→ How to write recursive solution

→ Working

→ TC / SC of recursive code. ← Next class

Why ?

- → Mergesort / quick sort
- → Binary Tree / BST / BBST
- → Segment trees / Tries
- → Dynamic Programming
- → Backtracking
- → Graphs

Recursion : Function calling itself.

→ Solving a problem using smaller version of same problem

subproblem

$$\text{sum}(N) = \underbrace{1 + 2 + 3 + \cdots \cdots + N-1}_{\text{sum}(N-1)} + N$$

$$\text{Sum}(N) = \underbrace{\text{Sum}(N-1)}_{\text{Subproblem}} + N$$

Recursion code

1. Trust : Decide what your fun will do, trust your function
2. main logic : recursive eq^n / main logic
3. Base condition : when your code will stop

```
int sum(int N)
{
    if(N==1) return 1
    return sum(N-1) + N
```

$$fact(N) = \underbrace{1 * 2 * 3 * 4 * \text{-- ---} * (N-1)}_{fact(N-1)} * N$$

$$fact(N) = fact(N-1) * N$$

will return factorial of N

```
int fact (int N)
{
    if(N==0) return 1

    return fact(N-1) * N
}
```

Q. Cal N^{th} Fibonacci number

0th	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th
0	1	1	2	3	5	8	13	21	34	------

$$Fib(N) = Fib(N-1) + Fib(N-2)$$

Nth Fibonaci number

```
int  Fib( int N )
{
    If( N ==0  ||  N==1 )  return  N

    return  Fib( N-1 ) + Fib(N-2)
}
```

✓ Fib(0) = ✗ Fib(-1) + ✗ Fib(-2)

✓ Fib(1) = Fib(0) + ✗ Fib(-1)

Fib(2) = Fib(1) + Fib(0)

Fib(3) = Fib(2) + Fib(1)

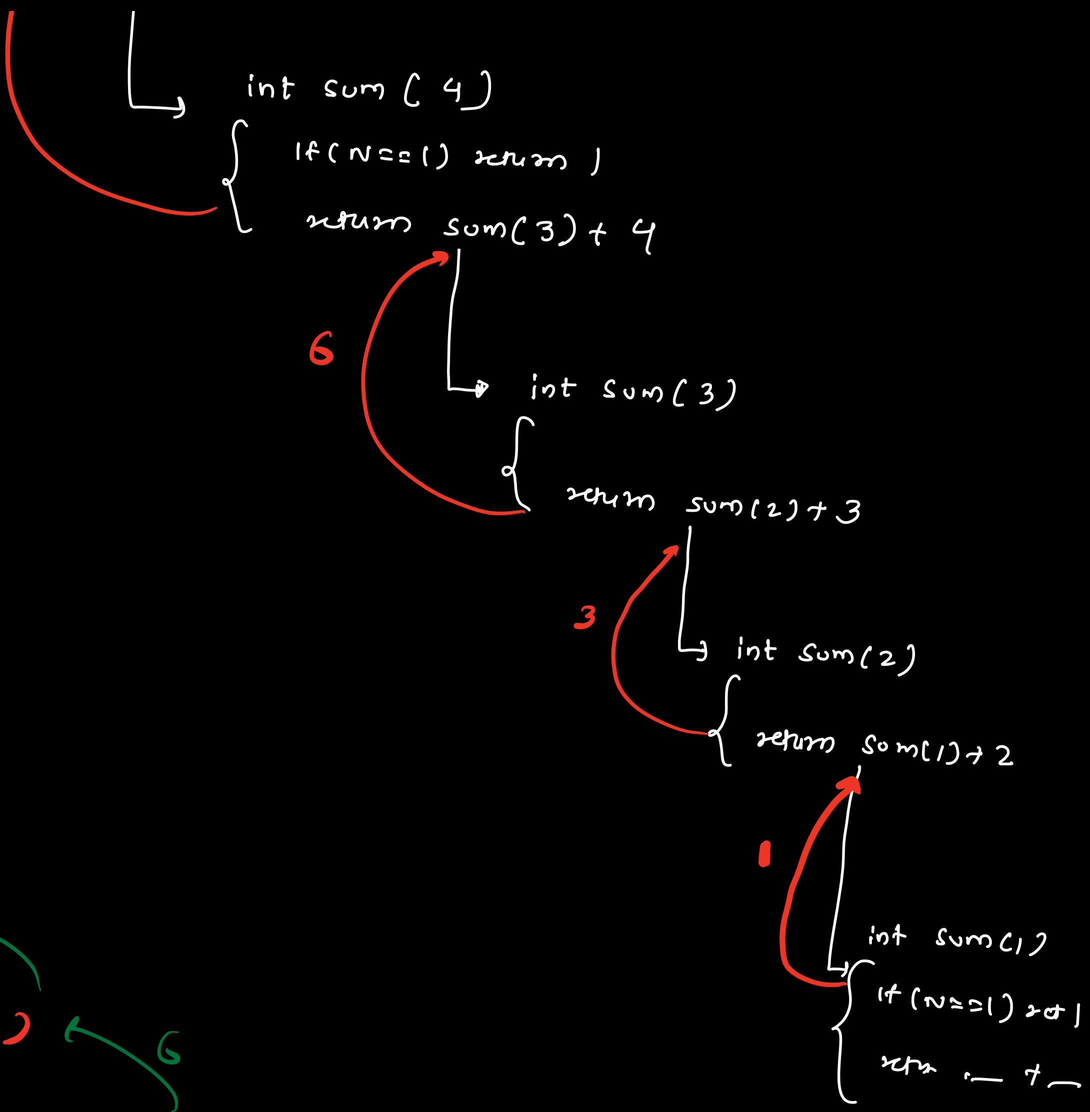
Working of Recursion

```
int  Sum(int N)
{  If(N==1) return 1
   return  Sum(N-1)+N
```

