Python datastructure

- A datastructure is a way of sorting and organizing data so that we can go use it efficiently (access and modifying)

Python provides Both datastructures.

1) Built in data structure and
2) allows us to user defined datastructure.

Types:-

1) list = []
2) Tuple = ()
3) set = {}
4) dict = { key:value}

List:-

#one type

Value = input ("enter some values:").split()

#2nd type.

num = list (map (list, input ("enter a Value:").split ()))

easy method:-

ex:-

string float int complex

details = ["Riya", "22-12-1994", 5.7, True, 2387, (5+8j)]

Boolean

To add ⇒ details.append ("data analyst")

o/p = ['Riya', '22-12-1994', 5.7, True, 2387, (5+8j), 'data analyst']

To update we the list position and update keyword.

update false ['Riya', '22-12-1994' 5.7, False, 2387, (5+8j), 'data analyst']

details [3] = False.

To delete one argument means use pop keyword.

details.(pop) ⇒ directly pop out the last attribute.

o/p ⇒ ['Riya', '22-12-1994', 5.7, False, 2387, (5+8j)]

To print position and value in order:-

```
for i in range (details):
    print (i)
```

o/p:- Riya.
22-12-1994
5.7
False
2387
5+8j

To print position and value is order:-

```
For i in enumerate (details):
    print (i)
```

o/p:⇒ (0, "Riya")
(1, "22-12-1994")
(2, "5.7")
⋮
etc.

details [1] = 9345274526.
details

"22-12-1994"

o/p = ['Riya', 9345274456", 5.7, False, 2387, '4+8j', 5.7)

delete specific element means:-

details [3] .pop (2)

o/p:- [Riya, 9345274527, 22-12-1994, 5.7, False, 2387, 4+8j, 5.7]

in the list 22-12-1994 ⇒ sent out

ex: details . pop(2)

Riya ⇒ 0

9345274527 ⇒ 1

2. 22-12-1994 ⇒ # poped out [2] in the list.

To clear all elements means :-

```
    details . clear ()
    detail
o/p=> []
```

To delete details.

```
    del details
    details.
```

1. Create square num for the given list number,

```
    L1 = [5, 2, 9, 7]
ex. o/p = [25, 4, 81, 49].
Prgrm:-
    L1 = [5, 2, 9, 7] .
    L2 = []
    for i in range L1 :
        s = i * i
        L2. append (s)

L2. => o/p => [25, 4, 81, 49]
```

2) Create odd, even, prime num from the list 1 to 20.

```
    oc = 0
    ec = 0
    pN = 0
    for i in range (1, 21, 1):
        if (i % 2 == 0):
            ec. append (i)
        else:
            oc. append (i)
    for j in range (2, i, 1) :
        if (i % j == 0):
        break.
```

```
    else :
        pN. append (i)
    print (oc, ec, pN)

o/p =>
          oc                                 ec
    [1, 3, 5, 7, 9, 11, 13, 15, 17, 19] [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
                  [1, 2, 3, 5, 7, 11, 13, 17, 19] => pN => prime Number.
```