optional

Sunday : Problem solving session

Recursion ?

How to write a recursive code?

TC / SC ? [Wednesday]

Merge sort / Quicksort / Heap sort

Trees

Dynamic Programming

Graphs

Backtracking

Recursion : Function calling itself ∝

Solving a problem with the help of similar smaller problems
subproblems

$$sum(N) = 1 + 2 + 3 + 4 .. (N-1) + N$$

$$sum(N) = sum(N-1) + N$$

How to write recursive codes ?

Assumption : Decide what your function need to do and assume it does it

Main logic : Solve problem with help of subproblems

Base condition : 1) Decide when to stop your code
2) Decide when main logic fails

1) Find sum of N natural no.s using recursion

1) Assumption: Find sum of no.s from 1 ... N

```
int sum(N) {
    if (N == 0) { return 0 }
    return sum(N-1) + N
}
```

Base condition

Main logic:

$sum(N) = \underbrace{1+2+3+4 \cdots (N-1)} + N$

$= sum(N-1) + N$

$sum(0) = sum(-1) + 0$

Factorial : $4! = 4 \times 3 \times 2 \times 1 = 24$

$5! = 5 \times 4 \times 3 \times 2 \times 1 = 120$

Find factorial using recursion

Assumption: fact(N) will find product of N to 1

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

Main logic:

$fact(N) = N \cdot (N-1) \cdot (N-2) \cdots 1$

$fact(N) = N \cdot fact(N-1)$

$fact(0) = 1$

Function call stacking

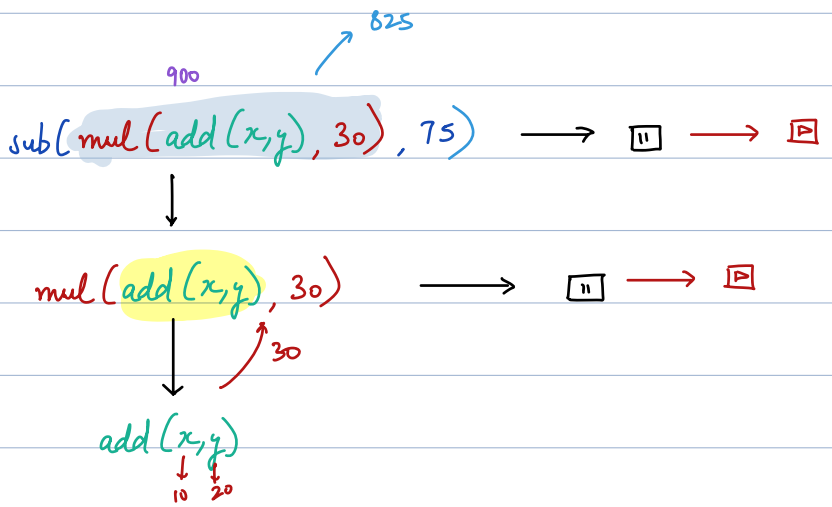
```
int add (x, y){
    return x+y
}

int mul(x, y){
    return x*y
}

int sub(x, y){
    return x-y
}
```

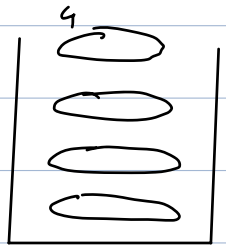
```
main(){
    x=10, y=20
    print (sub( mul(add(x,y), 30) , 75)
}
```