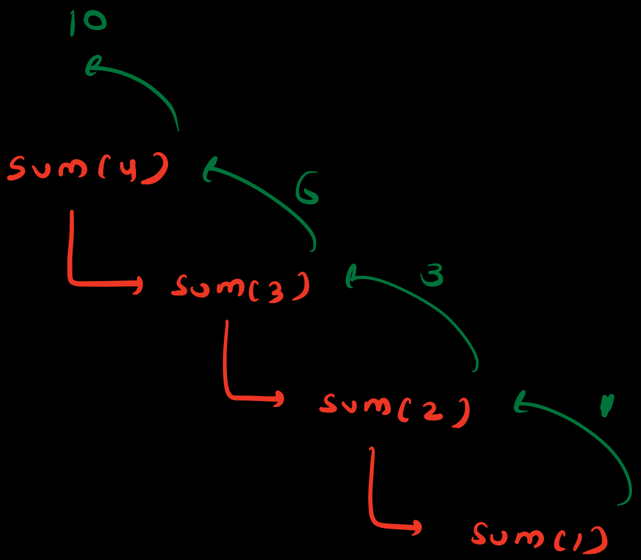
sum(4)

```
main()
{
    aN = sum(4)
}
```

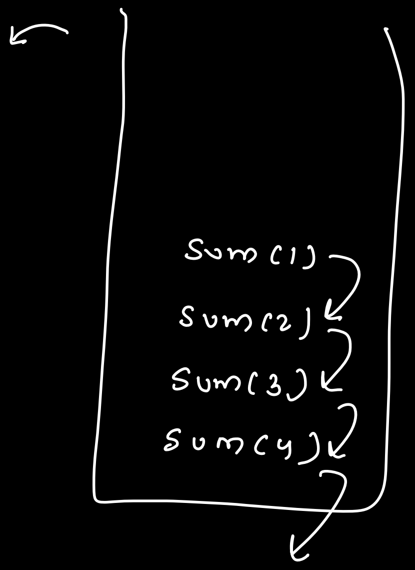
10



```
int sum ( 4)
{
  If(N==1) return 1
  return sum(3)+ 4
}
int sum(3)
{
  return sum(2)+3
}
int sum(2)
{
  return sum(1)+2
}
int sum(1)
{
  if (N==1) ret 1
  retn __ + __
}
```



