26-09-2025 Friday

Python Data Structure: These are the ways of organizing & sorting data so that they can be accessed & modified efficiently.

- Python provides both built-in data structure and allow us to implement user defined data structure.

Built-in ds : i) List:[] (ii) tuple:() (iii) set:{} (iv) dict : {key: values}

map - It will map/convert the value into integer value

- num=list(map(int, input("Enter values:").split()))
  print(num)

  o/p [1,4,5,6]

- names = input("Enter names:").split()
  print(names)

o/p ['Medium','56','Vijay','63']

Split()-Split the value into multiple values

- list() , []   [] indicates list str
                                                  list () - list f^n.

(i) List() / [] :- It is a heterogenous data collector and its ordered, mutable (changeable) and allow duplicates

- Heterogenous - Allow all types of data type values
  (or diff)
- Ordered - Which hold the position of values
- Mutable - Add, delete, update - Changeable

- n1 = list([5,4,6,7,9])   → # for multiple value inside f^n we need []

```
n1
```

- n2 = [5,4,6,7,9]

```
n2
o/p [5,4,6,7,9]
```

⇒ Heterogenous

```
l1 = ["lekhs", "18-11-2002", 5.2, 7002, True, 7j+2]
l1
o/p  [- ———— (2+7j)]
```

→ Access element using index.

```
l1[0]    o/p "Lekhs"
```

→ All the elements. access

⇒ # Ordered

Using for loop with sequence to access (or get all the values at once

```
for i in l1:            o/p Lekhs
    print(i)                18-11-2002
                            5.2
                            7002
                            True
                            (2+7j)
```

⇒ # Mutable - Add, Delete, Update   (1) ADD * Using Append()

```
l1.append(56)     # add element at the end of list
l1
 ["lekhs', ' ———— (2+7j), 56]
```

→ To add multiple values

```
l1.append([22,55])
l1
o/p [- ———— 56, [22,55]]
```

But ill will add 2 values as a single value

So, we can add multiple value by creating one more list and add both list as a single list.

```
l1 = ["Lekhs", "18-11-2002", 7002, True, 7+2j]
l2 = [555, "John"]
l1 = l1+l2
l1
```

* Using ~~[illegible]~~ index value to update (ii)

```
l1[1] = "8888"        o/p ['Lekhs', '8888', 7002, True, (7+2j)]
l1
```

But it will delete the ~~ad~~ present value in that position and get replaced by new value.

This is update

⇒ Using Insert() to add element at particular position without deleting present value

```
l1.insert(2, "21-01-2006")
l1
o/p ["Lekhs", '8888', '21-01-2006', 7002, True, (7+2j)]
```

⇒ To print all the elements with position

```
for i in enumerate(l1):
    print(i)
o/p    (0, 'Lekhs')
       (1, '5555')
          ⋮
       (5, (7+2j))
```

(iii)
⇒ Delete()

* pop() - Delete the last element

```
l1.pop()
l1
```

* l1.pop(6) - Delete the element @ ~~[illegible]~~ the position of index no. 6
l1

* remove() - Instead of index no, we mention element/value to delete

```
l1.remove("8888")
l1
```

* l1.clear() - To delete all the elements from the list.
* del l1 - To delete the list

⇒ Create a new list square of this given list
num = [5,8, 3,2,6]   o/p [25, 64, 9, 4, 64]

```
num = [5, 8, 3, 2, 6]
sq_num = []                      o/p [25, 64, 9, 4, 64]
for i in num :
    s = i * i
    sq_num.append (s)
 sq_num
```

→ To count the no. of elements
```
len (~~sq~~_num)     o/p 5
```

→ To count the presence of particular element
```
sq_num.count (64)        o/p 1
```

⇒ Create even, odd, prime numbers list from 1 to 20 no.

```
even = []
odd = []
prime = []
for i in range (1, 21):
    if (i % 2==0):
        even.append (i)
    else :
        odd.append (i)
    for j in range (2, i):
        if (i % j ==0):
            break
    else :
        prime.append (i)
print ("Even no.s:", even)
  —   "   —           , odd)
  —   "   —           , prime)
```

o/p

```
Even no.s : [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
Odd no.s  : [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]
Prime no.s : [1, 2, 3, 5, 7, 11, 13, 17, 19]
```