**1.Reading and Writing Text Data to file use all attributes (r,w,r+,w+,ab..etc)**

f=open(‘mydata’,’r’)

print(f.read())

f1=open(‘myfile’,’w’)

f1.write(“hi”)

f1=open(‘myfile’,’a’)

f1.write(“my class”)

f=open("kp.txt","a+")  
f.write("krishna.")  
f.seek(0)  
print(f.read())  
  
f=open("kp.txt","r+")  
print(f.read())  
f.write("krishna")  
  
f=open("kp.txt","w+")  
f.write("prasad")  
f.seek(0)  
print(f.read())

**2 Reading and Writing Binary Data**

f=open("D:\IMG\_20190914\_093621.jpg","rb")  
f1=open("ty.bin","wb")  
for j in f:  
 f1.write(j)

**3. Writing to a File That Doesn’t Already Exist**

with open('kp.txt', 'a') as f:  
 if f.tell() == 0:  
 print('a new file or the file was empty')  
 f.write('The header\n')  
 else:  
 print('file existed, appending')  
 f.write('Some data\n')

**4 Reading and Writing Compressed Datafiles**

**5 Zip or Unzip in python**

**import** gzip  
**import** io  
**import** os  
  
output\_file\_name = **'jd\_example.txt.gz'**file\_mode = **'wb'  
  
with** gzip.open(output\_file\_name, file\_mode) **as** output:  
 **with** io.TextIOWrapper(output, encoding=**'utf-8'**) **as** encode:  
 encode.write(**'We can write anything in the file here.\n'**)  
  
print(output\_file\_name,  
 **'contains'**, os.stat(output\_file\_name).st\_size, **'bytes'**)  
os.system(**'file -b --mime {}'**.format(output\_file\_name))  
  
  
  
**import** gzip  
**import** io  
**import** os  
  
read\_file\_name = **'jd\_example.txt.gz'**file\_mode = **'rb'  
  
with** gzip.open(read\_file\_name, file\_mode) **as** input\_file:  
 **with** io.TextIOWrapper(input\_file, encoding=**'utf-8'**) **as** dec:  
 print(dec.read())

**6. Performing I/O Operations on a String**

a=input(**"enter a string"**)  
print(a.upper())  
print(a.capitalize())  
print(a.strip())  
print(a.count(a,0,len(a)))  
print(a.istitle())  
print(a.startswith(**'m'**,0,len(a)))  
print(a.lower())

**7. Printing to a File with a Different Separator or Line Ending**( not sure about the program)

**print('Python', 3, 'Rocks', end=' ')  
print('I love Python')**

or

**print('G', 'F', 'G', sep='')  
print('09', '12', '2016', sep='-')  
print('pratik', 'geeksforgeeks', sep='@')**

**or**

**with open("kp.txt", "r") as my\_file:  
 for line in my\_file:  
 str = line.split()  
 print(str)**

**8.Adding or Changing the Encoding of an Already Open File(**not sure about the program**)**

import io

with io.open(filename,'r',encoding='utf8') as f:

    text = f.read()

# process Unicode text

with io.open(filename,'w',encoding='utf8') as f:

    f.write(text)

**9. open file and read data .. write to some other text file.**

**with** open(**"kp.txt"**) **as** f:  
 **with** open(**"b.txt"**, **"w"**) **as** f1:  
 **for** line **in** f:  
 f1.write(line)

**10 Parsing Simple XML Data**

Items.xml file

<**data**>  
 <**items**>  
 <**item name="expertise1"**>SQL</**item**>  
 <**item name="expertise2"**>Python</**item**>  
 </**items**>  
</**data**>

**import** xml.etree.ElementTree **as** ET  
tree = ET.parse(**'items.xml'**)  
root = tree.getroot()  
  
*# all items data*print(**'Expertise Data:'**)  
  
**for** elem **in** root:  
 **for** subelem **in** elem:  
 print(subelem.text)

**11. Take the data in a Python dictionary and turn it into XML.**

**from** dict2xml **import** dict2xml  
  
data = {**'a'**: 2,  
 **'b'**: {  
 **'c'**: **'as'**,  
 **'f'**: **True**},  
 **'d'**: 7,  
 }  
  
xml = dict2xml(data)  
print(xml)

**12. read an XML document, make changes to it, and then write it back out as XML.**

**import** xml.etree.ElementTree **as** ET  
mytree = ET.parse(**'sample.xml'**)  
myroot = mytree.getroot()  
print(myroot)  
  
**for** description **in** myroot.iter(**'description'**):  
 new\_desc = str(description.text) + **'wil be served'** description.text = str(new\_desc)  
 description.set(**'updated'**, **'yes'**)  
  
mytree.write(**'new.xml'**)

**13. Parsing Huge XML Files Incrementally**

import xml.etree.ElementTree as ET

for event, elem in ET.iterparse("yourXMLfile.xml"):

... do something ...

events=("start", "end", "start-ns", "end-ns")

for event, elem in ET.iterparse("yourXMLfile.xml", events=events):

... do something ..

for event, elem in ET.iterparse("yourXMLfile.xml", events=("start","end")):

if elem.tag == "record\_tag" and event == "end":

print elem.text

elem.clear()

... do something else ...

