Recursion:

A function is said to be recursively defined, if a function containing either a call statement to itself or a call statement to a second function that may eventually result in a call back to original statement.

Properties of Recursion

1.) There must be certain criterion called base criteria for which function must stop

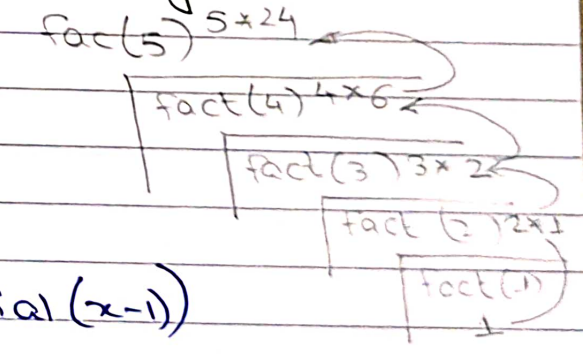
2.) Each time the function call itself (directly or indirectly), the argument of the function must be closer to a base value

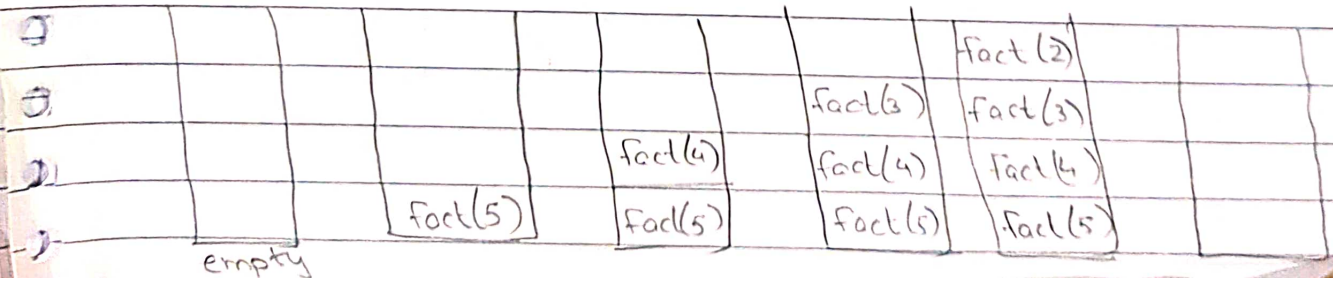
Q. Comparision between iteration & recursion

Q. Write pseudocode for printing array in reverse order using recursion

Q. Write recursive code for finding factorial

Sol.
```
def factorial(x):
    if x == 1
        return 1
    else:
        return (x * factorial(x-1))
```

