File Handling in `Python`

by …

Dr. Pradyumna Kumar Tripathy
Associate Professor & Head, Dept. of CSE,
Silicon Institute of Technology, Bhubaneswar

Goals

By the end of this class you should understand …

- Text File Vs Binary File
- Read & Write operations on file
- Random File Operation
- Reading & Writing CSV files
- Use of pickle module

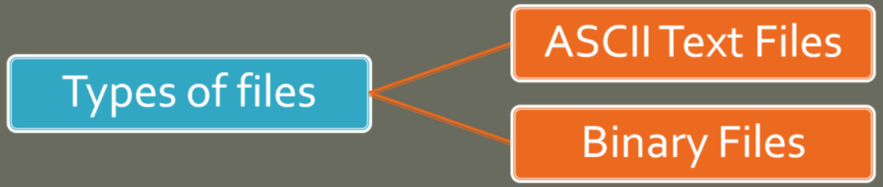
What is a File ...

- A file is a collection of related data that a computers treats as a single unit.
- Computers store files to secondary storage so that the contents of files remain intact when a computer shuts down.
- When a computer –
  - reads a file, it copies the file from the storage device to memory;
  - when it writes to a file, it transfers data from memory to the storage device.

Buffers

- A buffer is a “special work area” that holds data as the computer transfers them to/from memory.
- The physical requirements of the devices can deliver more data for input than a program can use at any one time. The buffer handles the overflow data until a program can use it.
- Moreover, the buffer also holds data until it is efficient to write that data to the storage device for output.

Text Files & Binary Files



- Text File consist of sequential characters divided into lines.
  - Each line terminates with the newline character (`\n`).
- Binary File consist of data values such as integers, floats or complex data types, “using their memory representation.”

So, what’s the difference between text mode and binary mode and which mode to use???

- **Analyze with 3 factors:**
  I. **How newlines (\n) are stored?**
  II. **How end-of-file is indicated?**
  III. **How numbers are stored in the file?**

First factor

- In text mode, a newline character is converted into the carriage return-linefeed combination before being written to disk.
- Likewise, the carriage return-linefeed combination on the disk is converted back into a newline when the file is read by a Python program.
- However, if a file is opened in binary mode, as opposed to text mode, these conversions do not take place.
- In binary mode, each end of line is signified by a carriage return-linefeed combination and is counted as two characters in binary mode.

Carriage Return(\r) moves the cursor to the beginning of the line without advancing to the next line

Line Feed (\n) moves the cursor down to the next line without returning to the beginning of the line

End of Line (EOL) is actually two ASCII characters and is a combination of the \r and \n characters

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Second Factor

- In text mode, a special character EOF whose ASCII value is 26 is inserted after the last character in the file to mark the end of file.
- However, there is no such special character present in the binary mode files to mark the end of file.
- The binary mode files keep track of the end of file from the number of characters present in the directory entry of the file.

Third Factor

- In text mode, the text and numbers are stored as string of characters such that the number 12345 will occupy 5 bytes (1 byte/character).
- Similarly 1234.56 occupies 7 bytes on disk in text mode.
- However, in binary mode the numbers are stored in the same way as they are stored in RAM so that the number 12345 occupies only 2 bytes and 1234.56 occupies only 4 bytes on disk in binary mode.

Basic operations performed on a data file are:

1. Naming a file
2. Opening a file
3. Reading data from the file
4. Writing data in to the file
5. Closing a file

Opening a File

```
File_object = open(r"File_Name", "Access_Mode")
```

File handler is like a cursor, which defines from where the data has to be read or written in the file

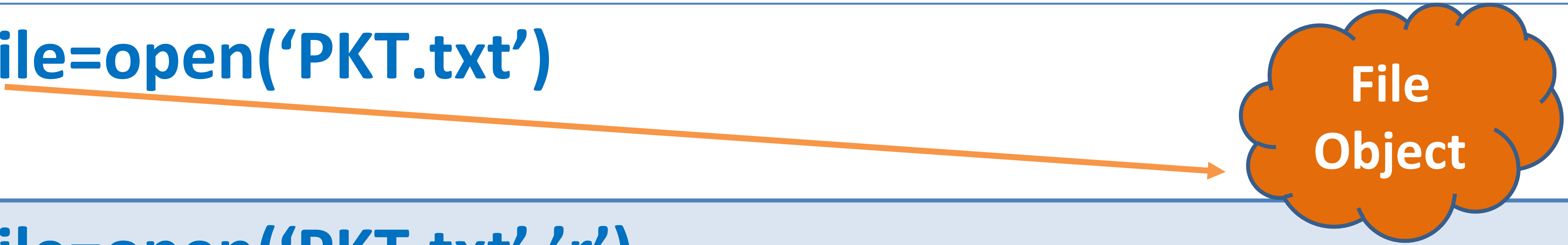
Access modes govern the type of operations possible in the opened file

