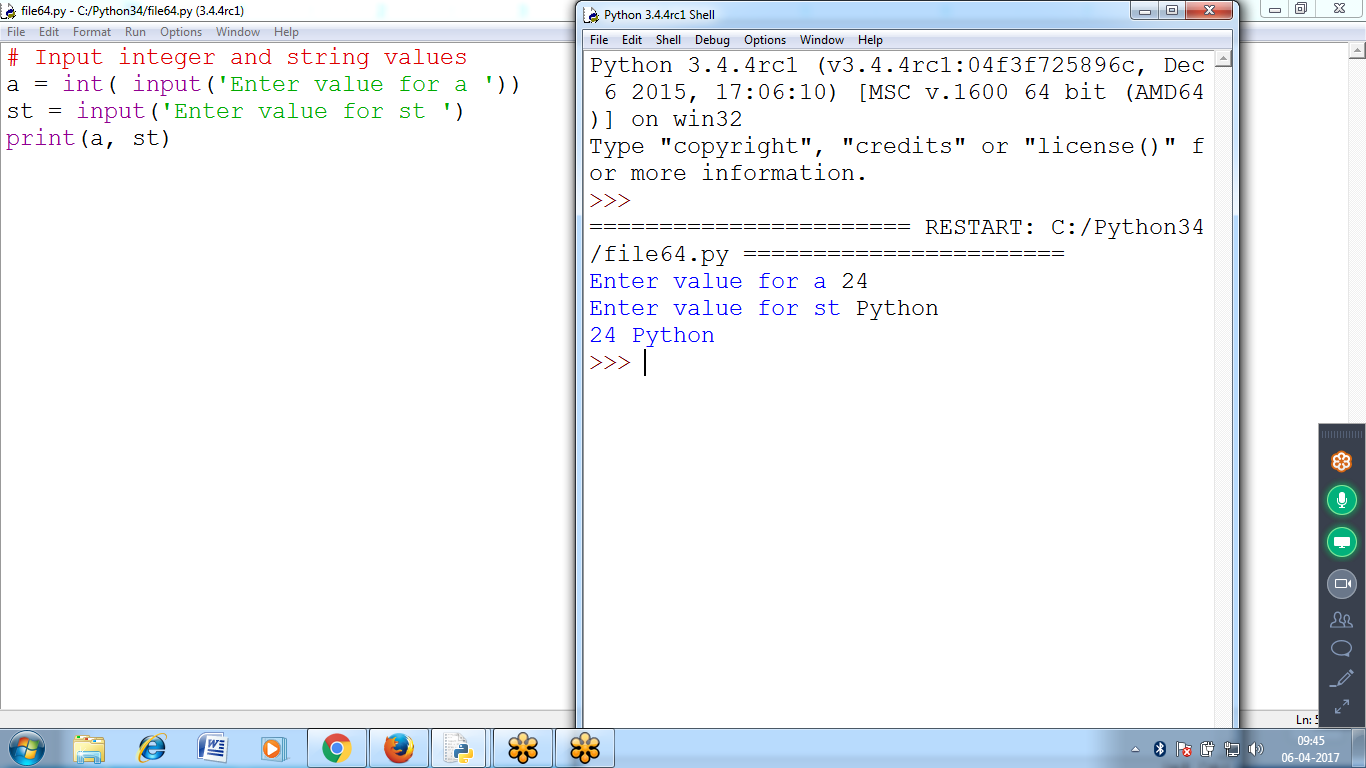
# **Processor :: RAM (Temporary)**

PRocessor : HD ( Permananent)

# 

# **Reading and Writing Files**

* Variables are used to store data while program is running
* To store data even after program has finished, you need to save it to a file



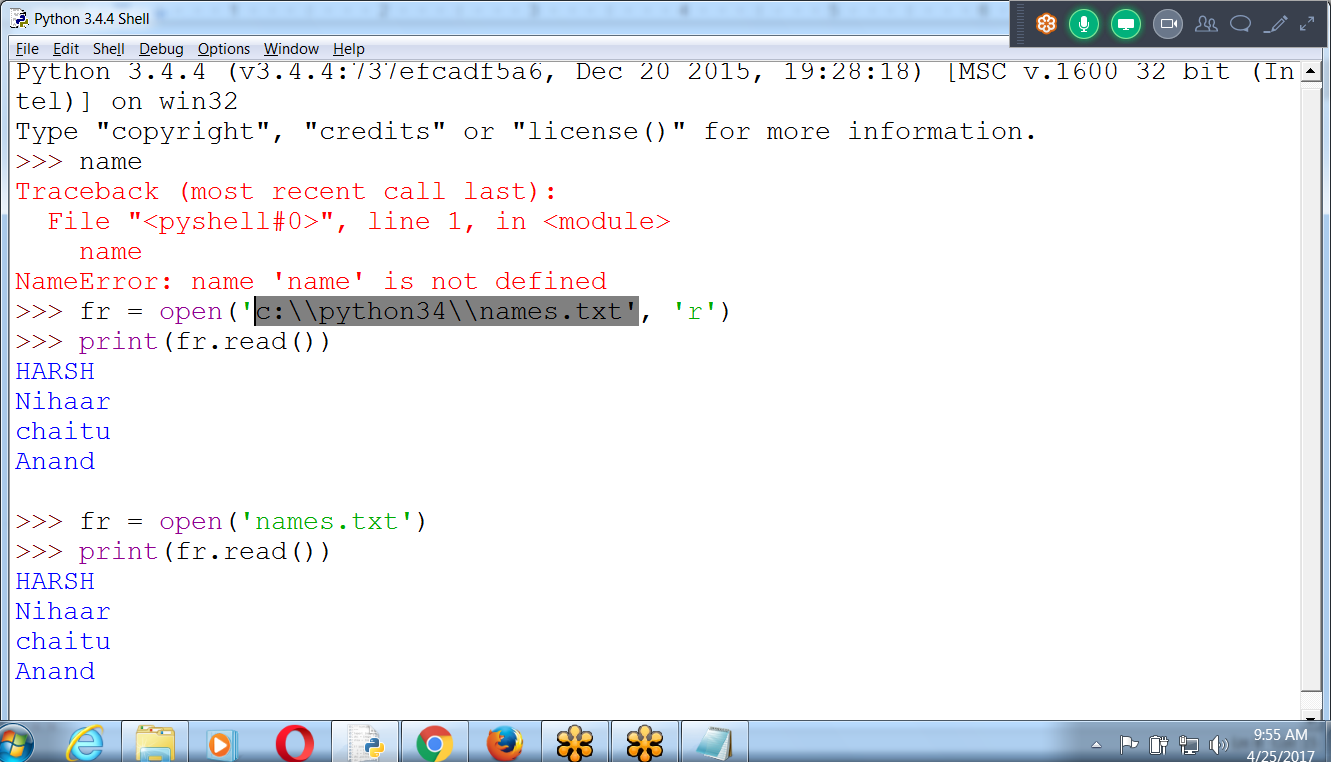
# **The File Reading/Writing Process**

* The functions relating to plaintext files. *Plaintext files* contain only basic text characters and **do not** include font, size, or color information.
* Text files with the *.txt* extension or Python script files with the *.py* extension are examples of **plaintext files**.
* Programs can easily read the contents of **plaintext files** and treat them as an ordinary string value.
* ***Binary files*** are all other file types, such as **word processing documents, PDFs, images, spreadsheets, and executable(.exe) programs**. If you open a binary file in Notepad or TextEdit, it will look like scrambled characters.

There are three steps to reading or writing files in Python.

