What is function recursion?
Calling function itself is called recursive function call or function recursion. Recursive functions are functions that calls itself. It is always made up of 2 portions, the base case and the recursive case. The base case is the condition to stop the recursion. The recursive case is the part where the function calls on itself.

Applications of function recursion
Building Algo

- Sorting
- Searching
- Dynamic Programming
- Traversing
- Data Structures

Recursive function is developed using one concept called divide and conq.

Increasing the recursion limit: Python has a default maximum recursion depth of **1000**. If a function exceeds this limit, it can be increased using the **sys. setrecursionlimit(n)** function. Developers should be careful when increasing the limit as this can cause a crash if the recursion is not properly controlled.

Example:
```
def fun1():
   print("Inside Fun1")
   fun1() # recursive function call



fun1()
```

Output:
```
Inside Fun1
Inside Fun1
Inside Fun1
Inside Fun1
Inside Fun1
Inside Fun1
Traceback (most recent call last):
  File "C:\Users\nit\PycharmProjects\pythonProject1\funtest50.py", line
4, in fun1
    fun1() # recursive function call
    ^^^^^^
  File "C:\Users\nit\PycharmProjects\pythonProject1\funtest50.py", line
4, in fun1
    fun1() # recursive function call
    ^^^^^^
  File "C:\Users\nit\PycharmProjects\pythonProject1\funtest50.py", line
4, in fun1
    fun1() # recursive function call
    ^^^^^^
  [Previous line repeated 996 more times]
  File "C:\Users\nit\PycharmProjects\pythonProject1\funtest50.py", line
3, in fun1
    print("Inside Fun1")
RecursionError: maximum recursion depth exceeded while calling a
Python object
```

Recursion call required 2 statements

1. Base case
2. Recursive case

Base case is condition, which defined how many times recursion has to be done
Recursive case is called function itself.

Function recursion is evaluated using data structure called stack (LIFO).

Example:

```python
def print_num(n):
    if n>1: # base case
        print_num(n-1) # recursive case
    print(n)

print_num(3)
```

Output:

```
1
```

```
2
3
```

Example

```
def factorial(num):
    if num==0:
        return 1
    else:
        return num*factorial(num-1)



number=int(input("Enter any number"))
result=factorial(number)
print(f'Factorial of {number} is {result}')
```

Output:

```
Enter any number6
Factorial of 6 is 720
```

Example:

```
count=0
def count_digits(num):
    global  count
    if num>0: # base case
        count=count+1
        count_digits(num//10) # recursive case


number=int(input("Enter any number "))
count_digits(number)
print(f'Count of digits or length of number {count}')
```

Output:

```
Enter any number 45678
Count of digits or length of number 5
```

Lambda Functions or lambda Expressions

What is lambda function or lambda expression?
Lambda function is anonymous function.
A function which does not have any name is called anonymous function.
Lambda function is single line function.

Applications of lambda functions

1. Lambda functions are used when working higher order functions
2. Callback functions
3. Function as an argument

What is higher order function?
A function which receives input as another function to perform operation is called higher order function or callback function.

Syntax:
lambda [parameters]:expression/statement

lambda function is defined with parameters or without parameters.