The file should exist in the same directory as the Python script, otherwise full address of the file should be written.

Access Modes

File Opening Modes	USE
r	Open text file for reading. The handle is positioned at the beginning of the file. If the file does not exist, raises I/O error. This is also the default mode in which the file is opened
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 exist.
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.
r+	Open the file for reading and writing. The handle is positioned at the beginning of the file. Raises I/O error if the file does not exist.
w+	Open the file for reading and writing. For existing file, data is truncated and over-written. The handle is positioned at the beginning of the file.
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.

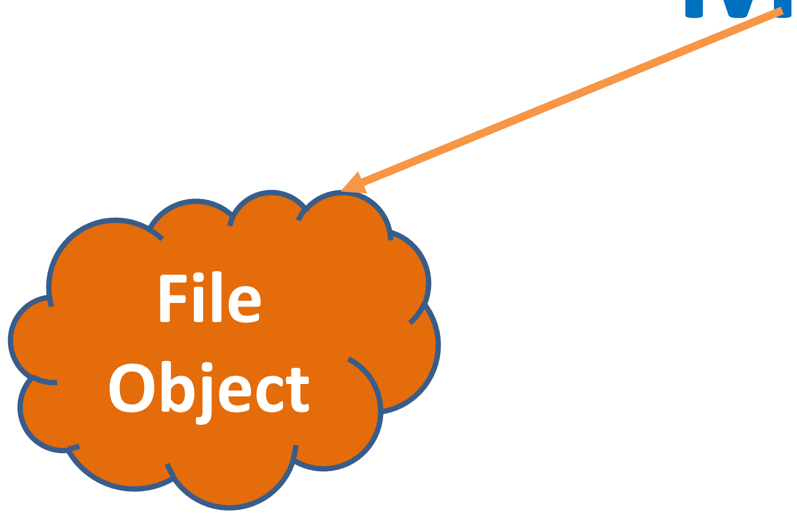
Opening a File

1	Myfile=open(‘PKT.txt’) 
2	Myfile=open(‘PKT.txt’,’r’)
3	Myfile=open(‘c:\\temp\\data.txt’,’r’)
4	Myfile=open(r’c:\temp\data.txt’, ‘r’)

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Closing a File

```
Myfile.close()
```



Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Reading from a File

Syntax:

FileObject= open(“Filename", “AccessMode")

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Reading from a File

```
str=myfile.read(30)
# Next read() reads the next 30 characters from the last read
file(r”E:/poem.txt”,”r”).read(30)
# Reads one line from the file
open(“poem.txt”,”r”).readline()
# reads all line from the file
open(“poem.txt”,”r”).readlines()
# Entire file will be read
open(“poem.txt”,”r”).read()
```

Example: Reading from a File

```
#Reading the content of the File
f = open("C:/Users/Pradyumna/Desktop/demofile.txt", "r")
print(f.read())
```

```
# Python code to illustrate read() mode character wise
f = open("C:/Users/Pradyumna/Desktop/demofile.txt", "r")
print (f.read(5))
print('After Reading Character by character')
```

```
#Reading the content one line at a time
f = open("C:/Users/Pradyumna/Desktop/demofile.txt", "r")
for i in range(1,5):
    print(f.readline(), end=‘ ')
print ('End of loop')
```

Example: Reading from a File

```
#Reading the First Line
f = open("C:/Users/Pradyumna/Desktop/demofile.txt", "r")
print(f.readline())
print('End of First Line')
#Reading the First Two Lines
print(f.readline())
print('End of First Two Line')
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Read a text file and display the number of vowels, consonants, uppercase, lowercase characters in the file