1. Call the **open() function** to return a File object.
2. Call the **read() or write()** method on the File object.
3. Close the file by calling the **close() method** on the File object

**Reading Data from a file**



>>> fr = open('c:\\python34\\names.txt', 'r')

>>> print(fr.read())

HARSH

Nihaar

chaitu

Anand

**(or) Default : c:\\python34 folder at the time PYTHON Installation**

>>> fr = open('names.txt')

>>> print(fr.read())

HARSH

Nihaar

chaitu

Anand

>>>

**To Know the Path where Python has been installed**

**>>> import os**

**>>> os.getcwd()**

'D:\\mypython'

>>>

>>>

>>>

>>>

>>> fobj = open('d:\\mypython\\hello.txt', 'r')

>>> print(fobj.read())

Hi Good Morning

HARSH Technologies

Ameerpet

>>>

>>>

>>> fobj = open('hello.txt')

>>> print(fobj.read())

Hi Good Morning

HARSH Technologies

Ameerpet

**>>> fr=open('namesall.txt') # Default “R” mode and path c:\\python34**

**>>> print(fr.read())**

Python Ranks First

Java is Second

Python uses in Data science

aws, devops

Kiran

Raju

Sanjay

Kumar

Arjun

siva

ramu is good bod

**>>> print(fr.read()) # Reached EOF**

>>> fr=open('namesall.txt')

>>> print(fr.read())

Python Ranks First

Java is Second

Python uses in Data science

aws, devops

Kiran

Raju

Sanjay

Kumar

Arjun

siva

ramu is good bod

>>>

**Writing data onto a file**

>>> fw = open('c:\\python34\\mynames.txt', **'w')**

>>> fw.write('Ramu')

4

>>> fw.write('sita')

4

>>> fw.close() # after closing, data moves to file

>>> fw.write(' Kiran kumar ')

>>**> fw = open('mynames.txt', 'w')**

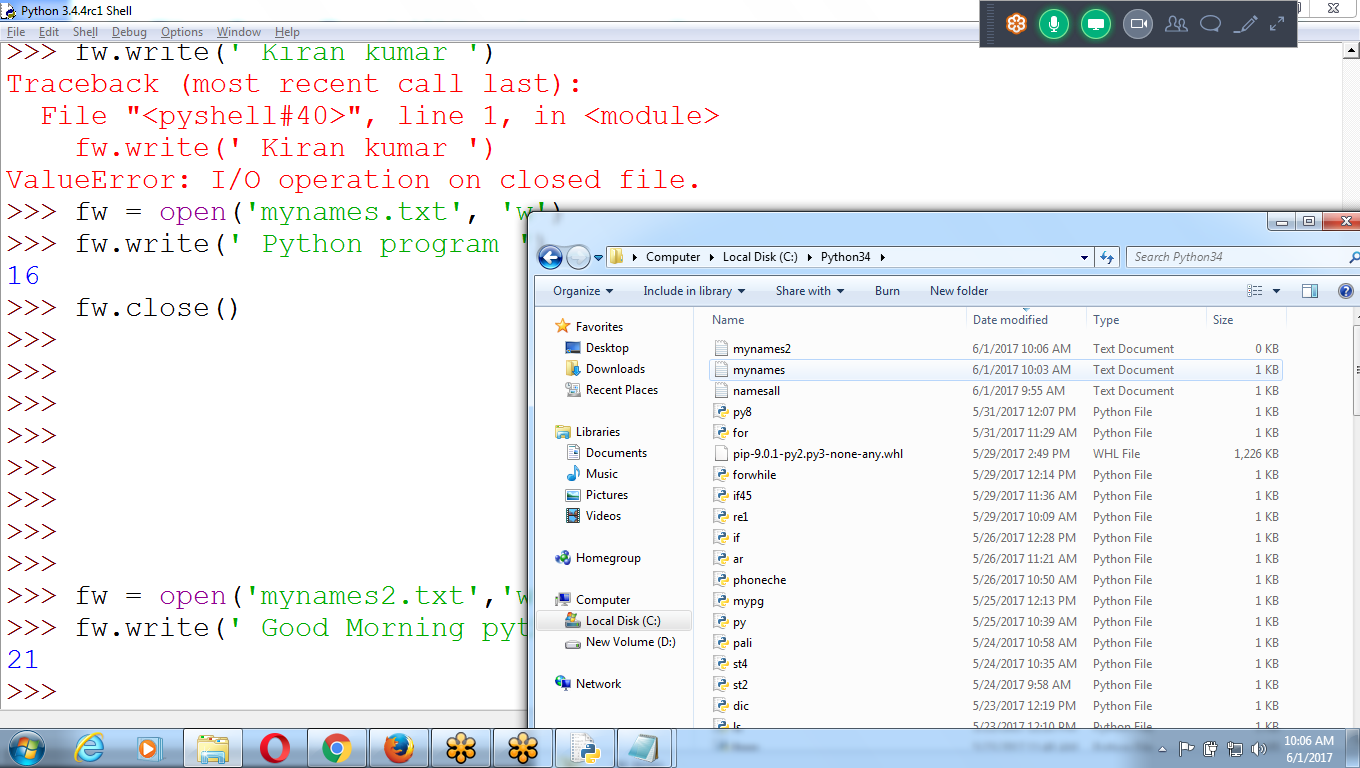
>>> fw.write(' Python program ')

16

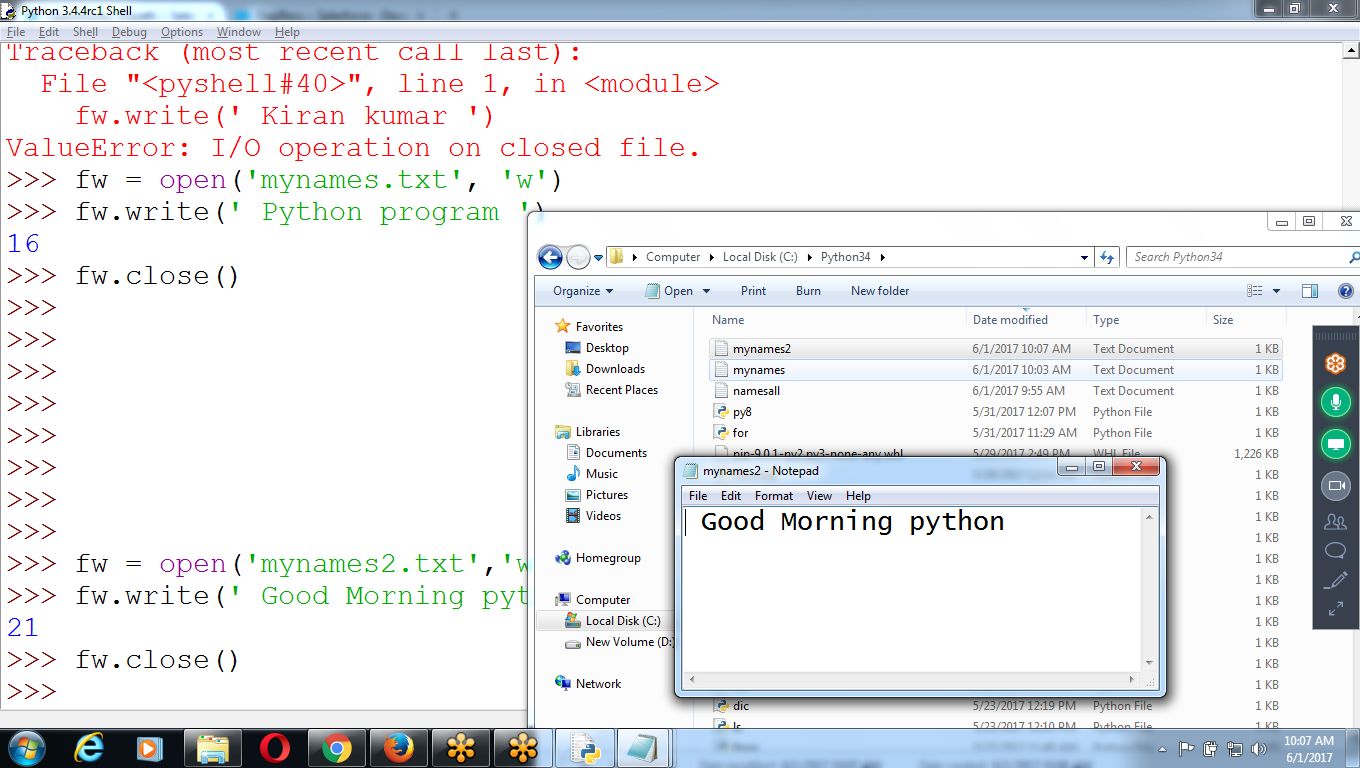
>>> fw.close()

