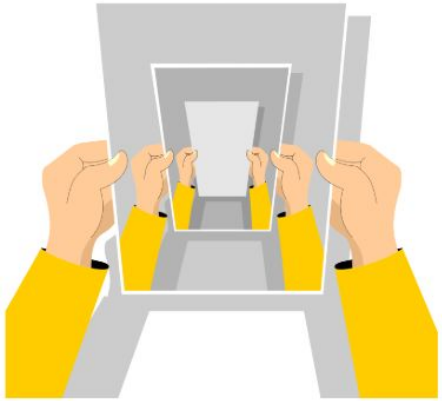
# RECURSION

Intro
Recursive Functions
Recursion vs Iteration

# Recursion



A function can call other functions.

But a function calling itself seems logically wrong.

Recursion refers to a programming technique in which a function calls itself directly or indirectly.

# Direct/Indirect Recursions

```
def A( ):
  A( )
```

Direct
Calling Itself

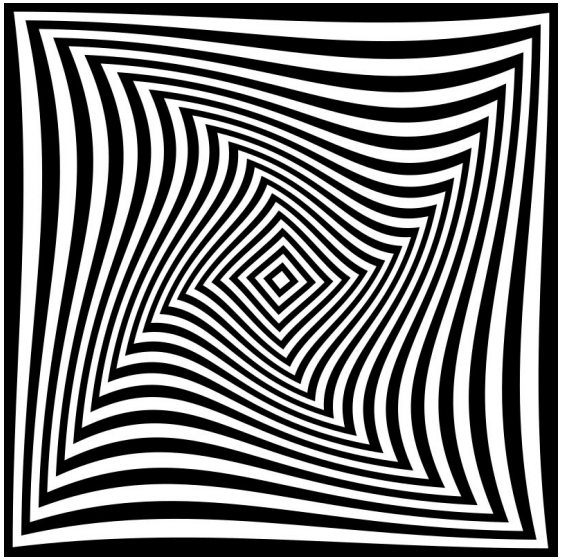
```
def A( ):
  B( )

def B( ):
  A( )
```

Indirect
Calling Itself

# Recursive Function

```
def func1( ) :
    print( ‘Hello World’ )
    func1( )
```



This is a recursive function calling itself.

# Recursion is Infinite...

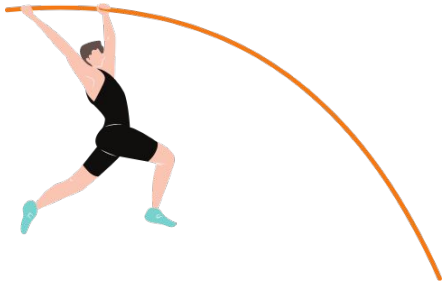
To make it usable we need to make it finite.
We will make a case which will break the recursion.

```
def a(i):
  if(i>0):
      print(i)
      i=i-1
      a(i)
  else:
      return
```

Recursive Condition

Exit Condition

# Base Case



We make a case whose result is already known. We use this case as the exit case.

e.g. recursive code for factorial.

factorial(1) = 1

We know it already.

BASE CASE

# Recursive Definition

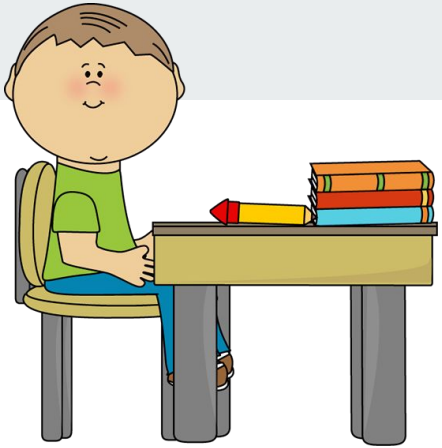
Function to compute : $a^n$

Iterative definition : $a^n = \underbrace{a * a * a \ldots a}_{n}$

Recursive definition : $a^n = a * a^{n-1}$

Further $a^{n-1} = a * a^{n-2}$

# Practice Time



Q1. Find the output:

```
def binod(n):
    if n==0:
        print('Finally')
    else:
        print(n)
        binod(n-3)
    binod(15)
```

Q2. Find the output:

```
def val(n):
    if(n<=1):
        return True
    elif(n%2==0):
        return val(n/2)
    else:
        return val(n/1)
```

# Recursion vs Iteration

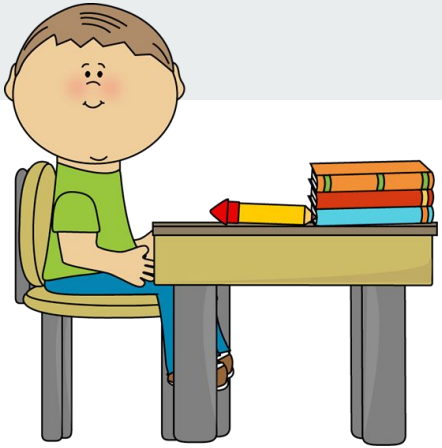
Recursion & Loops are related, one thing can be done by both methods.
Sometimes Recursion is better, sometimes loop is better.

When a loop runs, it repeats the same variables and the same unit of code.

In Recursion for each recursive call, new memory is allocated for the function.

# Important Points

- Recursion makes a program look **shorter** and easily **understandable** in terms of mathematics also.
- Recursion is **heavy on memory** due to allocation of new memory with each recursive call.
- When the code length & complexity matters, *Recursion* is recommended.
- When the time & efficiency of the code matters, *Iteration* is recommended.



# Practice Time

Q1. Write a program to print the string backwards (by both methods).

Do the following questions recursively.

Q2. Write a program to calculate $a^b$.

Q3. Write a program to calculate HCF.

Q4. Write a program to print fibonacci series.

# THANK YOU FOR WATCHING!

Milte hain next video me, BYEE!!!!