**14 Python Program to Find the Size (Resolution) of a Image**

def jpeg\_res(filename):

with open(filename,'rb') as img\_file:

# height of image (in 2 bytes) is at 164th position

img\_file.seek(163)

# read the 2 bytes

a = img\_file.read(2)

# calculate height

height = (a[0] << 8) + a[1]

# next 2 bytes is width

a = img\_file.read(2)

# calculate width

width = (a[0] << 8) + a[1]

print("The resolution of the image is",width,"x",height)

jpeg\_res("img1.jpg")

**15. Reading and Writing JSON Data**

Reading json

Person.json

{**"name"**: **"Bob"**,  
**"languages"**: [**"English"**, **"Fench"**]  
}

Try.py

**import** json  
  
**with** open(**'C:/Users/Krishna/PycharmProjects/my project/person.json'**) **as** f:  
 data = json.load(f)  
  
*# Output: {'name': 'Bob', 'languages': ['English', 'Fench']}*print(data)

writing json to a file

**import** json  
  
person\_dict = {**"name"**: **"Bob"**,  
**"languages"**: [**"English"**, **"Fench"**],  
**"married"**: **True**,  
**"age"**: 32  
}  
  
**with** open(**'person.txt'**, **'w'**) **as** json\_file:  
 json.dump(person\_dict, json\_file)

**16 Reading and Writing CSV Data encoded as a CSV file.**

Css.txt

name,department,birthday month  
John Smith,Accounting,November  
Erica Meyers,IT,March

try.py

**import** csv  
  
**with** open(**'css.txt'**, mode=**'r'**) **as** csv\_file:  
 csv\_reader = csv.DictReader(csv\_file)  
 line\_count = 0  
 **for** row **in** csv\_reader:  
 **if** line\_count == 0:  
 print(**f'Column names are {", ".join(row)}'**)  
 line\_count += 1  
 print(**f'\t{row["name"]} works in the {row["department"]} department, and was born in {row["birthday month"]}.'**)  
 line\_count += 1  
 print(**f'Processed {line\_count} lines.'**)

write method

**import** csv  
  
**with** open(**'employee\_file.csv'**, mode=**'w'**) **as** employee\_file:  
 employee\_writer = csv.writer(employee\_file, delimiter=**','**, quotechar=**'"'**, quoting=csv.QUOTE\_MINIMAL)  
  
 employee\_writer.writerow([**'John Smith'**, **'Accounting'**, **'November'**])  
 employee\_writer.writerow([**'Erica Meyers'**, **'IT'**, **'March'**])

**17. write a script that will find a specific string in a .txt file and find how many times the string is avilable....print the o/p into another file.**

stringToMatch = **'specific'**matchedLine = **''***#get line***with** open(**'C:/Users/Krishna/PycharmProjects/my project/kp.txt'**, **'r'**) **as** file:  
 **for** line **in** file:  
 **if** stringToMatch **in** line:  
 matchedLine = line  
 **break***#and write it to the file***with** open(**'a.txt'**, **'w'**) **as** file:  
 file.write(matchedLine)

**18. Python Program to Merge Mails** (When we want to send the same invitations to many people, the body of the mail does not change. Only the name (and maybe address) needs to be changed.

Mail merge is a process of doing this. Instead of writing each mail separately, we have a template for body of the mail and a list of names that we merge together to form all the mails.)

**with** open(**"name.txt"**, **'r'**, encoding=**'utf-8'**) **as** names\_file:  
 *# open body.txt for reading* **with** open(**"body.txt"**, **'r'**, encoding=**'utf-8'**) **as** body\_file:  
 *# read entire content of the body* body = body\_file.read()  
  
 *# iterate over names* **for** name **in** names\_file:  
 mail = **"Hello "** + name + body  
  
 *# write the mails to individual files* **with** open(name.strip() + **".txt"**, **'w'**, encoding=**'utf-8'**) **as** mail\_file:  
 mail\_file.write(mail)