>>>

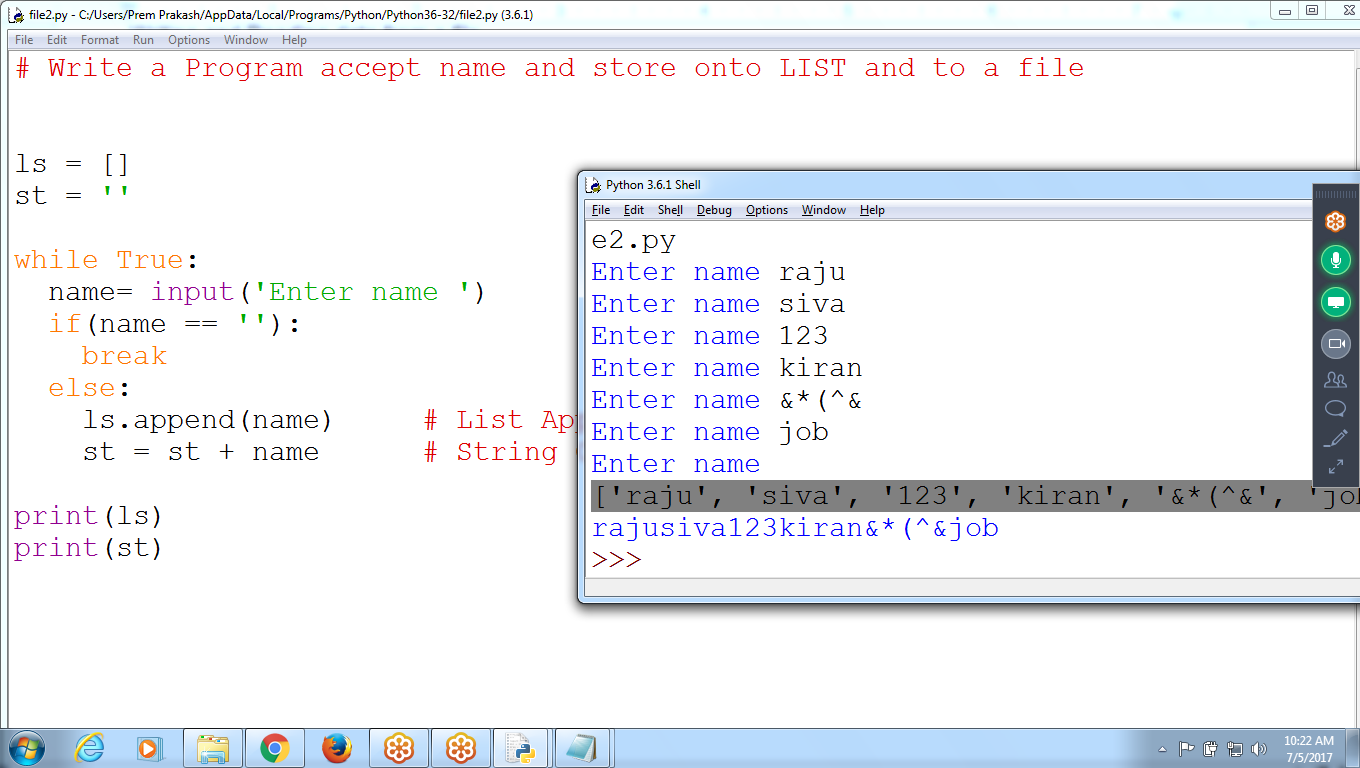
Until file is close, bytes size is ZERO



After CLosing



Storing Data onto a LIST and String



# Write a Program accept name and store onto LIST and to a file

ls = []

st = ''

while True:

name= input('Enter name ')

if(name == ''):

break

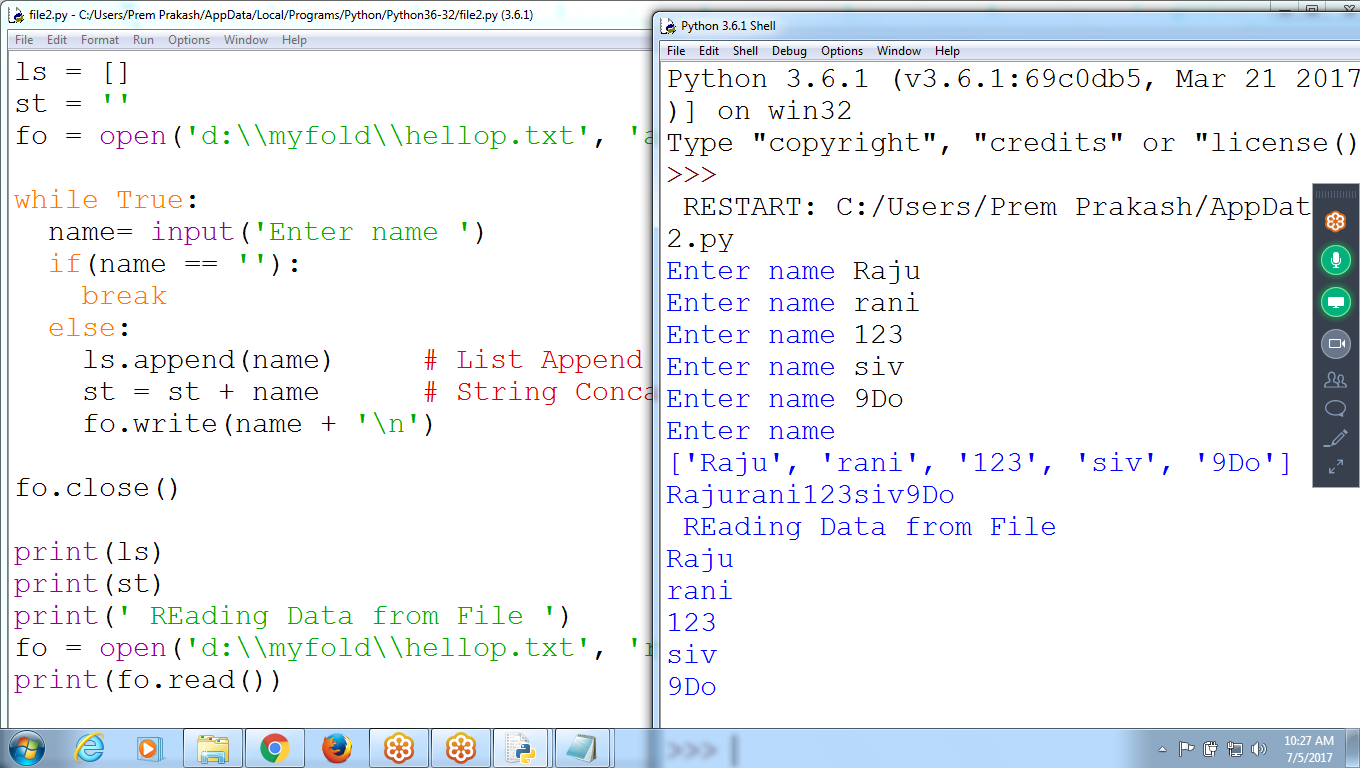
else:

ls.append(name) # List Append

st = st + name # String Concatenation

print(ls)

print(st)



# Write a Program accept name and store onto LIST and to a file

ls = []

st = ''

fo = open('d:\\myfold\\hellop.txt', 'a')

while True:

name= input('Enter name ')

if(name == ''):

break

else:

ls.append(name) # List Append

st = st + name # String Concatenation

fo.write(name + '\n')

fo.close()

print(ls)

