

MODULE 4 :PYTHON FILE OPERATION

ONE SHOT

Today's Target

- MOST IMPORTANT QUESTION
- ALL AKTU PYQS

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E)

AKTU QUESTIONS

Q1	what are files? How they are useful
Q2	Describe the opening a file function in python
Q3	Describe the closing a file function in python
Q4	Discuss writing to a file operation
Q5	Discuss reading from file with example
Q6	What are the file input and output operation (read , write ,open close) in python programming
Q7	Discuss file i/o in python. How to perform open, read, write and close into a file . Write a python program to read a file line by line store it into a variable
Q8	What is the purpose of reading data from file in python

14
14

10/5 = 19

7

7 Marks

Q8	What are the different file access mode in python
Q9	Explain the purpose of read() function in python . How does it operate
Q10	What is the purpose of readline() and readlines() in python? What does it return
Q11	Explain the purpose of write() function in python .how does it operate
Q12	Describe the behavior of writelines() function in python. What kind of input does it expect
Q13	There is a file named input.txt . Enter some positive number into the file name input.txt read the content of the file and if odd number write it to odd.txt and if the number is even write it to even.txt
Q14	Describe python program to write the number letter and digits in given input string into a file object
Q15	What is file handler demonstrate the file handler procedure in detail

AKTU QUESTIONS

Q16	What is purpose of manipulating the file pointer using seek() function
Q17	How you use seek() function to move read position to the beginning of file?
Q18	How you use seek() function to move WRITE position to the beginning of file?
Q19	How you use seek() function to move read position to the relative to the current position

- A file is a container in computer storage devices used for storing data.
- How they are useful?
- When a program is terminated, the entire data is lost. Storing in a file will preserve your data even if the program terminates.
- If you have to enter a large number of data, it will take a lot of time to enter them all.
- However, if you have a file containing all the data, you can easily access the contents of the file using a few commands in C / Python
- You can easily move your data from one computer to another without any changes.

- The Python `open()` function is used to open internally stored files. It returns the contents of the file as Python objects.
- Python `open()` Function Syntax
- The `open()` function in Python has the following syntax:
- Syntax: `open(file_name, mode)`
- Parameters:.
- file name: This parameter as the name suggests, is the name of the file that we want to open.
- mode: This parameter is a string that is used to specify the mode in which the file is to be opened. The following strings can be used to activate a specific mode:

FILE OPENING MODES



Mode	DISCRIPTION
r	Open text file for reading. Raises an I/O error if the file does not exist.
r+	Open the file for reading and writing. Raises an I/O error if the file does not exist.
w	Open the file for writing. Truncates the file if it already exists. Creates a new file if it does not exist.
w+	Open the file for reading and writing. Truncates the file if it already exists. Creates a new file if it does not exist.
a	Open the file for writing. The data being written will be inserted at the end of the file. Creates a new file if it does not exist.
a+	Open the file for reading and writing. The data being written will be inserted at the end of the file. Creates a new file if it does not exist.
rb	Open the file for reading in binary format. Raises an I/O error if the file does not exist.
rb+	Open the file for reading and writing in binary format. Raises an I/O error if the file does not exist.

FILE OPENING MODES

Mode	DISCRIPTION
wb	Open the file for <u>writing</u> in <u>binary</u> format. <u>Truncates the file if it already exists.</u> <u>Creates a new file if it does not exist.</u>
wb+	Open the file for <u>reading and writing</u> in <u>binary</u> format. <u>Truncates the file if it already exists.</u> <u>Creates a new file if it does not exist.</u>
ab	Open the file for <u>appending</u> in <u>binary</u> format. <u>Inserts data at the end of the file.</u> <u>Creates a new file if it does not exist.</u>
<u>ab+</u>	<u>Open the file for reading and appending</u> in <u>binary</u> format. <u>Inserts data at the end of the file.</u> <u>Creates a new file if it does not exist.</u>

write

How to open a file in Python?