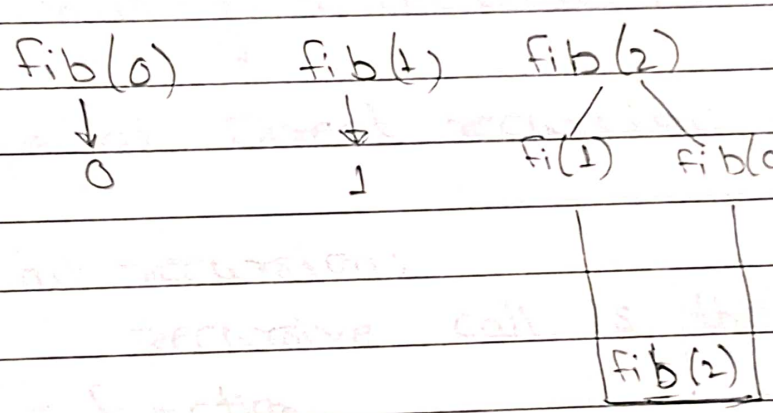


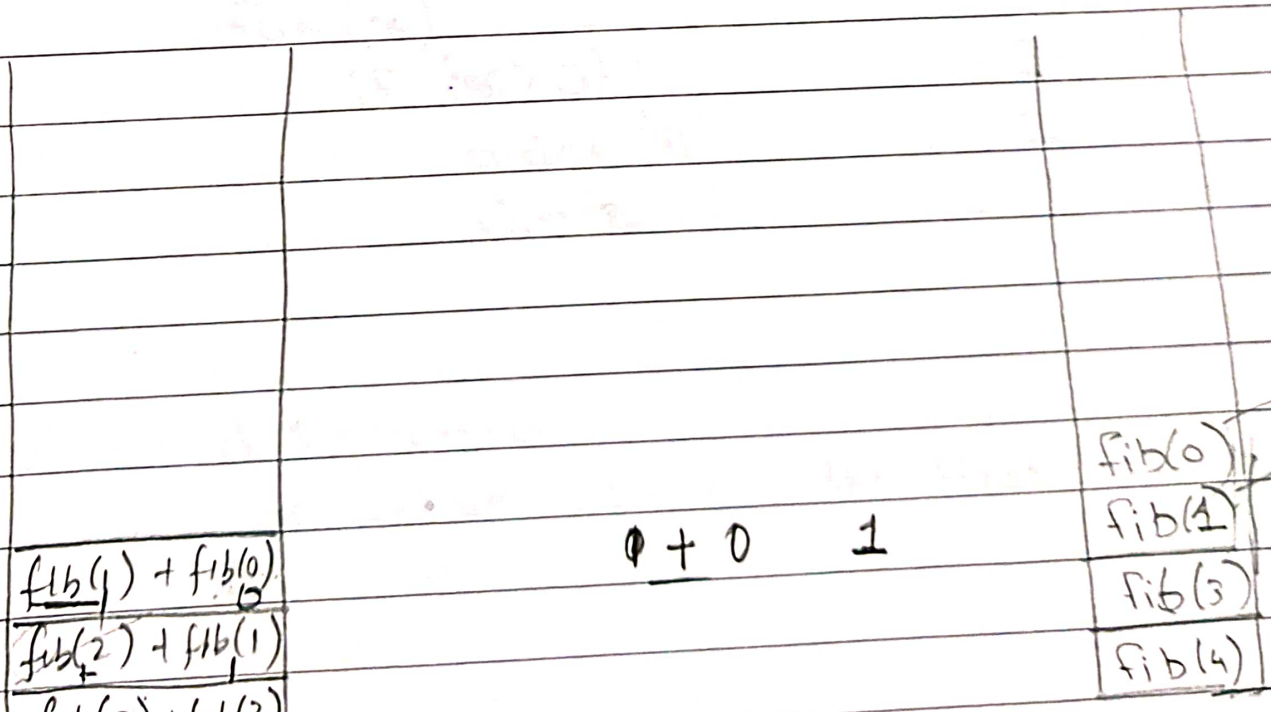


Q. Program to print fibonacci series using recursion

```
sol  fib(n)
        if n <= 0
            return n
        else
            return (fib(n-1) + fib(n-2))

n_terms = 10
for i in range 0 to n_terms:
    print fib(i)
```





Date : / /
Page No.

Types of Recursion:

1.) Direct Recursion:

In this, functions calls itself. This process involves a single step recursive call by the function from inside itself.

2.) In Direct Recursion:

In this function calls other function which in turn calls back to original function. This process consists of two steps when creating a recursive call.

Types of Direct recursion:

1.) Tail recursion:

recursive call is the last statement in the function.

```
fun(x)
    if (x > 0):
        print(x)
        fun(x-1)
```

fun(3)

3 . fun(2)
2 fun(1)
1 fun(0)

2.) Head Recursion:

recursive call is the first statement of the function

```
fun(x)
    if (x > 0):
        fun(x-1)
        print(x)
```

3:) Tree recursion:

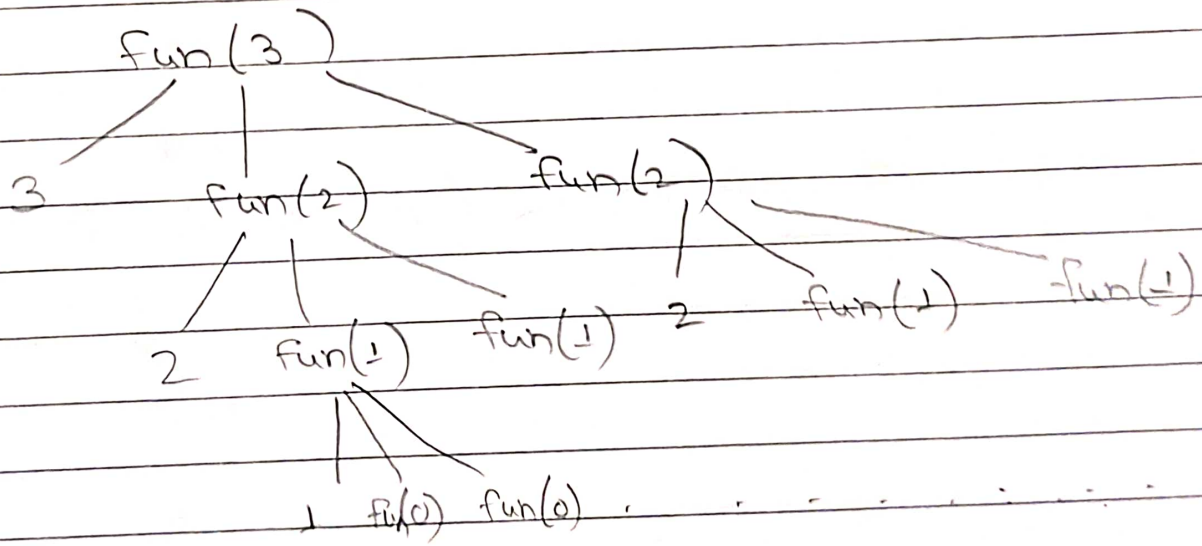
If a function calling itself for one time then it's known as Linear recursion. If a function calling itself more than one time it's called Tree recursion

```
def fun(n):
    if n>0:
        print n

        fun(n-1)

        f(n-1)
        fun(n-1)
```

get output for fun(3)



4) Nested Recursion:

In this recursion, a recursive function will pass the parameter as a recursive call. That means recursion inside recursion.

```
def fun(n):
    if (n>100)
        return n-10
    return fun(fun(n+11))
```

get output for : fun(95)

fun(95)

↓

fun(fun(95+11)) ⟶ 96 = fun(106)

↓

fun(96)

↓

fun(fun(96+11)) ⟶ 97 = fun(107)

↓

fun(97)

↓

fun(fun(97+11)) ⟶ 98 = fun(108)