Function
stack
or
call
stack



Last In First out

LIFO
↓
Stack

First In Last out

FILO
↓
Stack

Stack overflow exception

Q. Given N number. Print all the number from 1 to N in ascending order using recursion

N = 5

1
2
3
4
5

N = 4

1
2
3
4

print(N) =

1
2
3
4
5
.
.
.
N-1

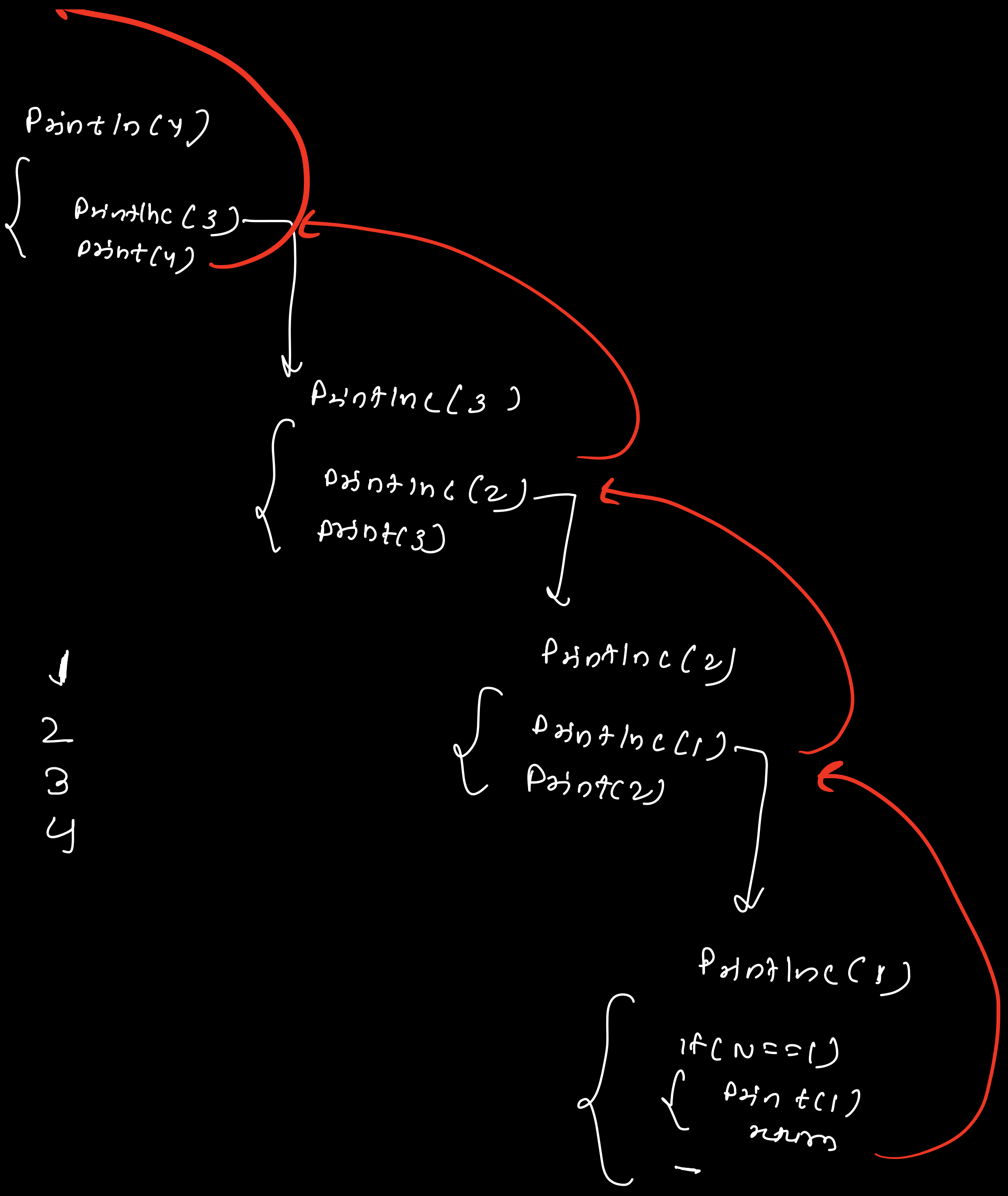
→ Print(N-1)