- In Python, we can open a file by using the `open()` function already provided to us by Python. By using the `open()` function, we can open a file in the current directory as well as a file located in a specified location with the help of its path. In this example, we are opening a file "gfg.txt" located in the current directory and "gfg1.txt" located in a specified location..

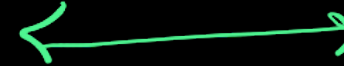
```
main.py
1  # opens gfg text file of the current directory
2  f = open("gfg.txt")
3
4  # specifying the full path
5  f = open("C:/HP/Desktop/gfg1.txt")
6
```

Reading and Writing the file

main.py

```
1 my_file = open("gee.txt", "w")
2 my_file.write("python is best for Dsa")
3 my_file.close()
4
5 #Let's read the contents of the file now
6 my_file = open("gee.txt", "r")
7 print(my_file.read())
8
```

- Python provides inbuilt functions for creating, writing and reading files. There are two types of files that can be handled in python, normal text files and binary files (written in binary language, 0s and 1s).
- **Text files**: In this type of file, Each line of text is terminated with a special character called EOL (End of Line), which is the new line character ('\n') in python by default.
- **Binary files**: In this type of file, there is no terminator for a line and the data is stored after converting it into machine-understandable binary language.
- **write()** : Inserts the string str1 in a single line in the text file.
- `File_object.write(str1)`
- **writelines()** : For a list of string elements, each string is inserted in the text file. Used to insert multiple strings at a single time.
- `File_object.writelines(L)` for `L = [str1, str2, str3]`



Writing to file in Python

```

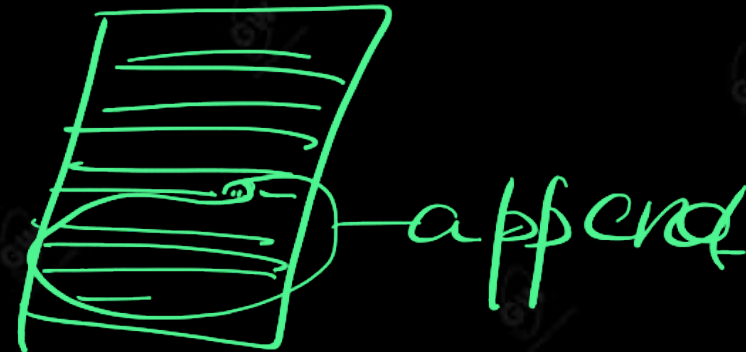
1 # Python program to demonstrate
2 # writing to file
3
4 # Opening a file
5 file1 = open('myfile.txt', 'w')
6 L = ["This is Delhi \n", "This is Paris \n", "This is London \n"]
7 s = "Hello\n"
8
9 # Writing a string to file
10 file1.write(s)
11
12 # Writing multiple strings
13 # at a time
14 file1.writelines(L)
15
16 # Closing file
17 file1.close()
18
19 # Checking if the data is
20 # written to file or not
21 file1 = open('myfile.txt', 'r')
22 print(file1.read())
23 file1.close()
24

```

~~Hello~~
 This is Delhi
 This is Paris
 This is London

- When the file is opened in append mode, the handle is positioned at the end of the file. The data being written will be inserted at the end, after the existing data. Let's see the below example to clarify the difference between write mode and append mode.
- Write() it will overwrite the existing content of the file

↓
write mode में
open करके
file



```

1  # Python program to illustrate
2  # Append vs write mode
3  file1 = open("myfile.txt", "w")
4  L = ["This is Delhi \n", "This is Paris \n", "This is London \n"]
5  file1.writelines(L)
6  file1.close()
7  # Append-adds at Last
8  file1 = open("myfile.txt", "a") # append mode
9  file1.write("Today \n")
10 file1.close()
11 file1 = open("myfile.txt", "r")
12 print("Output of Readlines after appending")
13 print(file1.read())
14 print()
15 file1.close()
16 # Write-Overwrites
17 file1 = open("myfile.txt", "w") # write mode
18 file1.write("Tomorrow \n")
19 file1.close()
20 file1 = open("myfile.txt", "r")
21 print("Output of Readlines after writing")
22 print(file1.read())
23 print()
24 file1.close()