```
vowels = set("AEIOUaeiou")
cons = set("bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ")
infile=open("C:/Users/Pradyumna/Desktop/demofile.txt","r")
countV = 0
countC = 0
countU = 0
countL = 0
for V in infile.read():
    if V in vowels:
            countV += 1
    if V in cons:
            countC += 1
    if V>='A' and V<='Z':
        countU += 1
    if V>='a' and V<='z':
        countL += 1
print("The number of Vowels is: ",countV,"\nThe number of consonants is: ",countC,"\nThe number of
uppercase is: ",countU,"\nThe number of lowercase is: ",countL)
```

Python program to copy one file to other

```
with open("C:/Users/Pradyumna/Desktop/demofile.txt") as f:
    with open("out1.txt", "w") as f1:
        for line in f:
            f1.write(line)
```

```
import shutil
original = r'C:/Users/Pradyumna/Desktop/demofile.txt'
target = r'C:/Users/Pradyumna/Desktop/demofile1.txt'
shutil.copyfile(original, target)
```

tell()

It returns the current position of the file handler

Syntax:

```
fileObject.tell()
```

Example: Reading from a File

```
#Another way of Reading the content one line at a time
f = open("C:/Users/Pradyumna/Desktop/demofile.txt", "r")
p=f.tell()
print('File Pointer at:',p)
f.seek(10)
p=f.tell()
print('File Pointer at:',p)
for x in f:
  print(x)
print ('End of loop')
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

seek()

Moves the file handler to the desired location

Syntax:

fileObject.seek(offset, whence)

offset:position of the read/write pointer within the file

whence:optional and defaults to 0

- **0**- Absolute file positioning
- **1**- seek relative to the current position
- **2**- seek relative to the end of File

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology

Example: Use of seek( )

```
#Use of seek()
f = open('C:/Users/Pradyumna/Desktop/workfile', 'wb+')
f.write(b'0123456789abcdef')
f.close()
f = open('C:/Users/Pradyumna/Desktop/workfile', 'rb+')
f.seek(5)                     # Go to the 6th byte in the file #'5'
print(f.read())               #b'56789abcdef'
f.seek(5)
print(f.read(1))              # b'5'
f.seek(-3, 2)                 # Go to the 3rd byte before the end
print(f.read())               #def
f.seek(-3, 2)
print(f.read(1))              #b'd'
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Writing into a File

```
fileout=open("C:/Users/Pradyumna/Desktop/stud.dat","w")
for i in range (5):
    name=input("Enter name of the student")
    fileout.write(name)
    fileout.write('\n')
fileout.close()
```

Writing into a File

```
fileout=open("C:/Users/Pradyumna/Desktop/stud.dat","w")
list1=[]
for i in range (5):
    name=input("Enter name of the student")
    list1.append(name+'\n')
fileout.writelines(list1)
fileout.close()
```

flush()

It forces the writing of the data on the disc still pending in output buffer

```
f=open("C:/Users/Pradyumna/Desktop/out.log","w+")
f.write("The output is \n")
f.write("My" + "work status" + "is :")
f.flush()
s='OK'
f.write('')
f.write(s)
f.write('\n')
f.write("Finally Over \n")
f.flush()
f.close()
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

```
# Removing EOL character
fn=open("C:/Users/Pradyumna/Desktop/demofile.txt","r")
line=fn.readline()
line=line.rstrip('\n')
print(line)
```

```
# Removing White Spaces
fn=open("C:/Users/Pradyumna/Desktop/demofile.txt","r")
line=fn.readline()
line=line.lstrip()
line=line.rstrip()
print(len(line))
```

Example: Appending and writing to a File

```
f = open(" C:/Users/Pradyumna/Desktop/ demofile.txt", "a")
f.write("Extra Lines written ...")
print()
f.write('My Name is Dr. Pradyumna Kumar tripathy')
f.close()
```

```
#open and read the file after the appending:
f = open(" C:/Users/Pradyumna/Desktop/ demofile.txt", "r")
print(f.read())
```

```
#Open the file "demofile.txt" and overwrite the content:
f = open(" C:/Users/Pradyumna/Desktop/ demofile.txt", "w")
f.write("See.... Contents deleted... This is the new content")
f.close()
```

