File Handling

- file:
  - to store some data
- file handling
  - used to do some operations
- open
  - open(filename,mode)
  - with open(filename,mode)
    - read,write,close,append
    - close()
    - read mode - "r"
    - write mode- "w"
    - append mode - "a"

In [2]:
```
data = open("newfile.txt","w")
info = data.write("Good evening")
data.close()
print(info)
```

```
12
```

In [5]:
```
data = open("newfile.txt")
info = data.read()
data.close()
print(info)
print(type(info))
print(type(data))
```

```
Good evening
<class 'str'>
<class '_io.TextIOWrapper'>
```

In [6]:
```
with open("newfile.txt","r") as d:
    info = d.read()
    print(info)
```

```
Good evening
```

In [7]:
```
with open("newfile.txt","w") as d:
    info = d.write("Hello Everyone....")
    print(info)

```

```
18
```

In [14]:
```
with open("newfile2.txt","r") as data:
    g = data.read()
    print(g)
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-14-6dea85d4c44a> in <module>
----> 1 with open("newfile2.txt","r") as data:
      2     g = data.read()
      3     print(g)

FileNotFoundError: [Errno 2] No such file or directory: 'newfile2.txt'
```

In [22]:
```
with open("newfile.txt","a") as k:
    d = k.write("welcome to machine learning workshop")
    print(d)
```

```
36
```

In [16]:
```
with open("newfile.txt","r") as f:
    s = f.read()
    print(s)
```

```
Hello Everyone....welcome to machine learning workshop
```

In [18]:
```
#data = open("newfile.txt")
info = open("newfile.txt").read()
data.close()
print(info)
```

```
Hello Everyone....welcome to machine learning workshop
```

In [20]:
```
data = open("newfile.txt")
info = data.read()
data.close()
print(info)
```

```
Hello Everyone....welcome to machine learning workshop
```

Comprehensions

In [26]:
```
l = []
for i in range(1,101):
    l.append(i)
print(l)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42,
43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62,
63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82,
83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```

List Comprehension

- syntax:
  - newlist = [expression for iter in iterable if condition == TRUE]

In [33]:
```
l = [i for i in range(1,101)]
```

In [34]:
```
print(l)
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22,
23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42,
43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62,
63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82,
83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100]
```

In [36]:
```
s = [x for x in range(1,100) if x%2==0]
print(s)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 4
2, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64, 66, 68, 70, 72, 74, 76, 78, 80,
82, 84, 86, 88, 90, 92, 94, 96, 98]
```

In [43]:
```
f = [x for x in range(77,200) if x%2==0]
print(f)
```

```
[78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 106, 108, 110, 112,
114, 116, 118, 120, 122, 124, 126, 128, 130, 132, 134, 136, 138, 140, 142, 144,
146, 148, 150, 152, 154, 156, 158, 160, 162, 164, 166, 168, 170, 172, 174, 176,
178, 180, 182, 184, 186, 188, 190, 192, 194, 196, 198]
```

In [44]:
```
# 1-100,divisible by 2 and 7
[a for a in range(1,101) if a%2 == 0 and a%7 == 0]

```

Out[44]:
```
[14, 28, 42, 56, 70, 84, 98]
```

In [ ]:
```
# 1-10,
1---odd
2---even
# output====["odd","even","odd","even","odd"]
```

In [ ]:
```

```

In [48]:
```
f = ["Even" if i%2==0 else "odd" for i  in range(11)]
print(f)
```

```
['Even', 'odd', 'Even', 'odd', 'Even', 'odd', 'Even', 'odd', 'Even', 'odd', 'Ev
en']
```

In [51]:
```
out = ["Even" if i%2==0 else "odd" for i  in range(11)]
print(out)
```

```
['Even', 'odd', 'Even', 'odd', 'Even', 'odd', 'Even', 'odd', 'Even', 'odd', 'Ev
en']
```

In [55]:
```
p = "welcome to machine learning using python workshop"
# output = ["welcome","to","machine","leraning","using","python","workshop"]
```

In [56]:
```
p.split()
```

Out[56]:
```
['welcome', 'to', 'machine', 'learning', 'using', 'python', 'workshop']
```

Dictionary comprehension

- syntax :
  - newdict = {key:value for iter in iterable if condition == TRUE}

In [58]:
```
# {1:1,2:4,3:9}
d = {i:i*i for i in range(1,10)}
print(d)
```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}
```

In [59]:
```
d={i:i**2 for i in range(1,101)}
print(d)