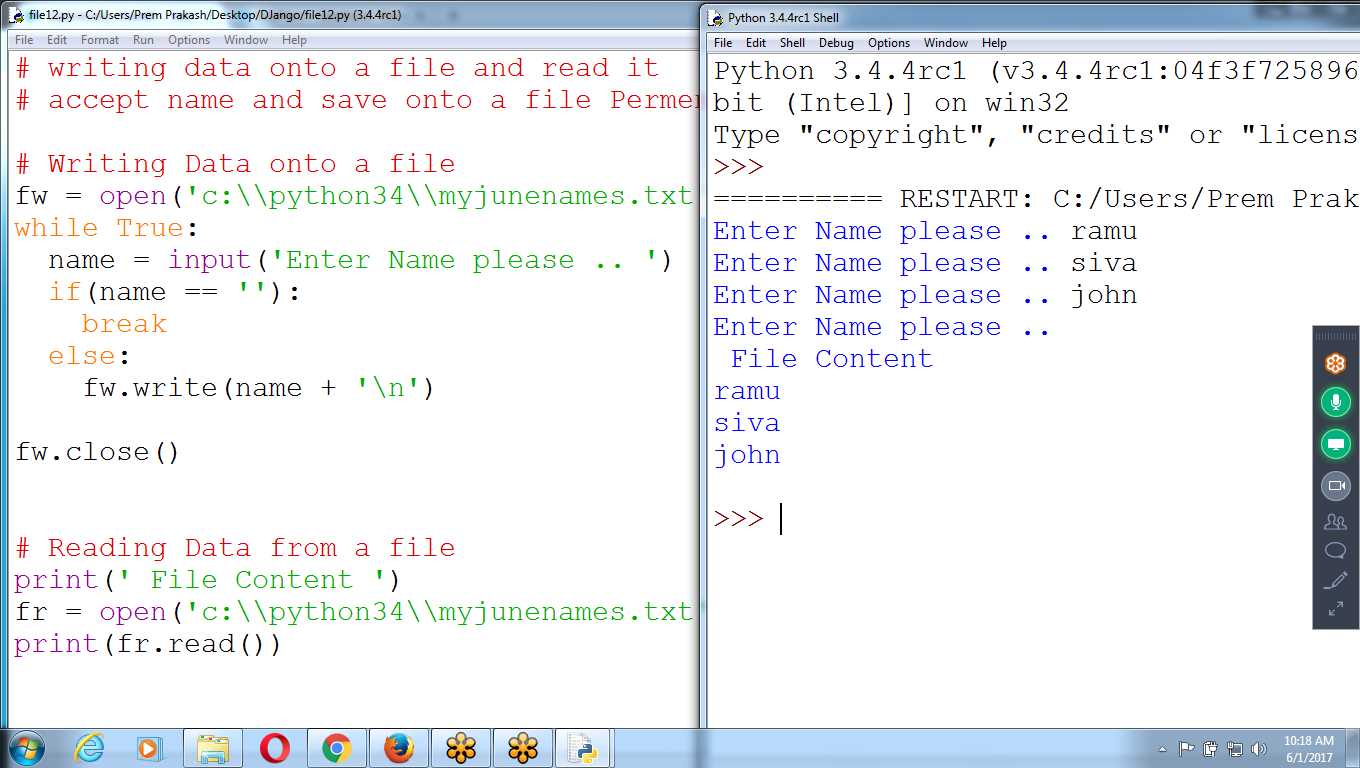
print(st)

print(' REading Data from File ')

fo = open('d:\\myfold\\hellop.txt', 'r')

print(fo.read())

Writing and Reading data from a file



# writing data onto a file and read it

# accept name and save onto a file Permanently

**# Writing Data onto a file**

fw = open('c:\\python34\\myjunenames.txt', 'w')

while True:

name = input('Enter Name please .. ')

if(name == ''):

break

else:

fw.write(name + '\n')

fw.close()

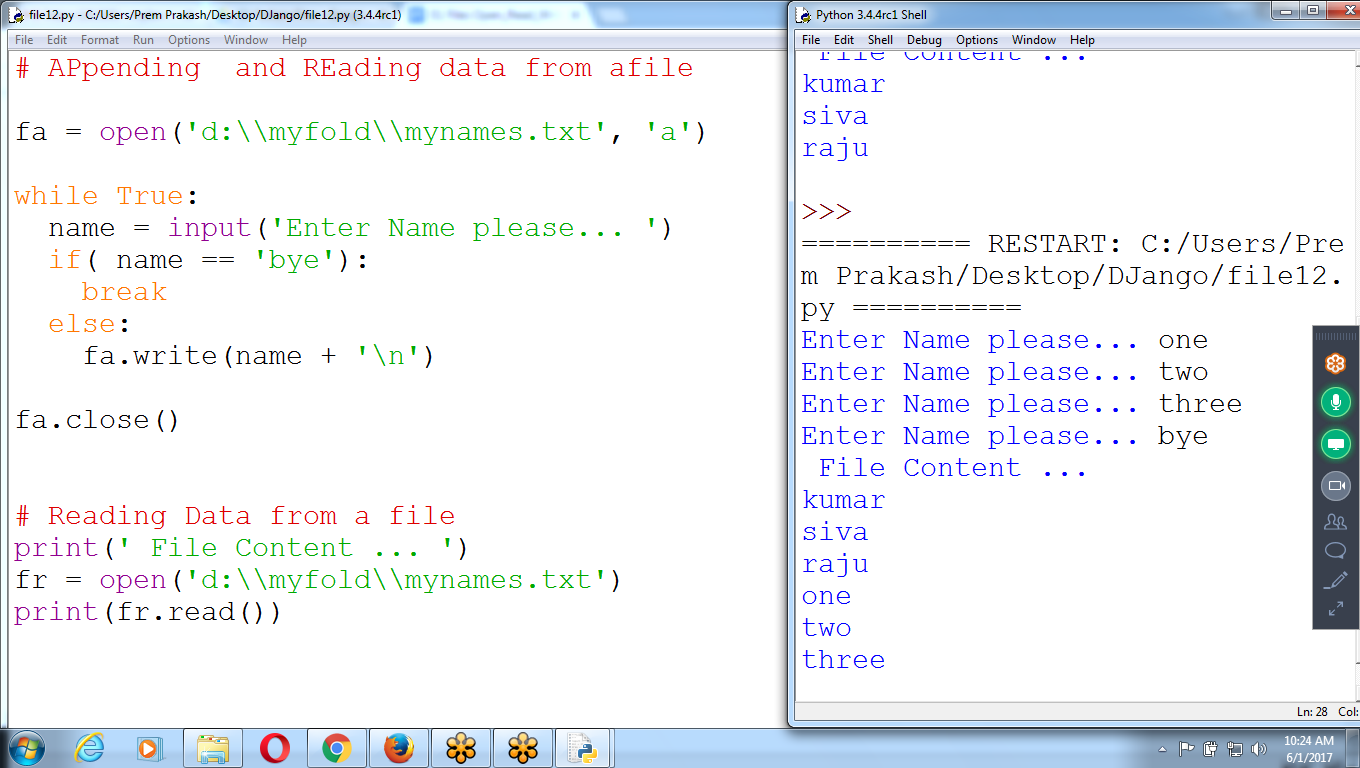
**# Reading Data from a file**

print(' File Content ')

fr = open('c:\\python34\\myjunenames.txt', 'r')

print(fr.read())

Appending and Reading data



**# APpending and REading data from a file**

fa = open('d:\\myfold\\mynames.txt', 'a')

while True:

name = input('Enter Name please... ')

if( name == 'bye'):

break

else:

fa.write(name + '\n')

fa.close()

