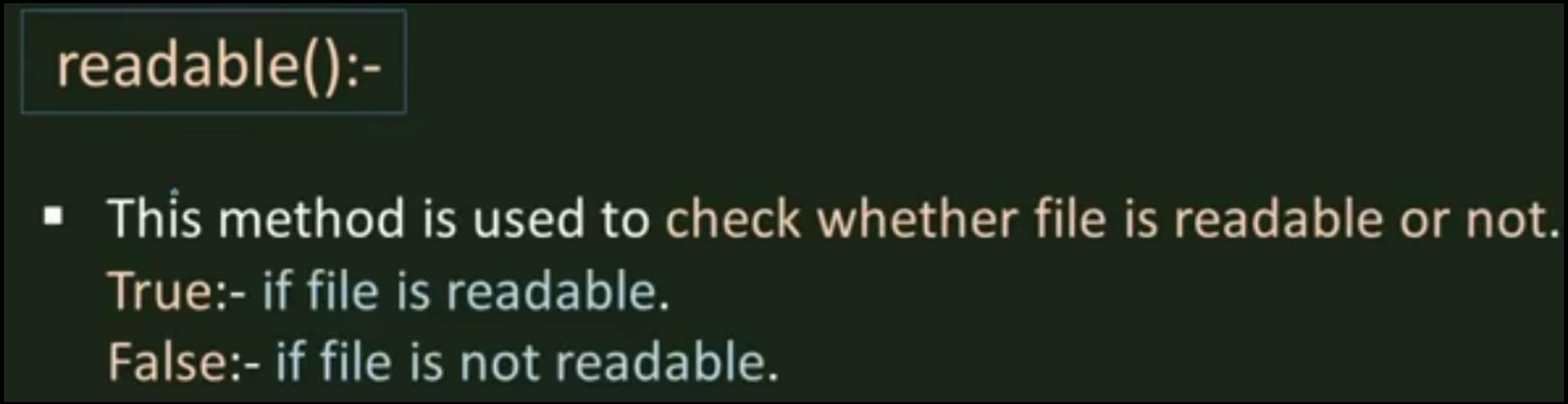


<u>Syntax:-</u>

File_object.variable_name

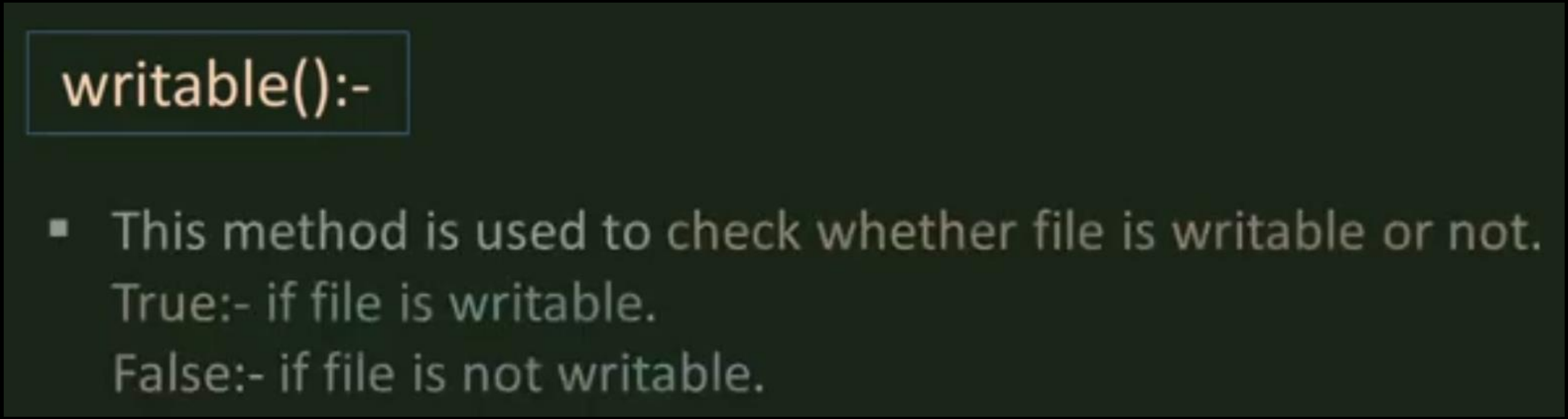
Example 6

CHECKING A FILE READABLE OR WRITEABLE



readable():-

- This method is used to check whether file is readable or not.
  True:- if file is readable.
  False:- if file is not readable.



