

DAY-17

September-26

DATA STRUCTURES:

- Python data structure are the ways of organising and sorting data so that they can be accessed and modified efficiently.
- Python provides both built-in data structure and allow us to implement user defined data structures.

Built-in data structures:

- list []
- tuple ()
- Set {}
- Dictionary dict {key:values}

To read multiple values with one variable we use the below format

- To read multiple integers:

```
num = list(map(int,input("Enter the values:").split()))
```

output:

Enter the values: 10 20 30 40

Num

[10, 20, 30, 40]

- To read multiple strings:

```
names = input("Enter your names:").split()
```

Any input given to this will be considered as strings and by default returns in list format.

Output:

Enter your names: Indu 30 priya

Names

```
['Indu', '30', 'priya']
```

Here 30 is number but it's considered as string.

1. **List:**

List is the heterogeneous data collector and it is ordered, mutable and allows duplicates.

- It can be written in two formats:

list() – as a function

[] – symbol

- Mutable-changes can be done
- Ordered-something that has position

Syntax:

```
n1 = list([2,6,4,7,1])
```

```
n1
```

```
output: [2, 6, 4, 7, 1]
```

```
n2 =[5,9,0,3,2]
```

```
n2
```

```
output: [5,9,0,3,2]
```

Heterogenous:

```
l1 = ["Indu","20-11-2002",5.3,1234,True,10j+1]
```

```
l1
```

```
output:
```

```
['Indu', '20-11-2002', 5.3, 1234, True, (1+10j)]
```

Ordered:

```
l1[0]
```

```
output:
```

```
'Indu'
```

```
For i in range(11):
```

```
Print(i)
```

Output:

Indu

20-11-2002

5.3

1234

True

(1+10j)

Mutable:

- Data manipulation can be done like adding, deleting and updating.

Adding:

```
l1.append(24)
```

```
l1
```

```
output: ['Indu', '20-11-2002', 5.3, 1234, True, (1+10j), 24]
```

- To add multiple values at once individually:

Create one more list and add the desired values then finally merge the lists.

```
l2 = [24,5.3,'Indu']
```

```
l2
```

```
[24, 5.3, 'Indu']
```

```
l1+l2
```

output:

```
['Indu', '20-11-2002', 5.3, 1234, True, (1+10j), 24, 24, 24, 5.3, 'Indu']
```

The update is temporary if we want to make permanent changes we need to assign it to any list.

```
l1 = l1+l2
```

```
l1
```

output:

```
['Indu', '20-11-2002', 5.3, 1234, True, (1+10j), 24, 24, 24, 5.3, 'Indu']
```

Duplicated Values:

- Allows repeated elements.

```
['Indu', '20-11-2002', 5.3, 1234, True, (1+10j), 24, 24, 24, 5.3, 'Indu']
```

- To update values in particular position

```
l1[1] = "9398072338"
```

```
l1
```

output:

```
['Indu', '9398072338', 5.3, 1234, True, (1+10j), 24, 24, 24, 5.3, 'Indu']
```

- To add values in a particular position.

```
l1.insert(2,"20-11-2002")
```

```
l1
```

output:

```
['Indu',  
'9398072338',  
'20-11-2002',  
5.3,  
1234,  
True,  
(1+10j),  
24,  
24,  
24,  
5.3,  
'Indu']
```

```
for i in enumerate(l1):
```

```
    print(i)
```

output:

```
(0, 'Indu')
```

```
(1, '9398072338')
```

```
(2, '20-11-2002')
```

```
(3, 5.3)
```

```
(4, 1234)
```

```
(5, True)
```

```
(6, (1+10j))
```

```
(7, 24)
```

```
(8, 24)
```

```
(9, 5.3)
```

```
(10, 'Indu')
```

- To delete a value by index:

```
l1.pop(3)
```

- To delete a value:

```
l1.remove(24)
```

```
l1
```

output:

```
['Indu', '9398072338', '20-11-2002', 1234, True, (1+10j), 24, 5.3, 'Indu']
```

- To remove all the elements at once, the list will be existing but the contents will get deleted.

```
l1.clear()
```

```
l1
```

output: []

- To delete the list itself permanently.

```
del l2
```

Programs:

```
1. num1 = [2,5,8,1,4]
```

Create a new list containing the square of the above list.

expected output:

```
[4,25,64,1,16]
```

Code:

```
num1 = [2,5,8,1,4]
```

```
ns = []
```

```
for i in num1:
```

```
    sr = i*i
```

```
    ns.append(sr)
```

```
ns
```

output:

```
[4, 25, 64, 1, 16]
```

- To find the number of elements of list we use length function

```
Len(ns)
```

```
5
```

- To count how many times the element is there go with count function

```
ns.count(64)
```

```
1
```

2. create odd,even,prime number list from 1 to 20 number.

Code:

```
even = []
```

```
odd = []
```

```
prime = []
```

```
num = 20
```

```
for i in range(1,num+1,1):
```

```
    if(i%2==0):
```

```
        even.append(i)
```

```
    else:
```

```
        odd.append(i)
```

```
    for j in range(2,i,1):
```

```
        if(i%j==0):
```

```
            break
```

```
    else:
```

```
        prime.append(i)
```

```
print(even,odd,prime)
```

output:

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20] [1, 3, 5, 7, 9, 11, 13, 15, 17, 19] [1, 2, 3, 5, 7, 11, 13, 17, 19]