In [ ]:
```
# Day-11
        #Agenda:
                1. File handling (continue)
                2. Sets in Python
                3. Problem Solving.
```

In [ ]:
```
# File :
 its named location on disk to store related information  in a permanent way.

    #File Handling???
      - To Perform some operations on files is called File Handling.
 # why we use ???
    - To store the Data in a permanent Way.
    - To avoids time consuming.
    - To overcome loss of data.
```

In [ ]:
```
#Types of files:
1. Text files ( Deals with Text Data-strings)
2. Binary files ( Deals with media files-mp3,source files,mp4,image files)

examples of Text:
    Web standards: html, xml,css,json
    source code: c,js,.py
    tabular data : csv,.xsl

examples of binary:
    Documents : .pdf,.doc,

    image files: .png,.jpg
                            video and audio .mp3,.mp4
```

In [ ]:
```
#File Operations:
For create or open a file we have to use open() predefined function.

we have 4 types of Operations:
     #operation           #Methods                              #Modes

    1. open--------     open("filename","mode")            "x"- creating the file

    2. read --------    read(), readline(),readlines()       "r"- for read the data

    3. write -----     write(),writelines()                 "w"  - for write contents to the file

                                                            "a"  - for write data to old
                                                                   file at the end


    4. close---------close()

    5. rename ---------rename()

    6. delete----------delete()
```

In [12]:
```
#How to create files?
 #Syntax: open("filename",mode")
#f= open("sample.txt","x")
f = open("sample.txt","w")
f.write("This is Sample File we just created")
f = open("sample.txt","r")
f.read()
f.close()
```

In [14]:
```
#File Artibutes:
print(f.closed)
print(f.mode)
print(f.name)
```

```
True
r
sample.txt
```

In [15]:
```
#close() with try-catch block
try:
    file = open("sample.txt","r")
   #do file operations
finally:
    f.close()
```

In [40]:
```
#auto close using "with" keyword:
with open("new.txt","w") as f:
    f.write("Hello Good evening to All This is from Apssdc\n")
    f.write("This is Online Python Programming\n")
    f.write("Today we discussing about file Handing\n")
    f.writelines(["New line added by writelines\n","End line of the life"])

with open("new.txt","r") as f:
    #print(f.read(100))   #its reads first 5 characters
    #print(f.readline())  # its reads data line by line
    #print(f.readline())
    ##print(f.readline())
    #print(f.readline())
    #print(f.readline())
    #print(f.readline())
    #print(f.readline())
    #print(f.readlines())  #its reads data in form of list of lines
    print(f.read())    # its reads entire data at once
    print(f.read())
```

```
Hello Good evening to All This is from Apssdc
This is Online Python Programming
Today we discussing about file Handing
New line added by writelines
End line of the life
```

In [ ]:
```
```

In [59]:
```
#apend operation:
with open("new.txt","a") as p:
    p.write("This is new line added by apend mode\n")
    p.writelines(["Today we completed file handling","Thank you to All"])
with open("new.txt","r") as p:
    print(p.read())
```

```
This is new line added by apend mode
Today we completed file handlingThank you to AllThis is new line added by apend mode
Today we completed file handlingThank you to AllThis is new line added by apend mode
Today we completed file handlingThank you to AllThis is new line added by apend mode
Today we completed file handlingThank you to AllThis is new line added by apend mode
Today we completed file handlingThank you to All
```

In [61]:
```
import os
os.getcwd()
```

Out[61]:
```
'C:\\Users\\Mission Impossible\\Desktop\\Python-Online Workshop Content\\Day-6 (Functions And Problem Solving) [29-08-2020]'
```

In [65]:
```
#renaming a file
os.rename("new.txt","newfile.txt")
```

In [70]:
```
with open("newfile.txt","r") as f:
    print(f.read())
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-70-d74c33fb84fe> in <module>
----> 1 with open("newfile.txt","r") as f:
      2     print(f.read())

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

In [69]:
```
#Delete a file
os.remove("newfile.txt")
```

In [85]:
```
#tell() and seek() method():
with open("sample.txt","r") as f:
    #print(f.read())
    print(len(f.read(5)))
    position = f.tell()  #which tells the current position of cursor point
    print("current position:",position)
    position = f.seek(0,0)  #which sets the cursor point at origin
    print("current position:",position)
    print(f.read(20))
    position = f.seek(0,1)  #which sets the cursor point at previous position
    print("current position:",position)
    print(f.read())
    position = f.seek(0,2)  #which sets the cursor point at the end of file.
    print("current position:",position)
```