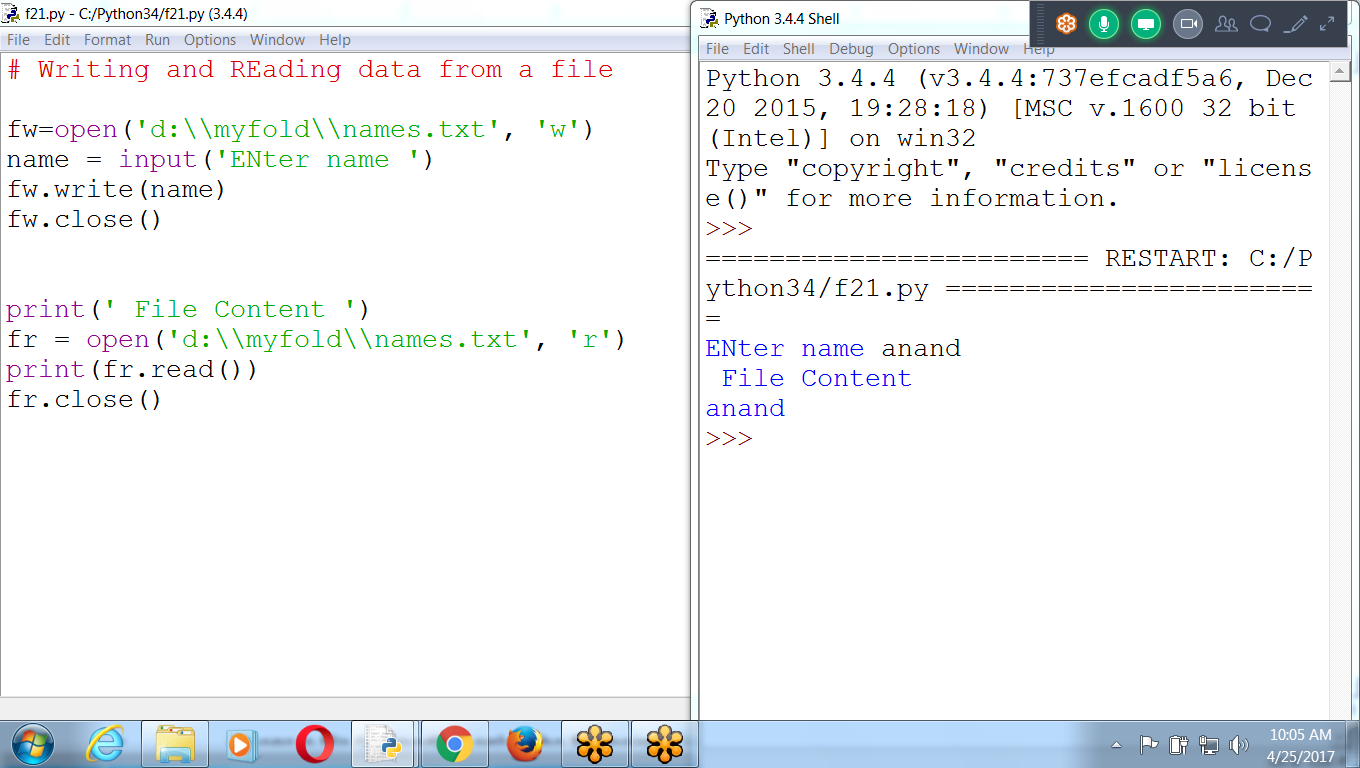
# Reading Data from a file

print(' File Content ... ')

fr = open('d:\\myfold\\mynames.txt')

print(fr.read())

**Writing Data and REading Data from D:\\myfold (folder)**

****

**# Writing and REading data from a file**

**# If file not existing, creates NEW File**

**fw=open('d:\\myfold\\names.txt', 'w')**

**name = input('ENter name ')**

**fw.write(name)**

**fw.close()**

**print(' File Content ')**

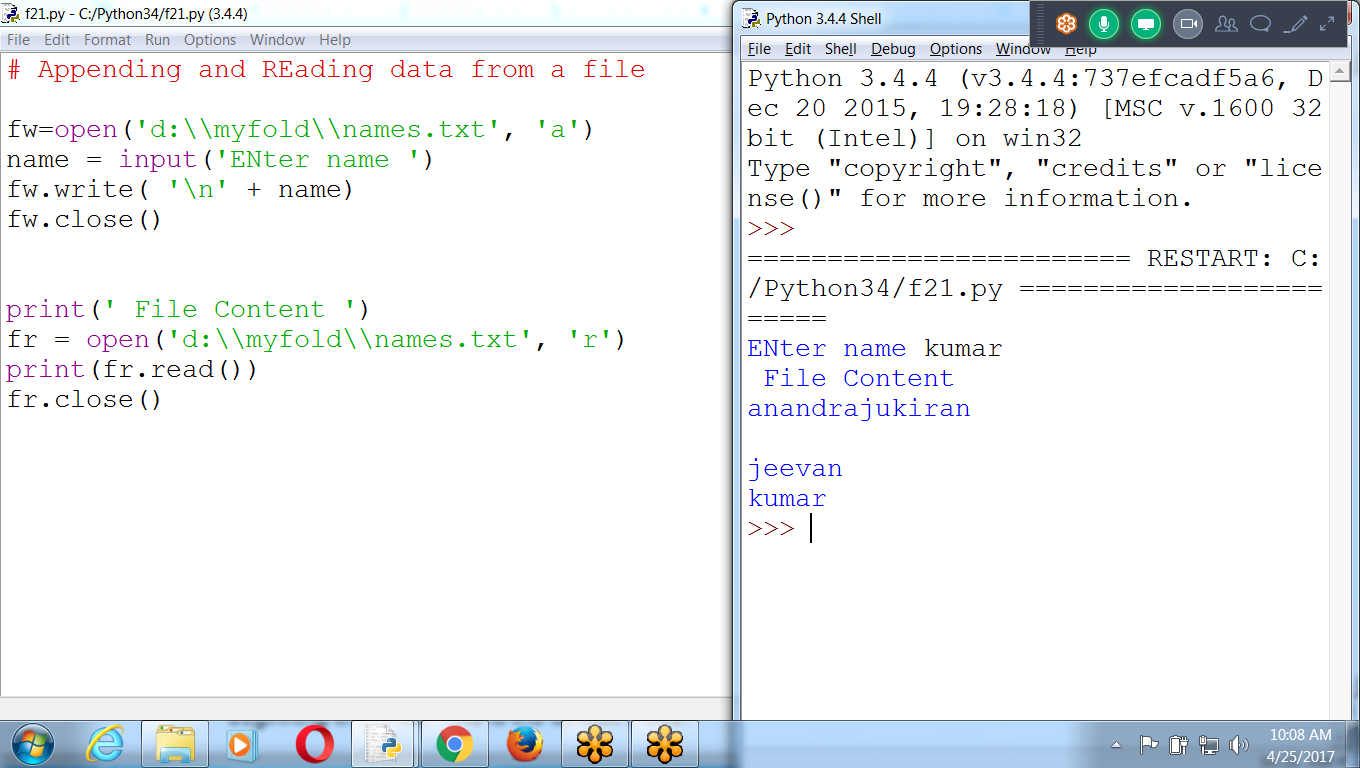
**fr = open('d:\\myfold\\names.txt', 'r')**

**print(fr.read())**

**fr.close()**

**# APpending data and Reading data**

**# append means : End of file adding new content**



# Appending and REading data from a file

fw=open('d:\\myfold\\names.txt', 'a')

name = input('ENter name ')

fw.write( '\n' + name)

fw.close()

print(' File Content ')

fr = open('d:\\myfold\\names.txt', 'r')

print(fr.read())

fr.close()

Different modes of opening a file −

|  |  |
| --- | --- |
| **Modes** | **Description** |
| r | Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode. |
| rb | Opens a file for reading only in binary format. The file pointer is placed at the beginning of the file. This is the default mode. |
| r+ | Opens a file for both reading and writing. The file pointer placed at the beginning of the file. |
| rb+ | Opens a file for both reading and writing in binary format. The file pointer placed at the beginning of the file. |
| w | Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing. |
| wb | Opens a file for writing only in binary format. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing. |
| w+ | Opens a file for both writing and reading. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing. |
| wb+ | Opens a file for both writing and reading in binary format. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing. |
| a | Opens a file for appending. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing. |
| ab | Opens a file for appending in binary format. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing. |
| **a+** | **Opens a file for both appending and reading. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.** |
| ab+ | Opens a file for both appending and reading in binary format. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing. |

## 

## 

## **Using ‘r+’ :: reading and writing at a time**

>>> frw = open('d:\\myfold\\names.txt', **'r+'**)

**>>> print(frw.read())**

anandrajukiran

jeevan

kumarPython Program

**>>> frw.write('Delll Laptop')**

12

>>> print(frw.read())

