Today's Content :-

→ Recursion ?

→ How to write a recursive code / Tracing

T.C → next Class

Why Recursion ?

Merge Sort / Quick Sort

B.T / B.S.T / BBST / Segment Tree / Tries

Dynamic Programming

Backtracking

Graphs

Today's Quote

Be tolerant with others & strict with yourself.

Recursion :- function calling itself → Recursion

↳ solving problems, using smaller instance of same problem.

↓

Subproblem.

$Sum(N) = 1 + 2 + 3 + \cdots N$

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

↳ Subproblem.

How to write recursive codes?

Assumption :- Decide what your fcn does, and assume its working from smaller problem.

Main logic :- Solve current problem with assumption.

Base Condn :- Input for which we need to stop.

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

↳ sum of first N-1 natural numbers.

fact(3) = 3*2*1 => 6 , fact(4) = 4*3*2*1 => 24

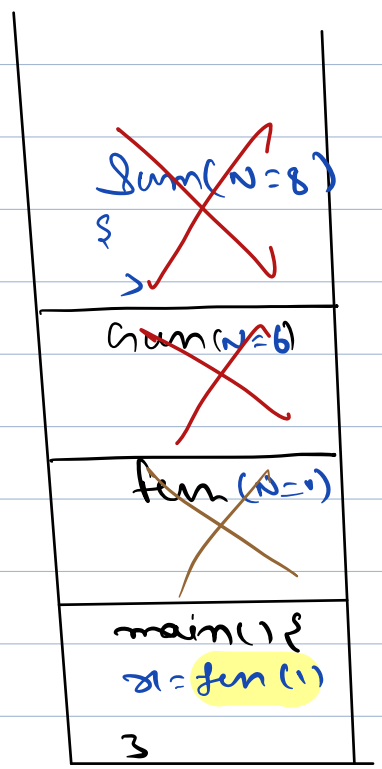
```
int fact(N) {
    if (N==1){ return 1 }
    return fact(N-1) * N
}
```

function call Tracing

```
int fun(N) {
    return gun(N+5)
}

int gun(N) {
    return sum(N+2)
}
```

64



```
int fun (N) {
    return N * N
}

main( ) {
    int x = fun (1);
    Print (x)
}
```

→ 6p

Ex 2 :-

```
int fun( N) {
    ret gun (N-2) + 3
}

int gun (N) {
    return fun(N+3)
}

int fun (N) {
    ret 5+N
}
```

5

1 + 3 | 4

3

3

3

6

3

~~fun(6)~~
~~gun(3)~~
~~fun(5)~~

sum(4) => 10.

```
int sum(N=4) {
    if(N==1) { return 1 }
    return sum(N-1) + N
```

6 4

3

```
int sum(N=3) {
    if(N==1) { return 1 }
    return sum(N-1) + N
```

3 3

3

```
int sum(N=2) {
    if(N==1) { return 1 }
    return sum(N-1) + N
```

1 2

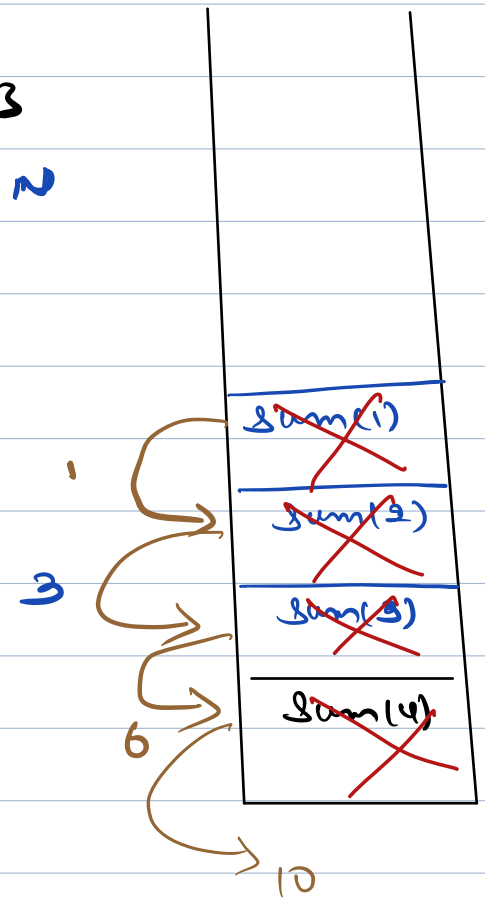
3

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

3

```
int sum(N) {
  ① if(N==1) { return 1 }
  ② return sum(N-1) + N
```

3



Note :- In recursion, if your code gives, memory Limit Exceeded (Stack Overflow) that means code is not properly stopped.

