TRICKS IN PYTHON - D. Bhuvanesh

→ Print () → Without Any Argument,
It just prints new line character

To find index of a character in string

1. find () → S = "Leaning"
```
print ( s.find ("nin"))
```
O/P : 3

2. index () → S = "Leaning"
```
print (S. index ("ng"))
```
O/P : 5

→ Counting substring in the given string :

→ S.count ()
```
S= 'abababa'
print (s.count ('a'))
print (s.count ('ab'))
```
O/P : 4
3

→ Replacing a string :

s. replace (oldstring, newstring)

Splitting of Strings :

```
S= "22-08-2018"
S1= S.split("-")
print (S1)
```

O/P :
```
22
08
2018
```

Joining of String

Syntax: S= Separator .join (group of strings)

Eg:
```
t= ('sunny', 'bunny', 'chinny')
S= '-'.join(t)
print(s)
```

O/P :
```
Sunny-bunny-chinny
```

To check type of characters present in string :

① isalnum () ② isalpha() ③ isdigit()
④ islower () ⑤ isupper() ⑥ istitle()
⑦ isspace ().

Slicing :

Syntax : list 2 = list 1 [Start : Stop : step]

→ To insert item at specified index position

insert () func :-

{ Syntax : . insert (index, element)

```
n = [1, 2, 3, 4, 5]
n. insert (1, 188)
print (n)
```

O/P : [1, 188, 2, 3, 4, 5]

→ Pop () function :

n. pop (index) ⇒ To remove & return element present at specified index

n. pop () ⇒ To remove & return last element of the list .

Eg :
```
n = [10, 20, 30, 40]
n. pop () #40
n. pop (1) #20
print (n)
```

O/P : [10, 20]

→ reverse () :

```
n = [10, 20, 30]
n. reverse ()
print (n)
```

O/P : [30, 20, 10]

Sort() function :-

```
n=[20,5,15,10]
n.sort()
print(n)
```

O/P : [5, 10, 15, 20]

```
S=["Dog", "Banana", "Apple"]
S.sort()
Print(S)
```

O/P : ['Apple', 'Banana', 'Dog']

To sort in reverse (Descending) order :-

```
n=[40,10,30,20]
n.Sort(reverse=True)
print(n)
```

O/P : [40, 30, 20, 10]

Clear() funtion :-

```
n=[1,2,3,4]
n.Clear()
print(n)
```

O/P : []

SET = { }

Important functions of Set :

1. add (x)
2. Update (x, y, z)
3. Copy ()
4. Pop () -> Pops out last element
5. Symmetric difference ()

   -> Returns elements present in either x (or) y but not both.

```
x = {10, 20, 30}
y = {10, 20, 40, 50}
print (x. symmetric_difference (y))
```

O/P : {30, 40, 50}

6. remove (x) -> To remove element
7. discard (x) -> It removes specified element
8. Clear () -> To clear all elements in set.

Mathematical operations on Set

1. **Union()** → To return all elements present in both sets.

   x = {10, 20, 30, 40}

   y = {10, 20, 30, 50}

   print(x.union(y)) → {10, 20, 30, 40, 50}

   print(x|y) → {10, 20, 30, 40, 50}

Note: We can use x.union(y) (or) x|y

2. **intersection()** → Returns common elements from both sets.

   Syntax: $\boxed{\text{x.intersection(y) (or) x \& y}}$

3. **difference()** → Returns elements present in x but not y

   Syntax: $\boxed{\text{x.difference(y) (or) x-y}}$

Dictionary Data Structure:

=> Syntax: d={} (or) d=dict()

d={key:'value'}

To update a Dictionary

d[key]=value

-> d={100:'Pavan', 200:'Bhu'}

d[300]= Vysh

Print(d) => {100:'Pava', 200:'Bhu', 300:Vysh}

=> To delete: $\boxed{\text{del d[key]}}$

Important functions in dictionaries:

1. get() -> d.get(key) -> Return the value for Specified key

2. Pop(): d.pop(key) -> Removes the value

3. keys(): d.keys() -> Returns all keys

4. values(): -> Returns all values

5. items(): Returns list of tuples representing key-value pairs

[(k,v), (k,v), (k,v)]

Function :

```
def add (a, b):
    return (a+b)
print ( add (5,7))
```

O/P : 12

Lamda function: Anonymous function, which can take any sort of argument.

Eg:
```
S= lambda n: n*n
   print (S(2))
```
O/P: 4

```
S= lambda n: n**n
    print (S(3))
```
O/P: 27

```
S= lambda a,b: a+b
   print (S(3,2))
```
O/P : 5

To find biggest num using lambda

```
S= lambda a,b: a if a>b else b
print (S(10,20))
```
O/P : 20

map() function: For every element present in the given sequence, apply some functionality & generate new element with required modification. For this requirement we should go for map() function

Syntax: map (function, sequence)

Eg: Without lambda

```
l = [1, 2, 3, 4, 5]
def doublit(x)
        return 2*x
l1 = list(map(double it, l))
print (l1)
```

O/P : [2, 4, 6, 8, 10]

With lambda :

```
l = [1, 2, 3, 4, 5]
l1 = list(map(lambda x: 2*x, l))
print (l1)
```

O/P : [2, 4, 6, 8, 10]

filter() function: It is same as the map() function, but it gives actual values as output, where as map () gives true or false

Eg: l=[12, 14, 1, 5, 6, 98]

```
l1 = list(map(lambda x: x%2==0, l))
print(l1)
```

O/P: True, True, False, Fal, True, True

where as:

```
l1 = list(filter(lambda x: x%2==0, l))
print(l1)
```

O/P: [12, 14, 6, 98]

Note: To filter values, go for filter() function, to perform certain Modification to values, Go for map() function

Modules :

A group of functions, variables & classes saved to a file, which is nothing but Module

Math Module :

① Sqrt (x) ③ floor (x) ⑤ log(x) ⑦ Tan(x)
② Ceil (x) ④ fabs (x) ⑥ Sin (x)

Eg:
```
from math import*
print (sqrt (4))
print ( ceil (10.1))
print (floor (10.1))
print (fabs (-10.6))
print (fabs (10.6))
```

O/P:
```
2.0
11
10
10.6
10.6
```

Random Module :

⇒ This Module defines several functions to generate random numbers

⇒ Used to develop games etc.

① random() funct:

This function always generate some float value b/w 0 & 1

② randint() funct:

To generate random integer b/w two given numbers (inclusive)

③ Uniform():

It returns random float values b/w 2 given numbers (not inclusive)

④ randrange([start], stop, [step])

⑤ Choice() function..

-> It wont return random number
-> It will return random object from the given list or tuple