```


- close() function closes the file and frees the memory space acquired by that file. It is used at the time when the file is no longer needed or if it is to be opened in a different file mode. Syntax:
- File_object.close()

2 marks

Difference between-

help (2 marks) / 7 marks
or shp

read()	readline()	readlines()
<p>➤ <u>The read method reads the entire contents of a file and returns it as a string.</u></p>	<p>➤ <u>The readline method reads a single line from a file and returns it as a string. This means that if you use readline, you can read the contents of a file line by line, which can be useful for processing large files that do not fit in memory</u></p> <pre>file = open("filename.txt", "r") file.readline()</pre>	<p>➤ <u>the readlines method reads the entire contents of a file and returns it as a list of strings, where each element of the list is a single line of the file</u></p> <pre>file = open("filename.txt", "r") file.readlines()</pre>

readline() Example

- readline() return a string containing the characters up to and including the newline character

```

1 File=open('example.txt' , 'r')
2 Line1=file.readline()
3 Line2=file.readline()
4 print(line1)
5 print(line2)
6 #if in this file
7 #hello, python
8 #bye file handling

```

Output
if in this field
hell, python

readlines() Example

- It return a list where each element is a string representing a line from the file

```
main.py
1 File=open('example.txt','r')
2 Lines=file.readlines()
3 print(lines)
4 #if in this file
5 #hello, python
6 #bye file handling
7 #hello again with function
8 the output will be
9 [ 'hello python \n' , 'bye file handling\n', 'hello again with function\n']
10
11
```

example.txt

file

*hello python
bye file handling
hello again with function*

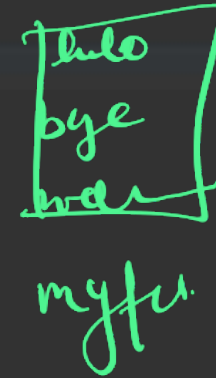
Purpose of reading the data from file in python

- Reading from file allow a program to retrieve and use the data that was previously saved.
- Files can serve as a input source for a program
- Files are used for data analysis
- File is the universal way to exchange the data between different program or system
- Files are commonly used to log events, errors and other information generated by a program
- files are primary means for creating backups or archiving data for future use

Write a python program to read a file line by line store it into a variable

main.py

```
1 L=["hello\n","bye\n","welcome to gateway classes\n"]
2 #writing to a file
3 file1=open("myfile.txt","w")
4 file1.writelines(L)
5 file1.close()
6 file1=open("myfile.txt",'r')
7 lines=file1.readlines()
8
```