```
5
current position: 5
current position: 0
This is Sample File
current position: 20
we just created
current position: 35
```

In [ ]:
```
#Sets in Python:
# KeyPoints:
    1. Its doest follow the ordering
    2. Its having unique elements ( Not Allow duplication of Values)
    3. Its Mutable itsself ( we can add or remove elements)
    4. we cant perform indexing or slicing operation on sets
    5. Its used to remove the duplicate values from the list and tuples.
```

In [87]:
```
#How to create sets?
# Create a empty sets
s = set()   # empty set
print(s)
print(type(s))
```

```
set()
<class 'set'>
```

In [90]:
```
s = {"surya","python","aps"}
type(s)
```

Out[90]:
```
set
```

In [97]:
```
# Set with values
s1 = {"abc",10,20,50,60,"python",50,10,"python"}
for s in s1:
    print(s,end=" ")
```

```
10 python 50 20 60 abc
```

In [125]:
```
#add or remove elements:
s1 = {"abc",10,20,50,60,"python",50,10,"python"}
s1.add(5000)
s1.add("xyz")
#s1.remove(50)
#print(s1)
#discard()
#s1.discard(5000)
#print(s1)
#s1.discard(1000)
#s1.add(500)
s1.pop()
s1.pop()
s1.pop()
s1.pop()
s1.pop()
s1.pop()
s1.pop()
s1.pop()
s1.pop()
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-125-934b3af68c33> in <module>
     18 s1.pop()
     19 s1.pop()
---> 20 s1.pop()

KeyError: 'pop from an empty set'
```

In [105]:
```
#Set opeations or methods
print(dir(set),end="")
```

```
['__and__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge_
_', '__getattribute__', '__gt__', '__hash__', '__iand__', '__init__', '__init_subclass__', '__ior__', '__isub_
_', '__iter__', '__ixor__', '__le__', '__len__', '__lt__', '__ne__', '__new__', '__or__', '__rand__', '__reduc
e__', '__reduce_ex__', '__repr__', '__ror__', '__rsub__', '__rxor__', '__setattr__', '__sizeof__', '__str__',
'__sub__', '__subclasshook__', '__xor__', 'add', 'clear', 'copy', 'difference', 'difference_update', 'discar
d', 'intersection', 'intersection_update', 'isdisjoint', 'issubset', 'issuperset', 'pop', 'remove', 'symmetric
_difference', 'symmetric_difference_update', 'union', 'update']
```

In [135]:
```
#copy,union,intersection,difference
#copy
s1 = {40,45,50,60,80,150,100}
s2 = {45,50,90,150,250}
s1=s1.copy()
print(s1)
#union operation
print(s1|s2)  #union
print(s1.union(s2))  #union
#intersection
print(s1&s2)
print(s1.intersection(s2))
```

```
{80, 50, 100, 150, 40, 60, 45}
{100, 40, 250, 45, 80, 50, 150, 90, 60}
{100, 40, 250, 45, 80, 50, 150, 90, 60}
{50, 45, 150}
{50, 45, 150}
```

In [138]:
```
#difference and symmetric difference:
s1 = {40,45,50,60,80,150,100}
s2 = {45,50,90,150,250}
#difference
print(s1-s2)
print(s1.difference(s2))
print(s2-s1)
#symmetric -difference
print(s1^s2)
print(s1.symmetric_difference(s2))
```

```
{40, 80, 100, 60}
{40, 80, 100, 60}
{90, 250}
{100, 90, 40, 80, 250, 60}
{100, 90, 40, 80, 250, 60}
```

In [ ]:
```
#Update:
Update() method updates a set, adding items from other iterables.
Update() returns the None object.
#syatax:
s.update(iterable)
s.update(iter1,iter2,iter3)
```

In [142]:
```
s1 = {20,30}
s2 = {10}
result = s1.update(s2)
print(result)
print(s1)
```

```
None
{10, 20, 30}
```

In [152]:
```
#Set boolen methods:
s1 = {40,60,80,100}
s2 = {45,50,90,60,150,250,40,60,80,100}
#disjoint()
print(s1.isdisjoint(s2))
#issubset()
print(s1.issubset(s2))
print(s2.issuperset(s1))
```

```
False
True
True
```

In [ ]:
```

```