Module C,

Opening a File
>>open(file.txt)

Table 1. Summary of the behaviour of the four most common modes.

Description	mode	open for reading?	open for writing?	Behaviour
read-only	`'r'`	Yes	No	Fails if the file does not exist.
write-only	`'w'`	No	Yes	If the file already exists, the file's contents is truncated (cleared), otherwise it will create an empty file.
exclusive creation	`'x'`	No	Yes	Creates an empty file; fails if the file already exists.
append-only	`'a'`	No	Yes	If the file already exists, all writes are appended to the end of it, otherwise it will create an empty file.

```
with open('file0.txt', 'r+') as f:
    # do something
```

Best Practice: Open files using "with-as"

```
myFile = open('file0.txt')
print(myFile.read())
myFile.close()
```

you can read either the entire file or a specific number of characters using the `read()` method.

```
myFile = open('file0.txt')
print(myFile.read(5))
myFile.close()
```

you can read the file line-by-line using the `readline()` method. Like, `read()` you can optionally choose the number of characters this function reads up to

```
myFile = open('file0.txt')
print(myFile.readline())
print(myFile.readline(4))
myFile.close()
```

writing to a file

```
myFile = open('file0.txt', 'w')
write('goodbye world')
myFile.close()

#append file ‘a’

myFile = open('file0.txt', 'a')
write('goodbye world')
myFile.close()
```

```
.read()   //read the entire characters in the file

.readline(5)  //read only 5 characters

.readlines()  // read and returns in list
```

```
>>> with open('dog_breeds.txt', 'r') as reader:

>>>     for line in reader.readlines():
>>>         print(line, end='')

or

>>> with open('dog_breeds.txt', 'r') as reader:

>>>     # Read and print the entire file line by line
>>>     for line in reader:
>>>         print(line, end='')
```

```
Now for write

with open('dog_breeds.txt', 'r') as reader:

    # Note: readlines doesn't trim the line endings
```

```
    dog_breeds = reader.readlines()

with open('dog_breeds_reversed.txt', 'w') as writer:
    # Alternatively you could use
    # writer.writelines(reversed(dog_breeds))

    # Write the dog breeds to the file in reversed order
    for breed in reversed(dog_breeds):
        writer.write(breed)



with open('data.txt', 'a') as f:

    line1 = "PS5 Restock India \n"
    line2 = "Xbox Series X Restock India \n"
    line3 = "Nintendo Switch Restock India"

    f.writelines([line1, line2, line3])
```

WORKING WITH CSV

reader: reads a row and returns the values as a list of strings
DictReader: reads each row and return the values as an ordered dictionary where the first row is used as the keys

```
import csv
dataFile =  open('customers.csv', 'r')

reader = csv.reader(dataFile)

for row in reader:
    print(row)

dataFile.close()
```

data1=['bivek',28,5092]
with open('/home/thebeast/Downloads/data.csv','a') as file:
  writer_object=csv.writer(file)
  writer_object.writerow(data1)

with open('/home/thebeast/Downloads/data.csv','r') as file:
  reader_object=csv.reader(file)
  for row in reader_object:
    print(row)

JSON

# some JSON:
x = '{ "name":"John", "age":30, "city":"New York"}'

# parse x:
y = json.loads(x)

# the result is a Python dictionary:
print(y["age"])

```
import json

with open('customers.json', encoding='utf-8-sig') as dataFile:
    # load data into dictionary
    jstr = dataFile.read()
    jdata=json.loads(jstr)

print(jdata)

print()
data = jdata["data"]
print(data)

print()
for i in range(len(data)):
    print(data[i])
    print(data[i].get('name'))
```

```
import json

data = [{'name':'John', 'age':52, 'postcode':5002},
        {'name':'Ye','age':18, 'postcode':3005},
        {'name':'Siobhan', 'age':34, 'postcode':2356}
       ]

with open('newFile.json','w') as file:
    json.dump(data,file)
```

XML

open svg file
then parse(file)

x= document.getElementsByTagName(“xyz”)
for n in x:

color=x.getAttribute(“sdsd”)
width=x.getAttribute(“xyxy”)
height=x.getAttribute(“ssss”)

We can also modify the DOM in memory and then write back the file as a new XML document.

```
for eye in ellipse_elements:
    eye.setAttribute("fill", "red")
    eye.setAttribute("rx", "10")
    eye.setAttribute("ry", "10")

with open('red-eye.svg', 'w') as out_file:
    document.writexml(out_file, indent = "  ", addindent = "  ", newl = "\n")
```

NumPY Arrays

import numpy as np
variable=np.array([[][]])
for size of the array: variable.size
for row and column
row,column=variable.shape

np.add → add
np.subtract → subtract
np.negative
np.multiply
np.divide
np.power
np.floor_divide → integer division
np.mod → modulo division

```
import numpy as np

# Addition

a1 = np.arange(9).reshape(3,3)
a2 = np.arange(3)
print('a1=\n', a1)
print('a2=\n', a2)

print('Addition a1 + 5 =\n', a1+5)
print ('np.add a1 + 5 =\n', np.add(a1,5))

# Addition two arrays
print ('np.add a1 + a2 =\n', np.add(a1,a2))
```

Function	Nan-safe Version	Description
np.all()	Not available	Evaluate whether all elements are true
np.any()	Not available	Evaluate whether any elements are true
np.argmax()	np.nanargmax()	Find index of maximum value
np.argmin()	np.nanargmin()	Find index of minimum value
np.max()	np.nanmax()	Find maximum value
np.mean()	np.nanmean()	Compute mean of elements
np.median()	np.nanmedian()	Compute median of elements
np.min()	np.nanmin()	Find minimum value
np.percentile()	np.nanpercentile()	Compute rank-based statistics of elements
np.prod()	np.nanprod()	Compute product of elements
np.std()	np.nanstd	Compute standard deviation
np.sort()	Not available	return a sorted copy of an array
np.sum()	np.nansum()	Compute sum of elements
np.transpose()	Not available	Permute the dimensions of an array
np.var()	np.nanvar()	Compute variance

```
arange([start,] stop[, step,], dtype=None)
```

```
the zeros() function will create an array filled with zeros for
you

zeros(shape, dtype , order)

a1=np.zeros(3)

a1=np.zeros((3), dtype=int)

a1=np.zeros(3,4)

np.concatenate([a1,a2,a3])

np.vstack([a1,a2])
hstack([a1,a2])
```

```
# Example of hsplit ()
# hsplit() always splits on columns
print("m1 =\n", m1)

# split array into 2 even sections of columns
left,right= np.hsplit(m1, 2)
print('left = \n',left)
print('right = \n',right)
```

Pandas

```
# Construct a series from specific indexes from a dictionary

pd.Series({2:'orange', 3:'apple', 1:'nectarine', 4:'strawberry'}, index=[2,4])
```

DataFrame

```
df1 = pd.DataFrame(np.array([[6.5, 90.3], [3.6, 3.2]]))
```

unique value

```
a1.unique()

a1.value_counts()

a1[a1.isin(['white','green']]
```

```
print('Number of null data values in series2 is',series2.isnull().sum())

print('Number of valid data values in series2 is',series2.notnull().sum())
```

Setting Column Labels

```
df1 = pd.DataFrame(np.array([[6.5, 90.3], [3.6, 3.2]]), columns =
['col1','col2'])

df1

df1 = pd.DataFrame(np.array([[6.5, 90.3], [3.6, 3.2]]), columns = ['c1','c2'],
index=['row1','row2'])
```