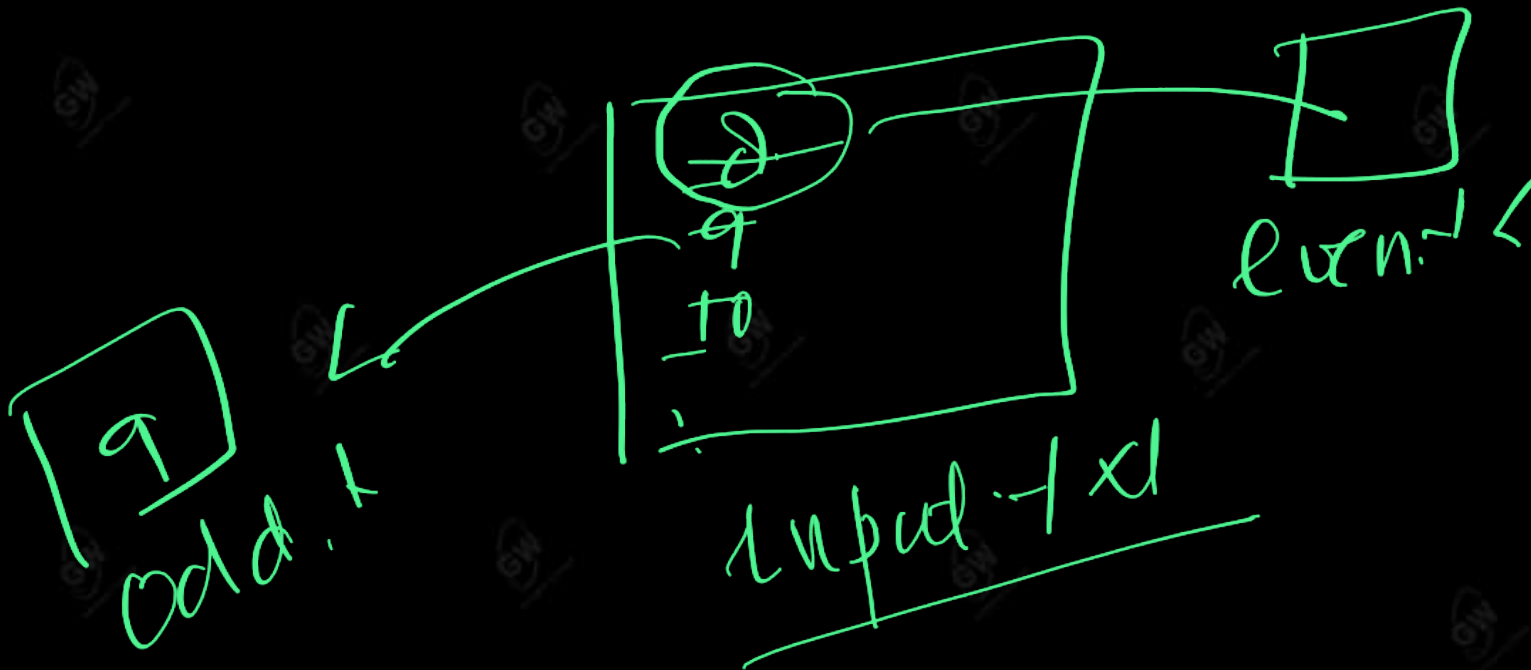


Write a python program to read a file line by line store it into a variable

copy X

```
main.py
1 L=["hello\n","bye\n","welcome to gateway classes\n"]
2 #writing to a file
3 file1=open("myfile.txt","w")
4 file1.writelines(L)
5 file1.close()
6 file1=open("myfile.txt",'r')
7 lines=file1.readlines()
8
```

There is a file named input.txt . Enter some positive number into the file name input.txt read the content of the file and if odd number write it to odd.txt and if the number is even write it to even.txt.



main.py

```

1  file=open("input.txt","w")
2  file.write("18\n")
3  file.write("15\n")
4  file.write("13\n")
5  file.write("2\n")
6  file.write("9\n")
7  file.write("88\n")
8  file.close()
9  file1=open("input.txt","r")
10 for i in file1:
11     if i.strip:
12         num=int(i)
13         if(num%2==0):
14             even=open("even.txt","a")
15             even.write(str(num))
16             even.write("\n")
17         else:
18             odd=open("odd.txt","a")
19             odd.write(str(num))
20             odd.write("\n")
21
22
23
24

```

Describe python program to write the number letter and digits in given input string into a file object