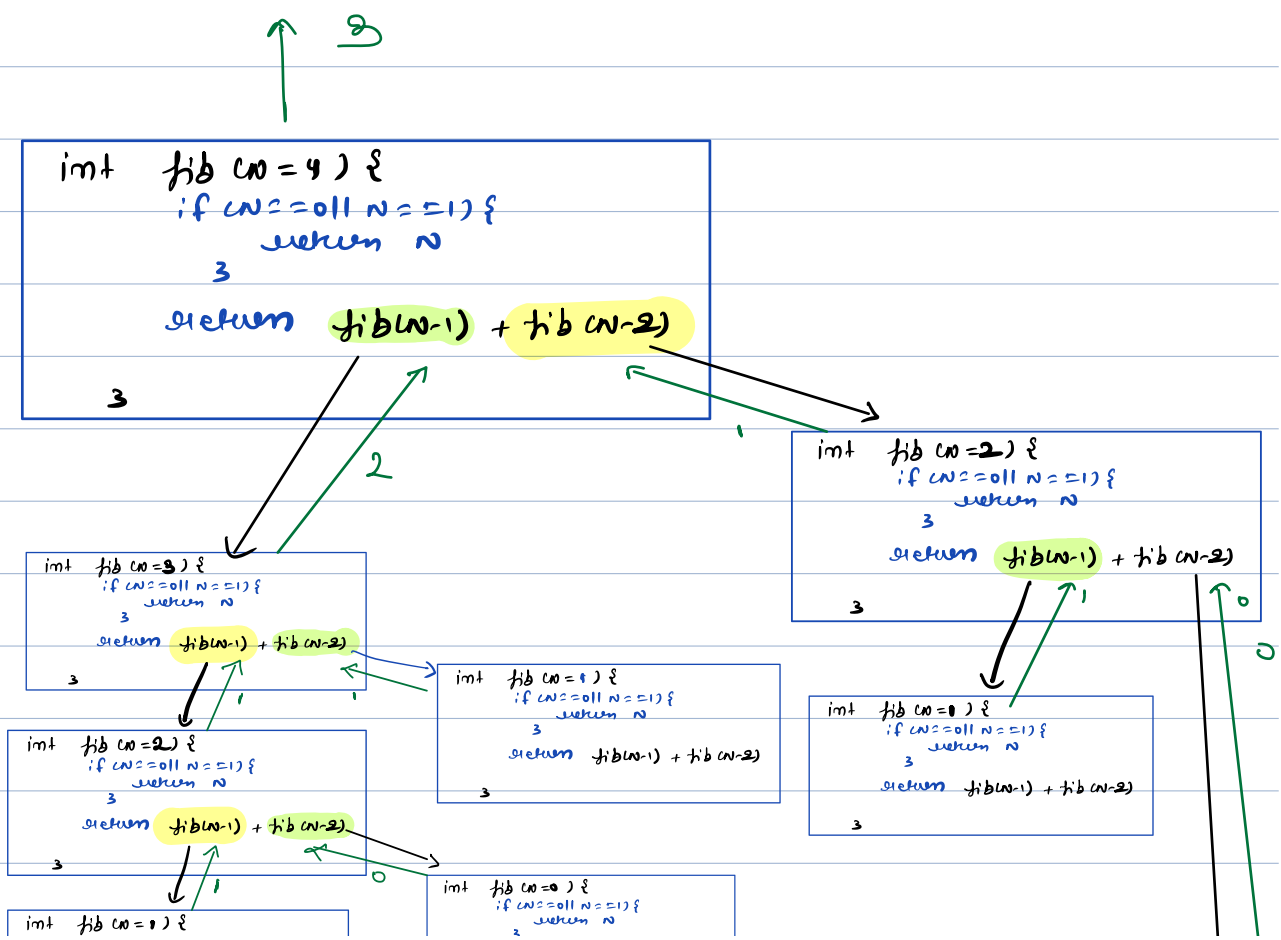
<verify Base Condns>

Wrong Base Case

TLE ✗ Stack Overflow ✓

fibonacci →	0	1	1	2	3	5	8	13	21	....
Input →	0	1	2	3	4	5	6	7		

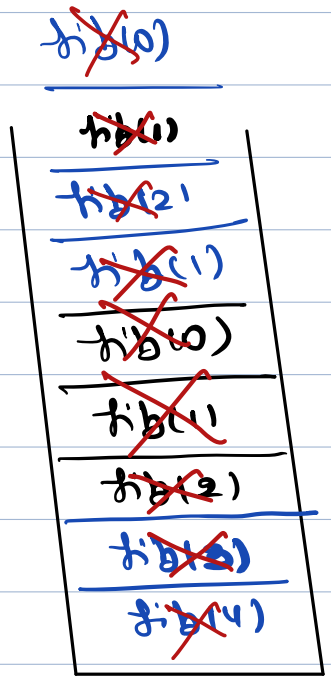
```
int fib (N) {
    if (N==0 || N==1) {
        return N
    }
    return fib(N-1) + fib(N-2)
}
```



```
if (N==0|| N==1){
    return N
}
return fib(N-1) + fib(N-2)
}
```

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

```
int fib (N=0) {
    if (N==0|| N==1){
        return N
    }
    return fib(N-1) + fib(N-2)
}
```



Please do it yourself.

If you can't make the tree diagram or show it in a stack → you don't know recursion.

Break 10:15 - 10:25 pm :-

Q) Given N, print all numbers from 1-N, in increasing order.

```
void printIncreasing (N=4) {
    if (N==1) { print(1) }
    printIncreasing (N-1);
    print(N);
}
```

```
void printIncreasing (N=3) {
    if (N==1) { print(1) }
    printIncreasing (N-1);
    print(N);
}
```

```
void printIncreasing (N=2) {
    if (N==1) { print(1) }
    printIncreasing (N-1);
    print(N);
}
```

```
void printIncreasing (N=1) {
    if (N==1) { print(1) }
    printIncreasing (N-1);
    print(N);
}
```

```
void PrintIncreasing (N=0) {
    if (N==1) { Print(1) }
    PrintIncreasing (N-1);
    Print(N);
}
```

Becoz of missing return statement, above code will result in stack Overflow.

```
void PrintIncreasing (N=4) {