N

Printf(N) = Printf(N-1)
N

→ Print all nos from 1 to N in f ing order

```
void PrintInc (int N)
{
        if (N==1)
        {   print(1)
            return          ]→ Base condition
        PrintInc (N-1)
        Print(N)            ]→ main logic
        return
}
```



```
Println(4)
{
    Printlnc(3)
    print(4)
```

```
Printlnc(3)
{
    Printlnc(2)
    Print(3)
```

```
Printlnc(2)
{
    Printlnc(1)
    Print(2)
```

```
Printlnc(1)
{
    if(N==1)
    {   print(1)
        return
```

1
2
3
4

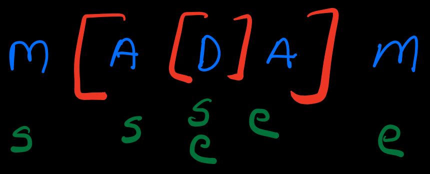
Q. Given a string. find if Palindrome or not

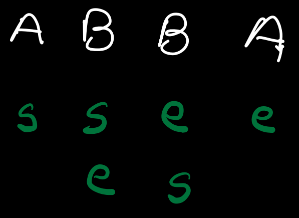
Recursive code.

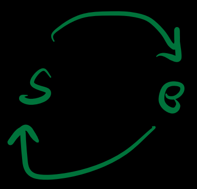
MADAM → True

NITIN → True

SCALER → False







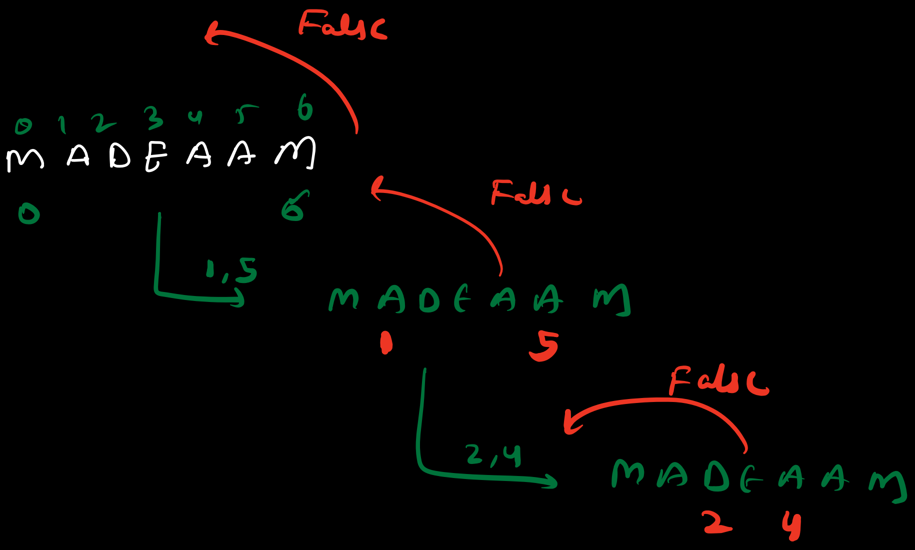
return is str is Palindrome
from s to e

```
boolean isPalindrome ( string str, int s, int e)
{
        if ( s ≥ e)  return True                    ]  Base
                                                       condition

        if ( str[s] != str[e])  return False        ]  main
                                                       logic
        return   IsPalindrom ( str, s+1, e-1)
}
```

Doubts

```
main (   )
{
        IsPalindrom ( "MADAM", 0, N-1 )
                                  4
}
```



```
if(N==1)
{
        Print(1)
        return
```