Example: Appending and writing to a File

```
#Create a new empty file called "myfile.txt":
f = open("C:/Users/Pradyumna/Desktop/myfile.txt", "x")
#gives error if the file already exists
#Create a new file if it does not exist:
f = open("myfile.txt", "w")
```

```
# Python code to illustrate with() alongwith write()
with open("C:/Users/Pradyumna/Desktop/file1.txt", "w") as f:
    f.write("Hi!!!! Every one....")
    f.write('Welcome here ....')
f=open('file1.txt','r')
print (f.read())
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Example: use of split()

```
# Python code to illustrate split() function
with open(" C:/Users/Pradyumna/Desktop/ file1.txt", "r") as file:
    data = file.readlines()
    for line in data:
        word = line.split()
        print (word )
```

```
#File name according to user input
a=input('Please enter the file with path')
f=open(a,'w')
f.write("New File Created")
f.close()
f=open(a,'r')
print(f.read())
f.close()
```

Example: Write & Read data from a File

```
# Program to show various ways to read data from a file.

L = ["This is India \n", "This is Odisha \n", "This is Silicon \n"]

# Creating a file
with open("C:/Users/Pradyumna/Desktop/myfile2.txt", "w") as file1:
        # Writing data to a file
        file1.write("Hi!!!!! \n")
        file1.writelines(L)
        file1.close()                    # to change file access modes

with open(" C:/Users/Pradyumna/Desktop/ myfile2.txt", "r+") as file1:
        # Reading form a file
        print(file1.read())
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

Example: Write & Read data from a File

```
#f= open(" C:/Users/Pradyumna/Desktop/ silicon.txt","w+")
f=open(" C:/Users/Pradyumna/Desktop/ silicon.txt","a+")
for i in range(10):
    f.write("This is line %d\r\n" % (i+1))
#f.close()
#Open the file back and read the contents
f=open(" C:/Users/Pradyumna/Desktop/ silicon.txt", "r")
if f.mode == 'r':
    contents =f.read()
    print (contents)
#or, readlines reads the individual line into a list
fl =f.readlines()
for x in fl:
    print (x)
```

```
import os
```

os.path.exists(path) - Returns True if path or directory does exists.

os.path.isfile(path) - Returns True if path is File.

os.path.isdir(path) - Returns True if path is Directory.

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

```
#Remove the file "demofile.txt":
import os
os.remove("demofile.txt")
```

```
#Check if file exists, then delete it:
import os
if os.path.exists("demofile.txt"):
        os.remove("demofile.txt")
else:
        print("The file does not exist")
```

```
#Remove the folder "myfolder":
import os
os.rmdir("myfolder")
```

```
import os
# Create a directory "test"
os.mkdir("test")
#To change the directory
os.chdir("newdir")
#give location of current directory
os.getcwd()
```

Example: Binary File Writing

```
#Writing
import pickle
output_file = open("C:/Users/Pradyumna/Desktop/myfile.bin", "wb")
myint = 42
mystring = "Hello, world!"
mylist = ["dog", "cat", "lizard"]
mydict = { "name": "Bob", "job": "Astronaut" }

pickle.dump(myint, output_file)
pickle.dump(mystring, output_file)
pickle.dump(mylist, output_file)
pickle.dump(mydict, output_file)

output_file.close()
```

Example: Binary File Reading

```
#Reading
import pickle
input_file = open("C:/Users/Pradyumna/Desktop/myfile.bin", "rb")
myint = pickle.load(input_file)
mystring = pickle.load(input_file)
mylist = pickle.load(input_file)
mydict = pickle.load(input_file)

print("myint = %s" % myint)
print("mystring = %s" % mystring)
print("mylist = %s" % mylist)
print("mydict = %s" % mydict)

input_file.close()
```

Example: Reading .csv File

```
#Reading from csv File
import csv
with open('C:/Users/Pradyumna/Desktop/test1.csv','rt’) as f:
        data = csv.reader(f)
        for row in data:
              print(row)
```

```
#Read a CSV as a Dictionary
import csv
reader=csv.DictReader(open("C:/Users/Pradyumna/Desktop/test1.csv"))
for raw in reader:
    print(raw)
```