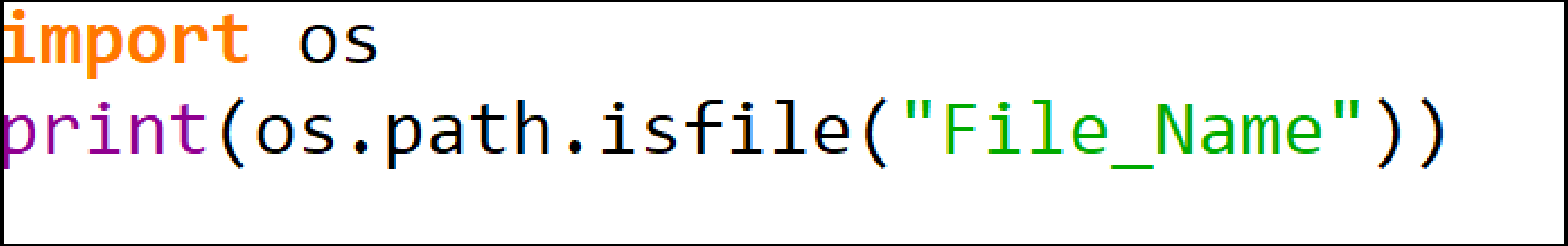
writable():-

- This method is used to check whether file is writable or not.
  True:- if file is writable.
  False:- if file is not writable.

Example 7

CHECKING A FILE EXIST OR NOT

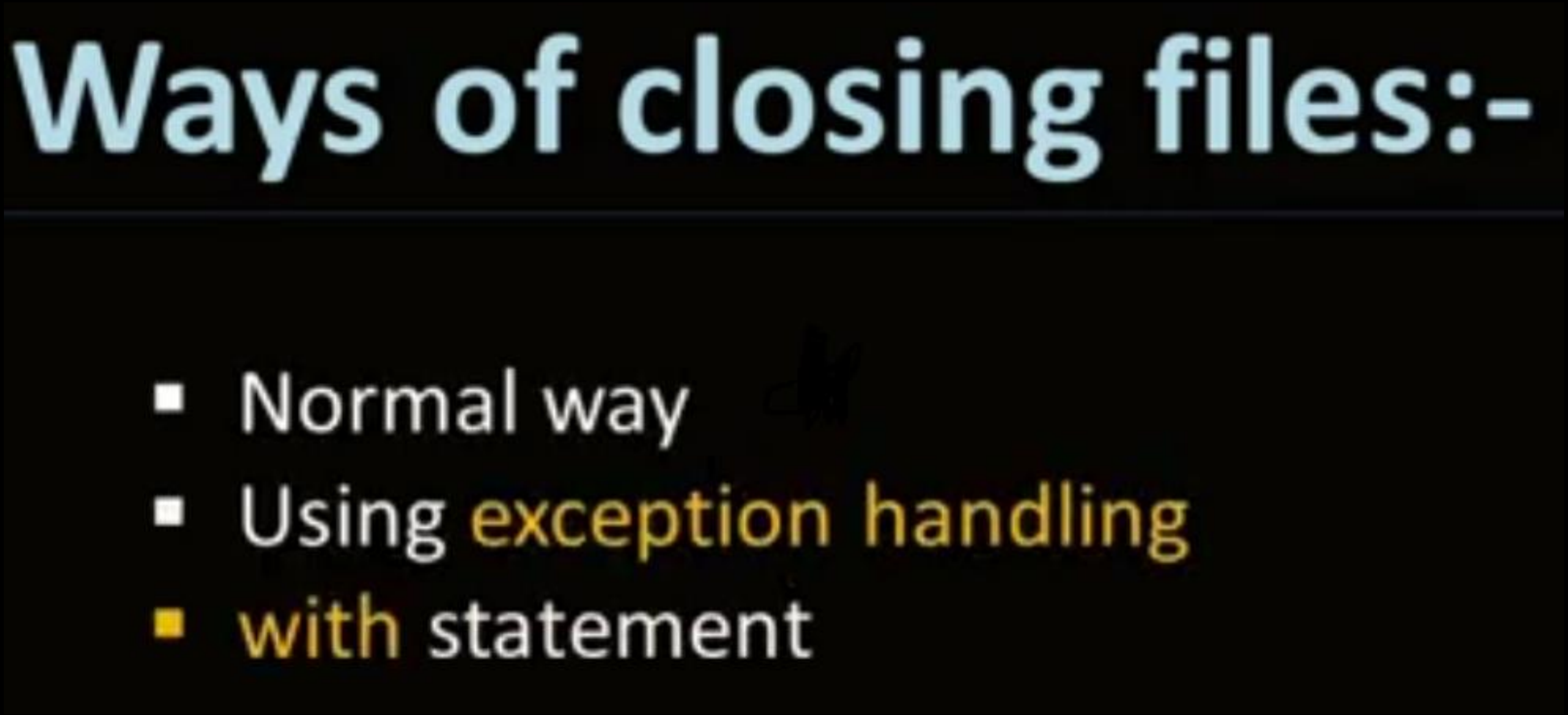
Syntax:



```
import os
print(os.path.isfile("File_Name"))
```