**>>> frw.seek(0)**  # file pointer moves 0th character

0

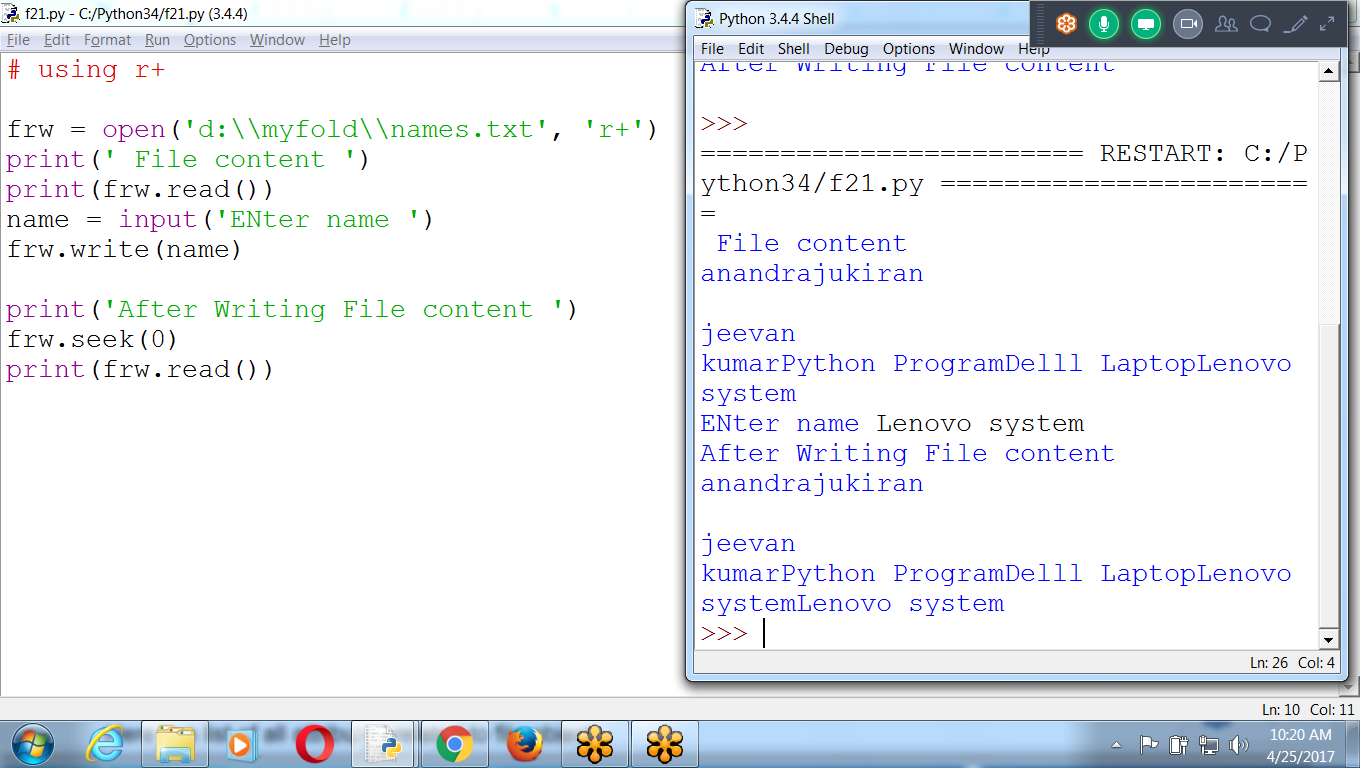
>>> print(frw.read()) # prints data from 0th character

anandrajukiran

jeevan

kumarPython ProgramDelll Laptop

>>>



## **The *file* Object Attributes**

Once a file is opened and get information related to that file.

Here is a list of all attributes related to file object:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| file.closed | Returns true if file is closed, false otherwise. |
| file.mode | Returns access mode with which file was opened. |
| file.name | Returns name of the file. |

>>> frw.name

'D:\\myfold\\names.txt'

>>> frw.mode

'r+'

>>> frw.closed

False

>>>

## **The *read()* Method**

The *read()* method reads a string from an open file.

## **The *close()* Method**

The close() method of a *file* object flushes any unwritten information and closes the file object, after which no more writing can be done.

Python automatically closes a file when the reference object of a file is reassigned to another file. It is a good practice to use the close() method to close a file.

**WITHOUT Closing file**

>>> fw = open('d:\\myfold\\example.txt','w')

>>> fw.write('Good morning ')

13

>>> fr = open('d:\\myfold\\example.txt','r')

**>>> print(fr.read())**

>>>

**Using close()**

>>> fw = open('d:\\myfold\\example.txt','w')

>>> fw.write('Good Morning ')

13

**>>> fw.close()**

>>> fr = open('d:\\myfold\\example.txt','r')

>>> print(fr.read())

Good Morning

>>>

## ***write()* Method**

The *write()* method writes any string to an open file. It is important to note that Python strings can have binary data and not just text.

**The write() method does not add a newline character ('\n')** to the end of the string

## f = open("test.txt") # equivalent to 'r' or 'rt' f = open("test.txt",'w') # write in text mode f = open("img.bmp",'r+b') # read and write in binary mode

## 

## the default encoding is platform dependent. In windows, it is 'cp1252' but 'utf-8' in Linux.

## the default encoding or else our code will behave differently in different platforms.

## Hence, when working with files in text mode, it is highly recommended to specify the encoding type.

## **f = open("test.txt",mode = 'r',encoding = 'utf-8')**

## 

## 

## **Opening Files with the open() Function**

* To open a file with the **open() function**, pass a string path indicating the file to open; it can be either **an absolute or relative path**.
* The open() function returns a **File object.**

Create a text file named *hello.txt* using **Notepad or TextEdit**. Type **Hello world!** as the content of this text file and save it in user home folder.

if you’re using **Windows**, enter the following into the interactive shell: (forward)

>>> **helloFile = open('C:\\Users\\*prem*\\hello.txt')**

If you’re using **OS X,** enter the following into the interactive shell instead: (Back word)

>>> **helloFile = open('/Users/*prem*/hello.txt')**

* When a file is opened in **read mode**, Python lets you only read data from the file;

**you can’t write or modify** it in any way.

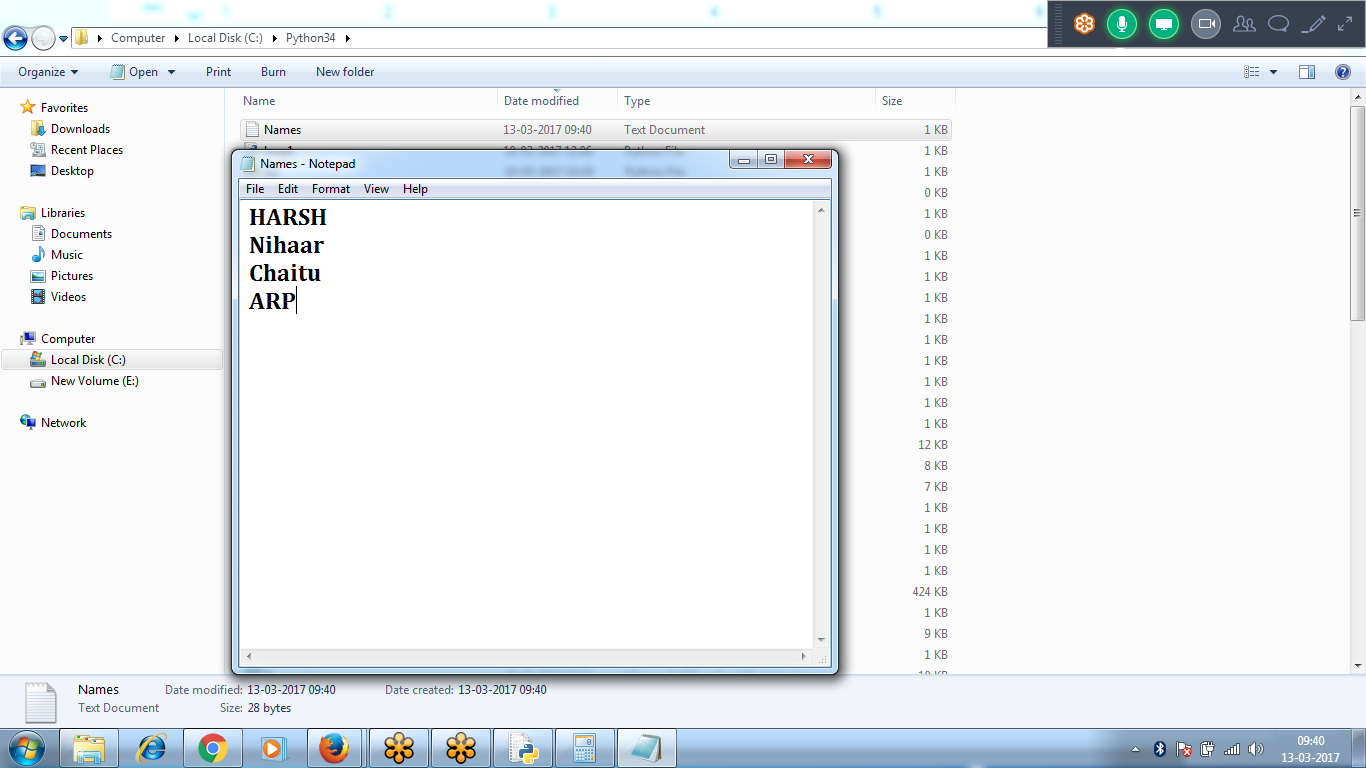
* Read mode is the **default mode** for files you open in Python.
* So open('/Users/prem/ hello.txt', 'r') and open('/Users/prem/hello.txt') do the same thing.
* A File object represents a **file** on your computer; it is simply **another type** of value in Python, much like the lists and dictionaries