    if (N==1) { Print(1)
        return }
    PrintIncreasing (N-1);
    Print(N);
}
```

```
void PrintIncreasing (N=3) {
    if (N==1) { Print(1)
        return }
    PrintIncreasing (N-1);
    Print(N);
}
```

```
void PrintIncreasing (N=2) {
    if (N==1) { Print(1)
        return }
    PrintIncreasing (N-1);
    Print(N);
}
```

```
void PrintIncreasing (N=1) {
```

1
2
3
4

void printIncreasing (N) {
if (N==1) { Print(1)
return }
PrintIncreasing (N-1);
Print(N);
}

Ques) Given a string, check if its palindrome or not?

Ex1:-

dad → yes
nitin → yes
malayalam → yes
vaibhav → No

```
bool checkPalindrome(string s){
    if(s.length==0||1) { return True }
    char first = s[0];
    char last = s[s.length-1];

    if (first != last){
        return false
    }

    ret checkPalindrom(s.substr(1,n-2))
}
```

malayalam

it's creating a new string every time

cp(malayalam)

↓

cp (alayala)

↓

cp (layal)

```
bool    checkpalindrome(string str, int s, int e){
        if (s>e) {return True}

        if (str[s] != str[e]) {
            return false;
        }
return checkpalindrome(str, s+1, e-1);
}
```

cp (str, 0, 8)

↓

str = malayalam
0 1 2 3 4 5 6 7 8

cp (str, 1, 7)
↓
cp (str, 2, 6)
↓
cp(str, 3, 5)
↓
cp (str, 4, 4)
↓
cp(str, 5, 3)

nim
0 1 2 3

cp (str, 0, 3)
↓
cp (str, 1, 2)
↓
cp (str, 2, 1)

vaibhav
0 1 2 3 4 5 6 → cp(str, 0, 6)
↓
cp(str, 1, 5)
↓
cp(str, 2, 4)

↑ false.
↑ false