```

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121,
12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 18: 324, 19: 361, 20: 40
0, 21: 441, 22: 484, 23: 529, 24: 576, 25: 625, 26: 676, 27: 729, 28: 784, 29:
841, 30: 900, 31: 961, 32: 1024, 33: 1089, 34: 1156, 35: 1225, 36: 1296, 37: 13
69, 38: 1444, 39: 1521, 40: 1600, 41: 1681, 42: 1764, 43: 1849, 44: 1936, 45: 2
025, 46: 2116, 47: 2209, 48: 2304, 49: 2401, 50: 2500, 51: 2601, 52: 2704, 53:
2809, 54: 2916, 55: 3025, 56: 3136, 57: 3249, 58: 3364, 59: 3481, 60: 3600, 61:
3721, 62: 3844, 63: 3969, 64: 4096, 65: 4225, 66: 4356, 67: 4489, 68: 4624, 69:
4761, 70: 4900, 71: 5041, 72: 5184, 73: 5329, 74: 5476, 75: 5625, 76: 5776, 77:
5929, 78: 6084, 79: 6241, 80: 6400, 81: 6561, 82: 6724, 83: 6889, 84: 7056, 85:
7225, 86: 7396, 87: 7569, 88: 7744, 89: 7921, 90: 8100, 91: 8281, 92: 8464, 93:
8649, 94: 8836, 95: 9025, 96: 9216, 97: 9409, 98: 9604, 99: 9801, 100: 10000}
```

Object oriented programming(oops)

In [60]:
```
a = "apssdc"
print(type(a))
```

```
<class 'str'>
```

class:

- Blueprint of an object
- collection of functions/methods and variables

In [61]:
```
print(dir(str))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__',
'__eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewa
rgs__', '__gt__', '__hash__', '__init__', '__init_subclass__', '__iter__', '__l
e__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__', '__new__', '__reduce
__', '__reduce_ex__', '__repr__', '__rmod__', '__rmul__', '__setattr__', '__siz
eof__', '__str__', '__subclasshook__', 'capitalize', 'casefold', 'center', 'cou
nt', 'encode', 'endswith', 'expandtabs', 'find', 'format', 'format_map', 'inde
x', 'isalnum', 'isalpha', 'isascii', 'isdecimal', 'isdigit', 'isidentifier', 'i
slower', 'isnumeric', 'isprintable', 'isspace', 'istitle', 'isupper', 'join',
'ljust', 'lower', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind', 'rind
ex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split', 'splitlines', 'startsw
ith', 'strip', 'swapcase', 'title', 'translate', 'upper', 'zfill']
```

In [65]:
```
class Person:
    age = 20
    def greet(self):
        print("Hello")
print(Person.age)
Person.greet

```

```
20
```

Out[65]:
```
<function __main__.Person.greet(self)>
```

- object:
  - instance of class

In [66]:
```
class Person:
    age = 20
    def greet(self):
        print("Hello")
p = Person()
p.age
```

Out[66]:
```
20
```

In [67]:
```
p.greet()
```

```
Hello
```

Constructor

In [71]:
```
class Apssdc:
    age = 90
    def __init__(self):
        print("i am a constructor")
d = Apssdc()
d.age
```

```
i am a constructor
```

Out[71]:
```
90
```

Inheritance:

- Inherits the properties and methods of parent class

In [73]:
```
class A:#base class /parent class
    def details(self,name):
        self.name = name
        print(self.name)

class B(A):#derived class/child class
    def details1(self,number):
        self.number=number
        print(self.number)
z = B()
z.details("alekhya")
z.details1("34567890")
```

```
alekhya
34567890
```

Multiple inheritance

- One class acquires the properties of one or more parent classes

In [75]:
```
class A:#base class /parent class
    def details(self,name):
        self.name = name
        print(self.name)

class B:#parent class
    def details1(self,number):
        self.number=number
        print(self.number)
class C(A,B):# child class/derived class
    def details2(self):
        print("i am derived class")

s = C()
s.details("apssdc")
s.details(567890)
s.details2()
```

```
apssdc
567890
i am derived class
```

Multilevel Inheritance

In [78]:
```
class A:
    def display():
        print("hello everyone")
class B(A):
    def display1():
        print("good evening")
class C(B):
    def display2():
        print("welcome to workshop")
c = C
c.display1()
```

```
good evening
```

Hierachical inheritance

- one to many

In [79]:
```
class A:
    def display():
        print("hello everyone")
class B(A):
    def display1():
        print("good evening")
class C(A):
    def display2():
        print("welcome to workshop")
x = B
x.display1()
x.display()
y = C
y.display()
y.display2()
```

```
good evening
hello everyone
hello everyone
welcome to workshop
```

Hybrid Inheritance

- it is a combination of multilevel inheritance and multiple inheritance

In [81]:
```
class A:
    def display():
        print("hello everyone")
class B(A):
    def display1():
        print("good evening")
class C(B):
    def display2():
        print("welcome to workshop")
class D(A):
    def display3():
        print("machine learning")
x = C
x.display()
x.display1()
x.display2()

y = D
y.display()
y.display3()
```

```
hello everyone
good evening
welcome to workshop
hello everyone
machine learning
```

In [ ]:
```
1
```