## 

## 

## **Reading OPTIONS the Content of the File**

To read the entire contents of a file as a string value, use the File object’s **read()** method.



>>> fr = open('d:\\myfold\\names.txt', 'r')

>>> print(fr.read())

anand learning Python

jeevan knows django

ramu don't know java

>>>

>>> fr = open('d:\\myfold\\names.txt', 'r')

>>> print(fr.read(10))

anand lear

**>>> print(fr.read(20)) # after 10th character to next 20 character**

ning Python

jeevan k

>>>

>>> fr = open('d:\\myfold\\names.txt', 'r')

>>> print(fr.read(10))

anand lear

**>>> fr.close() # file closed can’t access data**

>>> print(fr.read(20))

Traceback (most recent call last):

File "<pyshell#10>", line 1, in <module>

print(fr.read(20))

ValueError: I/O operation on closed file.

>>>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

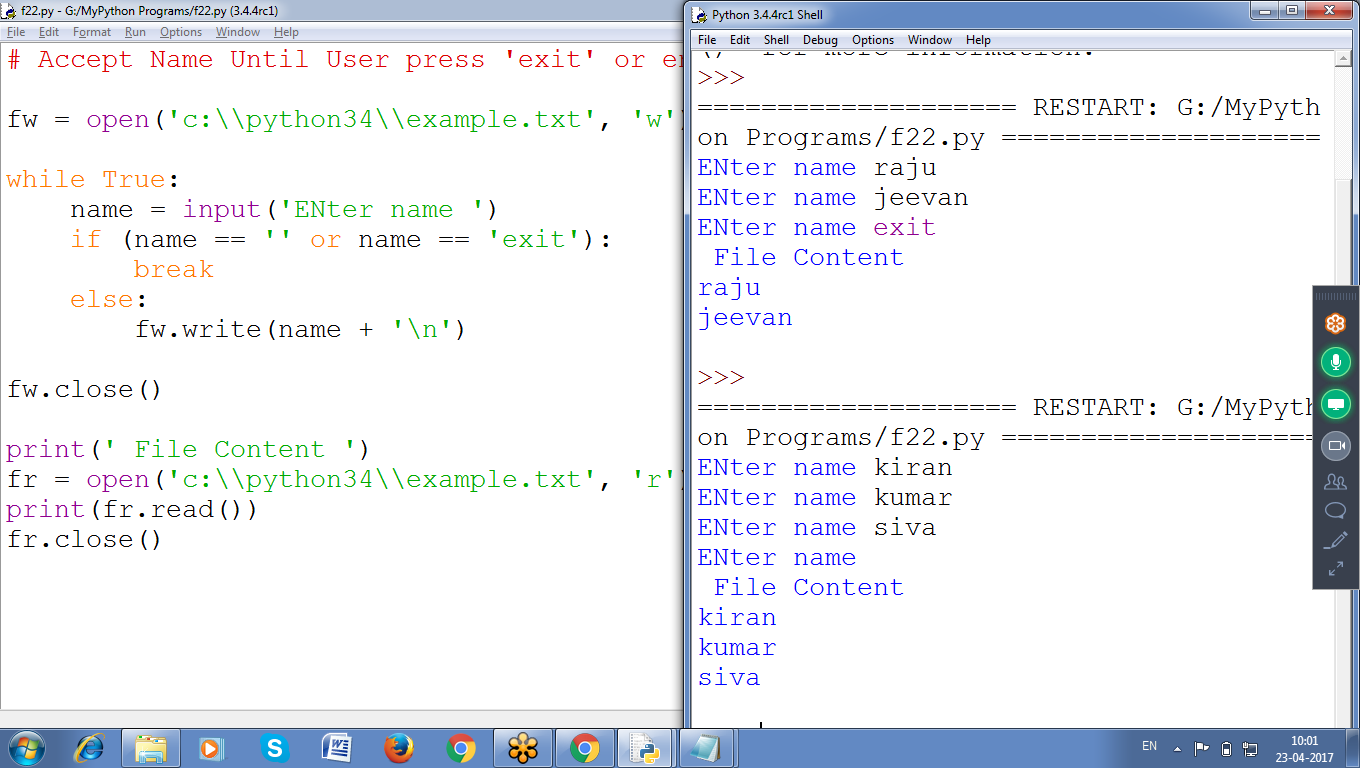
|  |  |
| --- | --- |
| Method | Description |
| close() | Close an open file. It has no effect if the file is already closed. |
| detach() | Separate the underlying binary buffer from the TextIOBaseand return it. |
| fileno() | Return an integer number (file descriptor) of the file. |
| flush() | Flush the write buffer of the file stream. |
| isatty() | Return True if the file stream is interactive. |
| **read(n)** | **Read atmost n characters from the file. Reads till end of file if it is negative or None.** |
| readable() | Returns True if the file stream can be read from. |
| readline(n=-1) | Read and return one line from the file. Reads in at most n bytes if specified. |
| readlines(n=-1) | Read and return a list of lines from the file. Reads in at most n bytes/characters if specified. |
| seek(offset,from=SEEK\_SET) | Change the file position to offset bytes, in reference to from (start, current, end). |
| seekable() | Returns True if the file stream supports random access. |
| tell() | Returns the current file location. |
| truncate(size=None) | Resize the file stream to size bytes. If size is not specified, resize to current location. |
| writable() | Returns True if the file stream can be written to. |
| write(s) | Write string s to the file and return the number of characters written. |
| writelines(lines) | Write a list of lines to the file. |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|  |  |
| --- | --- |
| Method | Description |
| close() | Close an open file. It has no effect if the file is already closed. |
| detach() | Separate the underlying binary buffer from the TextIOBaseand return it. |
| fileno() | Return an integer number (file descriptor) of the file. |
| flush() | Flush the write buffer of the file stream. |
| isatty() | Return True if the file stream is interactive. |
| **read(n)** | **Read atmost n characters from the file. Reads till end of file if it is negative or None.** |
| readable() | Returns True if the file stream can be read from. |
| readline(n=-1) | Read and return one line from the file. Reads in at most n bytes if specified. |
| readlines(n=-1) | Read and return a list of lines from the file. Reads in at most n bytes/characters if specified. |
| seek(offset,from=SEEK\_SET) | Change the file position to offset bytes, in reference to from (start, current, end). |
| seekable() | Returns True if the file stream supports random access. |
| tell() | Returns the current file location. |
| truncate(size=None) | Resize the file stream to size bytes. If size is not specified, resize to current location. |
| writable() | Returns True if the file stream can be written to. |
| write(s) | Write string s to the file and return the number of characters written. |
| writelines(lines) | Write a list of lines to the file. |