Example-8

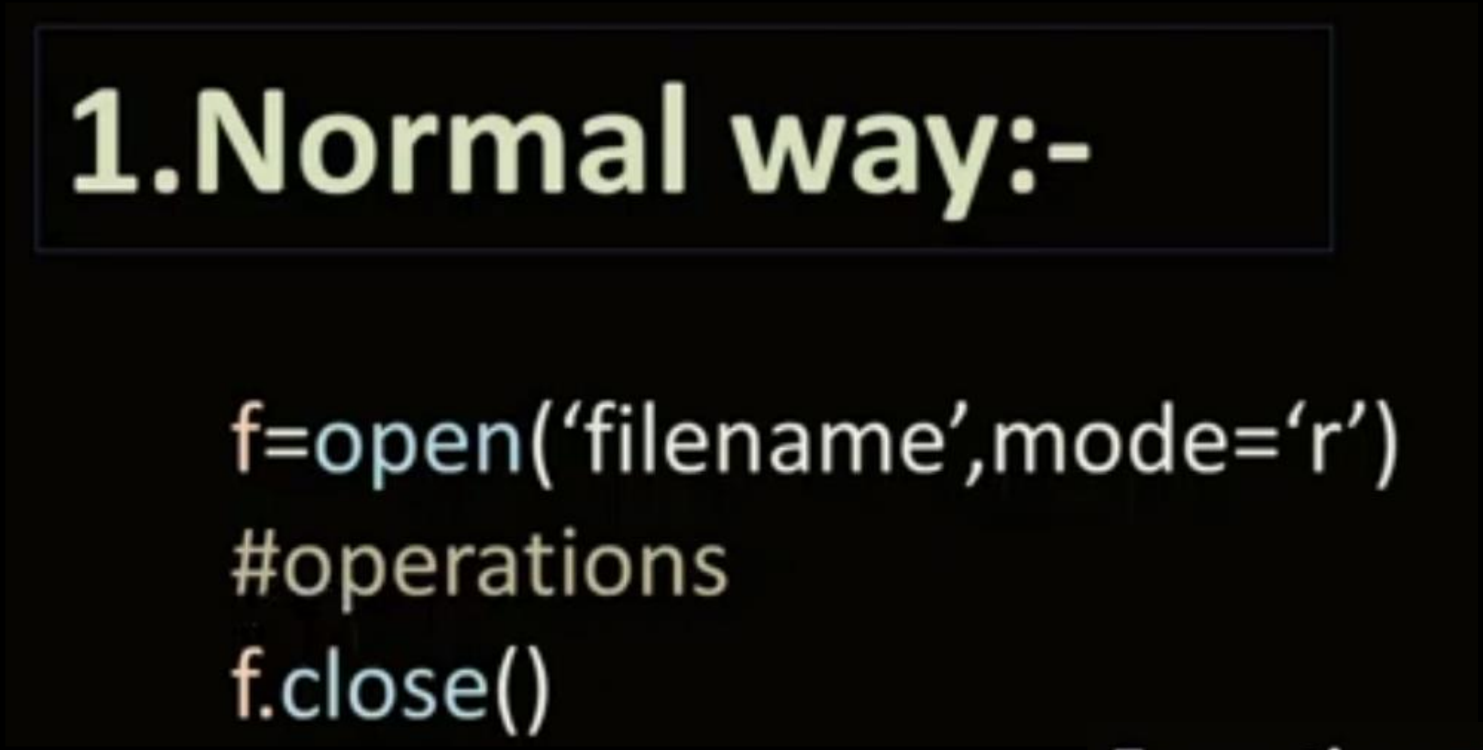
DIFFERENT WAYS OF CLOSING A FILE



Ways of closing files:-

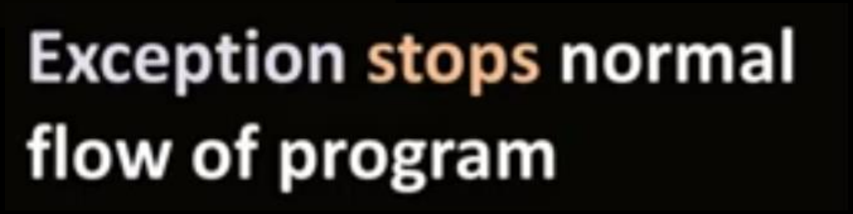
- Normal way
- Using exception handling
- with statement