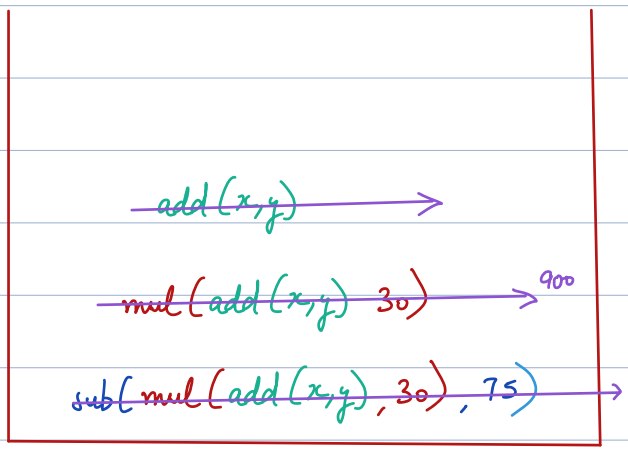
output: 825



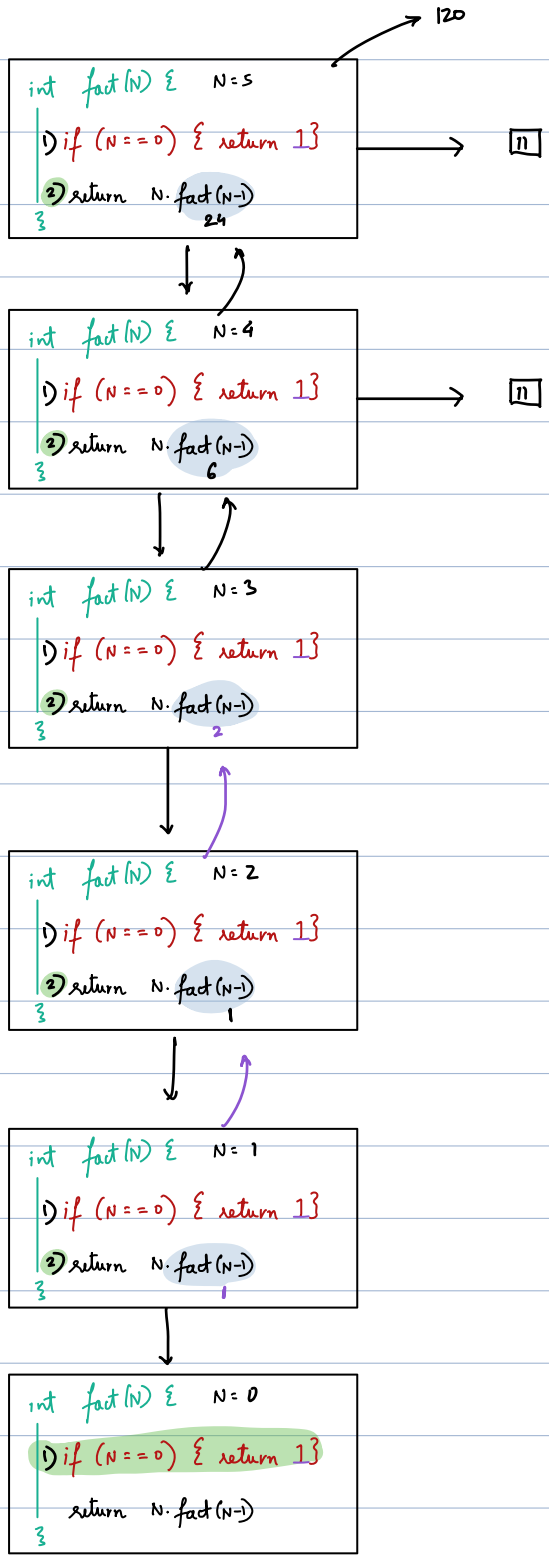
Stack

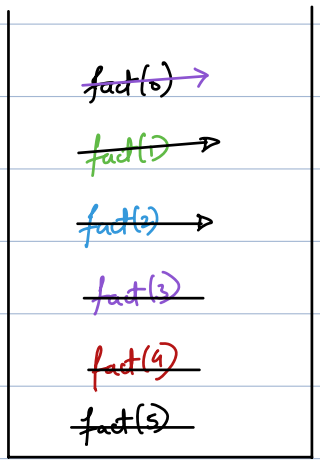
LIFO





10 s


120
int fact(N) { N = 5
1) if (N == 0) { return 1 }
2) return N * fact(N-1)
24
n
int fact(N) { N = 4
1) if (N == 0) { return 1 }
2) return N * fact(N-1)
6
n
int fact(N) { N = 3
1) if (N == 0) { return 1 }
2) return N * fact(N-1)
2
int fact(N) { N = 2
1) if (N == 0) { return 1 }
2) return N * fact(N-1)
1
int fact(N) { N = 1
1) if (N == 0) { return 1 }
2) return N * fact(N-1)
1
int fact(N) { N = 0
1) if (N == 0) { return 1 }
return N * fact(N-1)



1) Incorrect base case / No base case

→ Stack overflow

→ Memory limit exceeded

Fibonacci number

N =

Input:	0	1	2	3	4	5	6	7	8	9	10	...	N
fib() :	0	1	1	2	3	5	8	13	21	34			

fib(7) = fib(6) + fib(5)

fib(8) = fib(7) + fib(6)