**FILES Examples**

**# Accept Name Until User press 'exit' or enter key**



**# Accept Name Until User press 'exit' or enter key**

fw = open('c:\\python34\\example.txt', 'w')

while True:

name = input('ENter name ')

if (name == '' or name == 'exit'):

break

else:

fw.write(name + '\n')

fw.close()

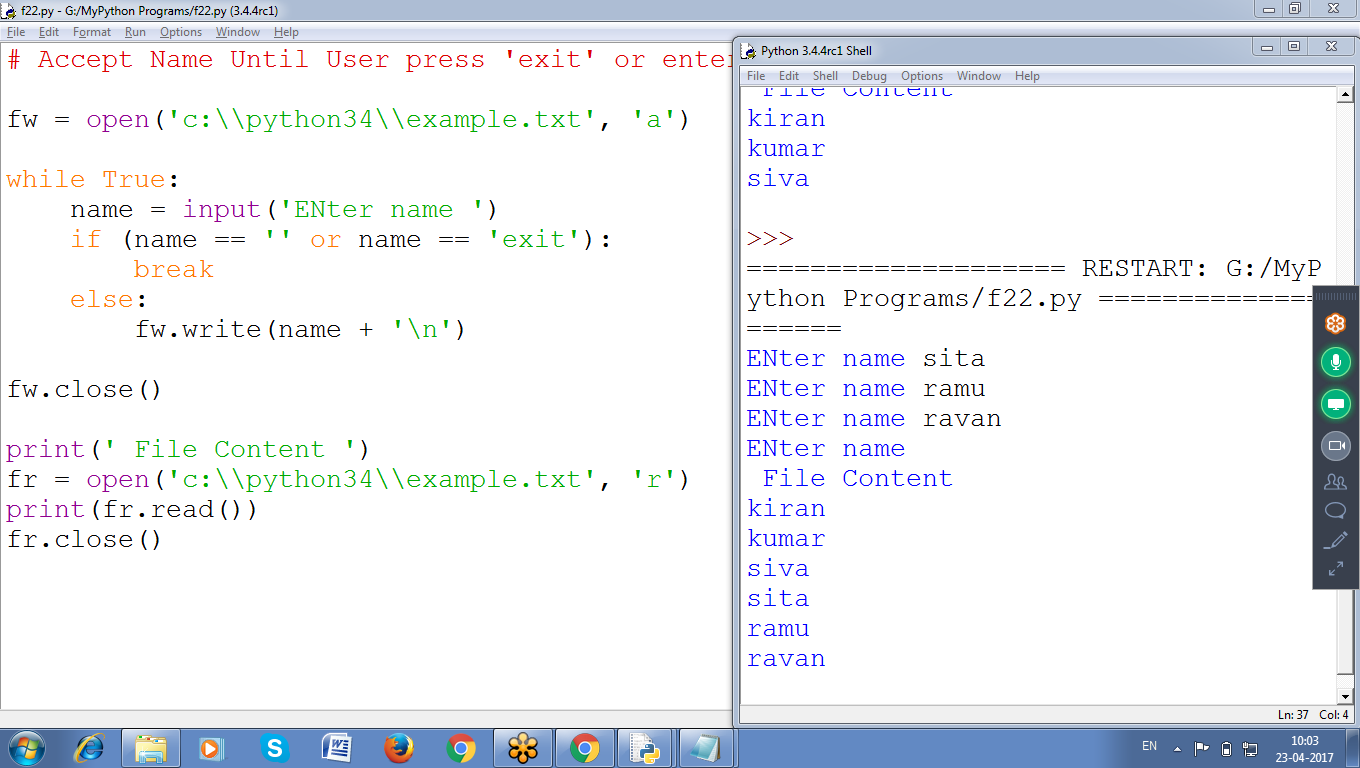
print(' File Content ')

fr = open('c:\\python34\\example.txt', 'r')

print(fr.read())

fr.close()

**Appending Data onto a file**



**# Appending data onto a file**

fw = open('c:\\python34\\myfile4.txt','a')

st = input('Enter any string ')

fw.write('\n'+st )

fw.close()

print(' File Content :: ')

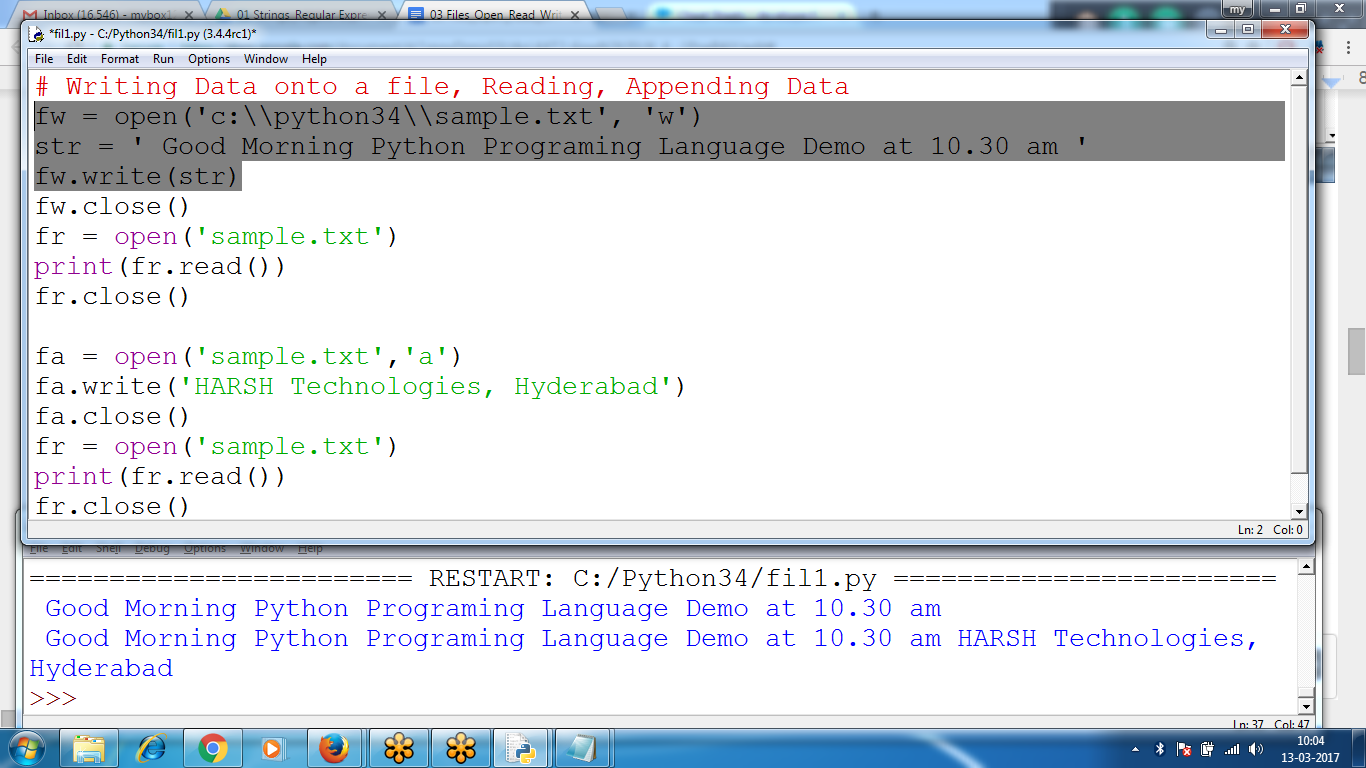
fw = open('c:\\python34\\myfile4.txt', )

print(fw.read())

fw.close()

**Program :: Writing data onto a file, Reading Data from a file and**

**Appending data onto a file**



# Writing Data onto a file, Reading, Appending Data

fw = open('c:\\python34\\sample.txt', 'w')

str = ' Good Morning Python Programing Language Demo at 10.30 am '

fw.write(str)

fw.close()

fr = open('sample.txt')

print(fr.read())

fr.close()

fa = open('sample.txt','a')

fa.write('HARSH Technologies, Hyderabad')

fa.close()

fr = open('sample.txt')

print(fr.read())

fr.close()