DIFFERENT WAYS OF CLOSING A FILE



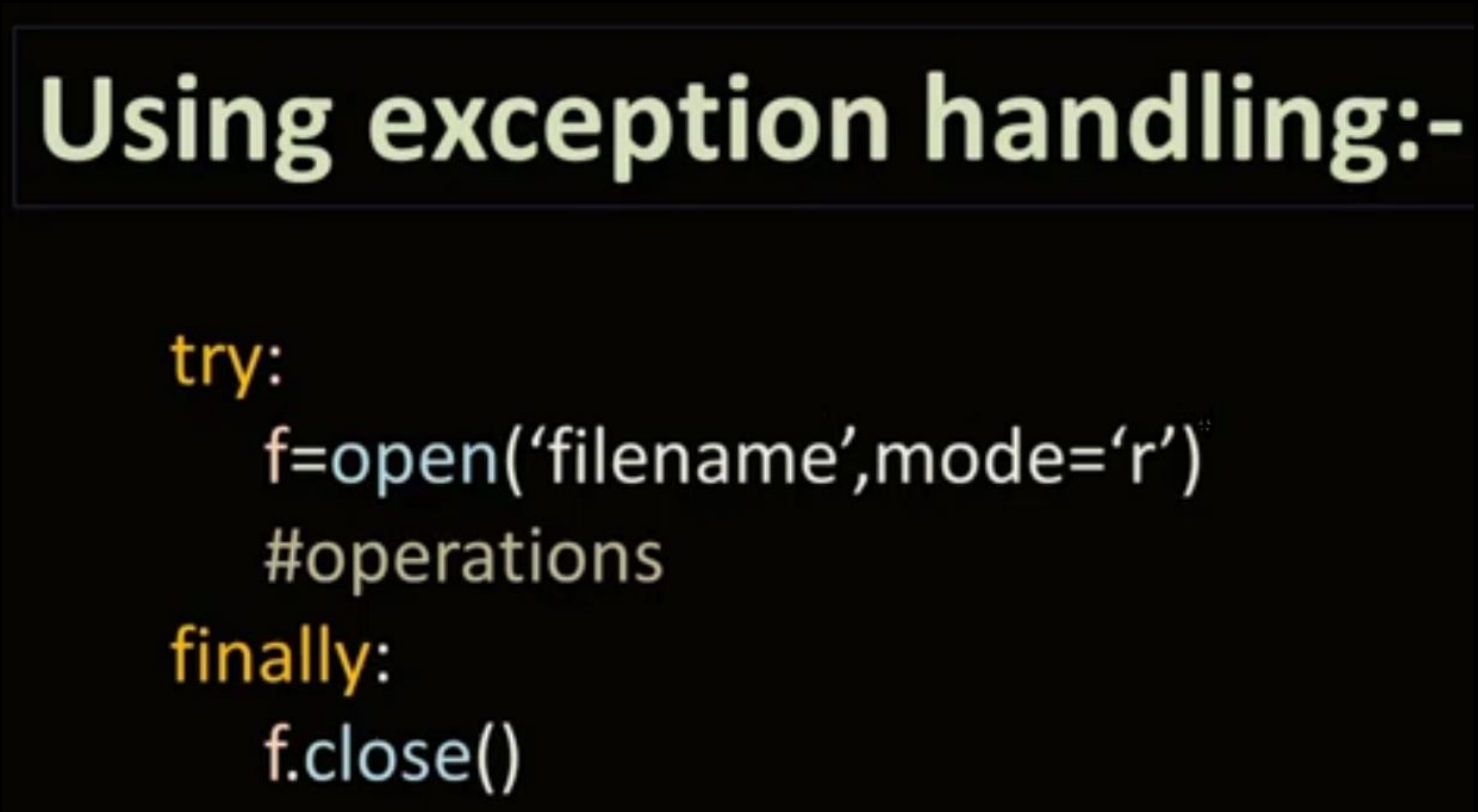
1.Normal way:-

```
f=open(‘filename’,mode=‘r’)
#operations
f.close()
```