Example: Reading .csv File

```
# importing csv module
import csv
file= "C:/Users/Pradyumna/Desktop/test1.csv"
fields = []
rows = []
# reading csv file
with open(file, 'r') as csvfile:
        # creating a csv reader object
        csvreader = csv.reader(csvfile)
        # extracting field names through first row
        fields = next(csvreader)
        # extracting each data row one by one
        for row in csvreader:
              rows.append(row)
# get total number of rows
print("Total no. of rows: %d"%(csvreader.line_num))
```

file object is converted to csv.reader object

```
# printing the field names
print('Field names are:' + ', '.join(field
for field in fields))

# printing first 5 rows
print('\n First 3 rows are:\n')
for row in rows[:3]:
            # parsing each column of a row
            for col in row:
                        print("%10s"%col),
            print('\n')
```

next() method returns the current row and advances the iterator to the next row

Example: Writing into a .csv File

```
# importing the csv module
import csv

# field names
fields = ['Name', 'Branch', 'Year', 'CGPA']

# data rows of csv file
rows = [ ['Pradyumna', 'COE', '2', '9.0'],
         ['Samaleswari', 'COE', '2', '9.1'],
         ['Bikram', 'IT', '2', '9.3'],
         ['Jasaswi', 'SE', '1', '9.5'],
         ['Sushree', 'MCE', '3', '7.8'],
         ['Kasturi', 'EP', '2', '9.1'],
        ['Pamela', 'SE', '3', '9.3']]
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

```
# name of csv file
filename="C:/Users/Pradyumna/Desktop/test10.csv
"


# writing to csv file
with open(filename, 'w') as csvfile:
        # creating a csv writer object
        csvwriter = csv.writer(csvfile)

        # writing the fields
        csvwriter.writerow(fields)

        # writing the data rows
        csvwriter.writerows(rows)
```

Example: Writing .csv File

```
#Write to csv File
import csv
with open('C:/Users/Pradyumna/Desktop/test2.csv', mode='w') as file:
       writer = csv.writer(file, delimiter=',', quotechar='"')
   #way to write to csv file
   writer.writerow(['Programming language', 'Designed by', 'Appeared', 'Extension'])
   writer.writerow(['Python', 'Guido van Rossum', '1991', '.py'])
   writer.writerow(['Java', 'James Gosling', '1995', '.java'])
   writer.writerow(['C++', 'Bjarne Stroustrup', '1985', '.cpp’])
```

```
#Write a csv as a dictionary
import csv
with open('C:/Users/Pradyumna/Desktop/test3.csv', 'w', newline='') as file:
   fieldnames = ['player_name', 'fide_rating']
   writer = csv.DictWriter(file, fieldnames=fieldnames)
   writer.writeheader()
   writer.writerow({'player_name': 'Magnus Carlsen', 'fide_rating': 2870})
   writer.writerow({'player_name': 'Fabiano Caruana', 'fide_rating': 2822})
   writer.writerow({'player_name': 'Ding Liren', 'fide_rating': 2801})
```

Example: Writing .csv File

```
# importing the csv module
import csv

# my data rows as dictionary objects
mydict =[{'branch': 'COE', 'cgpa': '9.0', 'name': 'Nikhil', 'year': '2'},
         {'branch': 'COE', 'cgpa': '9.1', 'name': 'Sanchit', 'year': '2'},
         {'branch': 'IT', 'cgpa': '9.3', 'name': 'Aditya', 'year': '2'},
         {'branch': 'SE', 'cgpa': '9.5', 'name': 'Sagar', 'year': '1'},
         {'branch': 'MCE', 'cgpa': '7.8', 'name': 'Prateek', 'year': '3'},
         {'branch': 'EP', 'cgpa': '9.1', 'name': 'Sahil', 'year': '2'}]

# field names
fields = ['name', 'branch', 'year', 'cgpa']

# name of csv file
filename = "C:/Users/Pradyumna/Desktop/test11.csv"

# writing to csv file
with open(filename, 'w') as csvfile:
    # creating a csv dict writer object
    writer = csv.DictWriter(csvfile,
                 fieldnames = fields)

# writing headers (field names)
    writer.writeheader()

    # writing data rows
    writer.writerows(mydict)
```

Dr. Pradyumna Kumar Tripathy,
Associate Professor, Silicon Institute of Technology, Bhubaneswar

ptripathy@silicon.ac.in

091-9437141874

Dr. Pradyumna Kumar Tripathy

Associate Professor & Head, Dept. of CSE,

Silicon Institute of Technology, Bhubaneswar