Recursion Fun: It is a fun that calls itself until a base condition is satisfied.

Ex: Find factorial no.

```
def fact(n):
    if(n == 1):
        return 1
    else:
        return n * fact(n-1)
fact(5)
```

fact(5)
if 5 == 1 fails
else
5 * fact(5-1) (4)
5 *
fact(4)
4 == 1 X
4 * fact(4-1) (3)
5 * 4 * 3 * 2 = 120

3 == 1 X
2 == 1 X

Nested function:

* Lambda fun: It is a small, anonymous fun in python.
* Defined using a 'lambda' keyword instead of 'def'.
* It can take any no. of arguments but must contain only one expression.
* expression is automatically returned (no need to use return)

1) Create a fun to print prime no. from the given range using with input & without return method.

```
def prime(a, b):
    print(f" prime no. b/w {a} & {b} are:", end=" ")
    for num in range(a, b+1, 1):   (2,9,1)
        if (num > 1):
            for i in range(2, num):
                if num % i == 0:
                    break
            else:
                print(num, end=" ")
prime(2,9).
```

num range (2, 9, 1)

2 > 1 True

i range (2, 2) ⇒ There is no range b/w (2, 2) so it will print 2 itself. check with 2 only. o/p 2

2 % 2 == 0 print & break.

For num = 3

3 > 1 True.

for i range will be (2, 3) Then it will go to else block & print 'num' value.

3 % 2 == 0 × (print 3 only). o/p 3

For num = 4

for i range (2, 4) i = 2, 3, 4.

4 % 2 == 0 ✓

For num = 5

for i range (2, 5): i = 2, 3, 4.

5 % 2 == 0 False then check with next 'i' value.

5 % 3 == 0 False.

5 % 4 == 0 False

No divisor is found then go to else block & print 'num' value. o/p 5

For num = 6

for i range (2, 6): i = 2, 3, 4, 5.

6 % 2 == 0 True (break num' will increment).

For num = 7

for i range (2, 7) i = 2, 3, 4, 5, 6

7 % 2 == 0 ×

7 % 3 == 0 ×

7 % 4 == 0 ×

7 % 5 == 0 ×

7 % 6 == 0 ×

No divisor is found so go to else block & print 'num' value.

o/p 7

Then check with with 8 & 9 final o/p will be (2, 3, 5, 7).

Nested lambda:

```
add1 = lambda x : (lambda y : x+y)
add1(5)(3).
```

o/p : 8

outer lambda

add(5) x = 5 & returns (lambda y = 5+y)

inner lambda

y = 5+y take y = 3 return 5+3 = 8.