06/00/25

Function as parameter :

→ A function can be passed as a parameter (argument) to another function.

→ This is possible because functions are first-class objects in python - meaning they can be :

Assigned to variables :

1. Stored in data structures (list, dict, etc..,)
2. Returned from other functions.
3. passed as arguments to other functions.

Nested functions :

Nested function means a function defined another function. This is called an inner function.

Recursion function :

A recursive function is a function that calls itself until a base condition met.

Lambda function:

→ A lambda function is a small, anonymous (nameless) function in python.

1. Defined using a keyword lambda instead of def.
2. It can take any number of arguments but must contain only one expression.
3. Expression is automatically returned (no need to use return).

Syntax:

lambda arguments : expression.

Ex:

1) `s = lambda num: num * num`

`s(25)`

o/p:

→ 25

2) `add = lambda a, b : a+b`

`add (4, 8)`

o/p:

= 12.

Data structure:

→ python data structure are ways of organizing and sorting data so that they can be accessed and modified efficiently.

→ python provides both built-in data structure and allows us to implement user defined data structure

Main Data structures in python:

1. List [ ]
2. Tuple ( )
3. Set { }
4. Dict = { key : values }

07/10/25

List:

1. It is a multivalued variable and it is hetero-geneous (pass anything) in nature.

Heterogeneous:

A Heterogeneous Collection is one that can hold different types of data in a single container.

Ex:

```
l1 = [34, "vasu", 43.9, "True"]
print(l1)
```

o/p:

```
[34, 'vasu', 43.9, True]
```

2. It is mutable, ordered and allow duplicates.
3. It is represented by list( ) or [ ]

Mutable :

Mutable means the elements of a list can be changed after creation – you can add, remove, or modify items.

Ordered :

Ordered means the elements in a list have a specific sequence, and that order is preserved.

→ Each item has an index position (starting from 0).

→ Even if you modify the list, the order of element remains predictable unless you explicitly change it

E.g: by sorting or reversing.

Example :

```
L1 = [34, "vasu", 27.9, True, 34, 1+8j, "vasu"]
print(L1)
```

O/p :

```
[34, "vasu", 27.9, True, 34, (1+8j), 'vasu']
```

~~Methods in list~~ :

List Methods in python :

1. append( )
2. insert( )
3. Extend( )
4. remove( )
5. pop( )
6. clear( )

7. index( ) → Returns index of x.

8. Count( ) — Counts occurences of x

9. Sort( ) → Sort list

10. reverse( ) → reverse order

11. Copy( ) → Returns a shallow copy