Exception stops normal flow of program

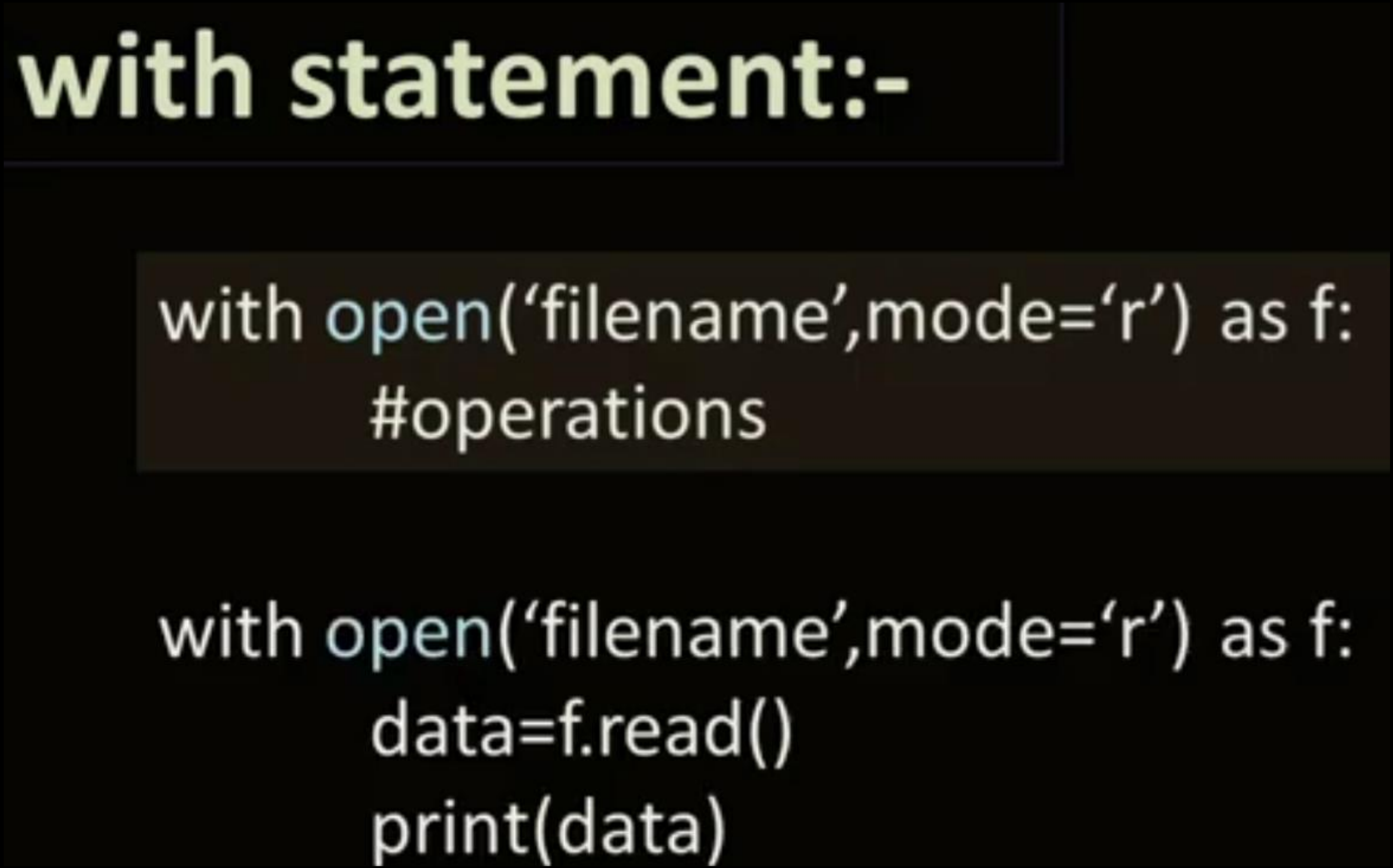
DIFFERENT WAYS OF CLOSING A FILE



Using exception handling:-

```
try:
    f=open('filename',mode='r')
    #operations
finally:
    f.close()
```

DIFFERENT WAYS OF CLOSING A FILE



with statement:-

```
with open('filename',mode='r') as f:
        #operations
```

```
with open('filename',mode='r') as f:
        data=f.read()
        print(data)
```

Thank You