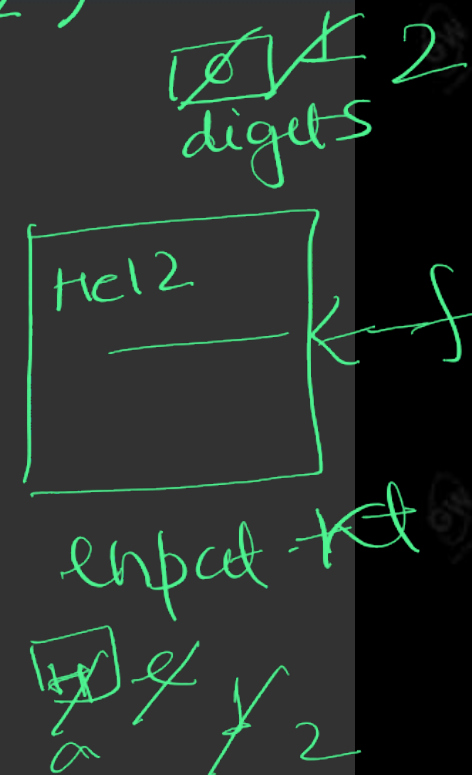
```

1 file=open("input.txt","w")
2 file.close()
3 letters=0
4 digits=0
5 f=open("input.txt","r")
6 for a in f:
7     if a.isalpha():
8         letters=letters+1
9
10
11
12     else:
13         digits=digit+1
14
15 Print("number of letters", letters)
16 Print("number of digit", digit)
17
18

```

Handwritten notes on the code:

- Next to line 1: $\frac{1}{\boxed{0}} \times 2$ letters
- Next to line 2: $\frac{1}{\boxed{0}} \times 2$ digits
- Next to line 15: $\frac{1}{\boxed{0}} \times 2$
- Next to line 16: $\frac{1}{\boxed{0}} \times 2$



- File handlers often referred as file objects, are essential components in python for working with python. They serve as an interface between your python program and external files on your computer storage.

Purpose of file handlers

- File handlers provide a method to read from and write to files
- File handlers help manage system resources efficiently. They automatically handle the opening and closing of files.
- File handlers ensure file is accessed according to the specified mode
- File handlers keep track of the current position within the file. this allows you to read or write at specific location or to move the pointer to different position within the file
- They handle error that occur during file operations

- Tell about how to open file close file and write in a file append file
- All these topic covered in previous slide

R)

→ write()
→ close()

Python seek() function

- seek() method
- In Python, seek() function is used to change the position of the File Handle to a given specific position.
File handle is like a cursor, which defines from where the data has to be read or written in the file.
- Syntax: f.seek(offset, from_what), where f is file pointer
- Parameters:
 - Offset: Number of positions to move forward
 - from_what: It defines point of reference.
 - Returns: Return the new absolute position.

- The reference point is selected by the `from_what` argument. It accepts three values:
- 0: sets the reference point at the beginning of the file
- 1: sets the reference point at the current file position
- 2: sets the reference point at the end of the file
- By default `from_what` argument is set to 0.
- Note: Reference point at current position / end of file cannot be set in text mode except when offset is equal to 0.
- Example 1: Let's suppose we have to read a file named "GfG.txt" which contains the following text:
- "hello guys python is a important language."

How you use seek() function to move read position to the beginning of file?

```
ain.py
1 f = open("GfG.txt", "r")
2 f.seek(0)
3 print(f.tell())
4 print(f.readline())
5 f.close()
6
7
8
```

Head file

Python seek() function

Seek Operation	Meaning
<code>f.seek(0)</code>	Move file pointer to the beginning of a File
<code>f.seek(5)</code>	Move file pointer five characters ahead from the beginning of a file.
<code>f.seek(0, 2)</code>	Move file pointer to the end of a File
<code>f.seek(5, 1)</code>	Move file pointer five characters ahead from the current position.
<code>f.seek(-5, 1)</code>	Move file pointer five characters behind from the current position.
<code>f.seek(-5, 2)</code>	Move file pointer in the reverse direction. Move it to the 5 th character from the end of the file

1 2 3 4 5
- - - - -
 5

- - - - -

How you use seek() function to move WRITE position to the beginning of file?

main.py

```
1 f = open("GfG.txt", "r")
2 f.seek(0)
3 file.write("this is the new content")
4 print(f.readline())
5 f.close()
```

How you use seek() function to move read position to the relative to the current position

```
main.py
1 f = open("GfG.txt", "r")
2 f.seek(5,0)
3 print(f.readline())
4 f.close()
5
6
7
8
```