N^{th} fib number

fib(N) =

N ≤ 1 return N

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

N=2

fib(N-1) + fib(N-2)

fib(1) + fib(0)

valid

N=1

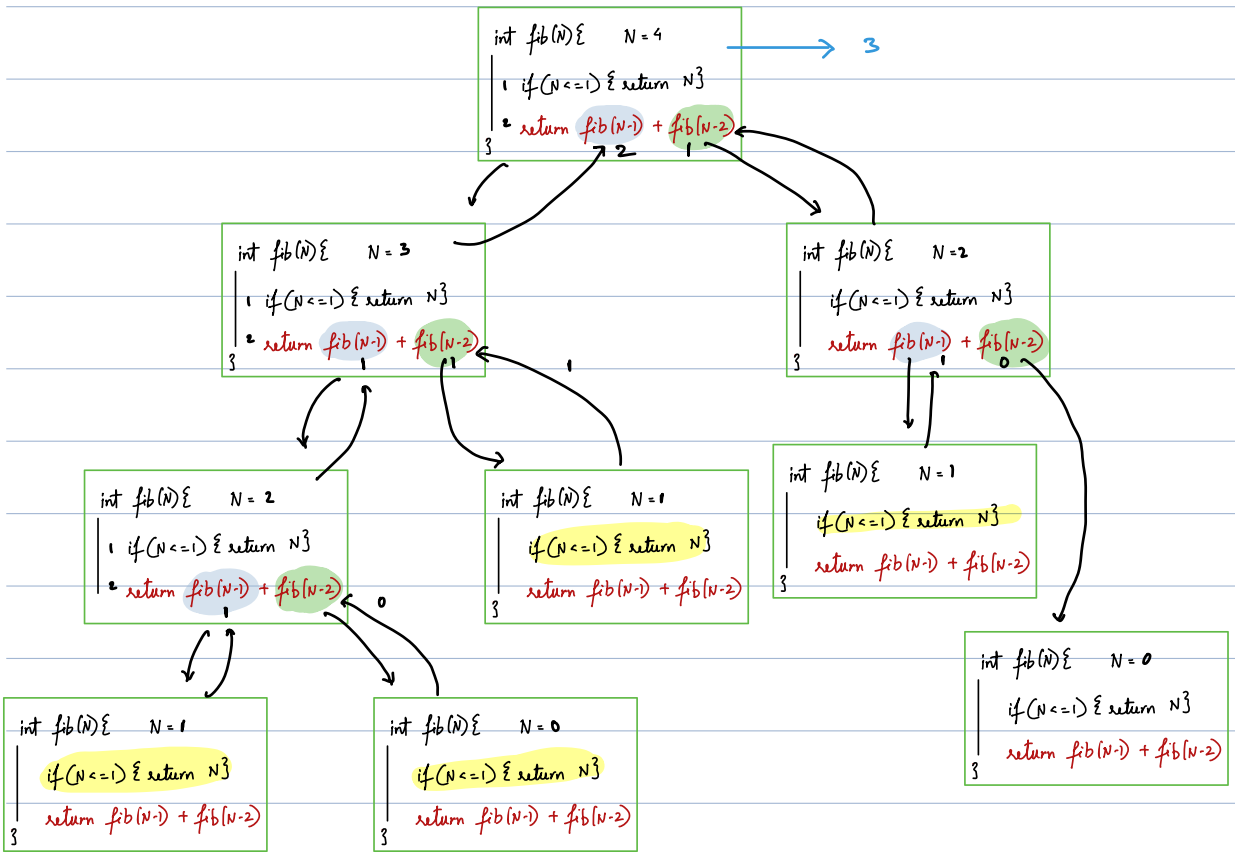
fib(0) + fib(-1)

main logic failij

N=0

fib(-1) + fib(-2)

main logic failij



Break (10:30 - 10:40)

Q3) Print all numbers from 1 - N in increasing order using recursion

N = 5

1 2 3 4 5

inc: Should print num from 1... N

```
void inc(N){
    if(N==0){return}
    inc(N-1)
    print(N)
}
```

TODO:

Dry run

print(N) 1 2 3 4 5 ... N

1 2 3 4 .. (N-1)

```
void inc(N){
    if(N==0){return}
    print(N)
    inc(N-1)
}
```

string = rev(string)

Q5) Given a substring of a string check if it is is a palindrome ?

0	1	2	3	4	5
M	A	D	D	A	M

0	1	2	3	4	5
G	O	O	D	O	G

A, s, e

bool palin(A, s, e)

A	rev(A)	
roar →	raor	✗
deed →	deed	✓
maddam →	maddam	✓
moon →	noom	✗

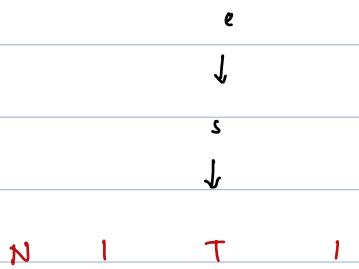
B & C

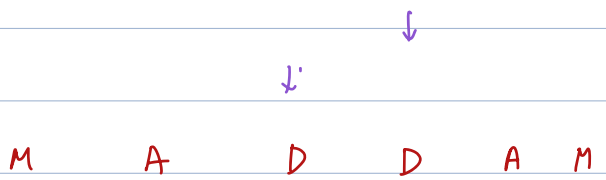
s = = e

return true

s >= e

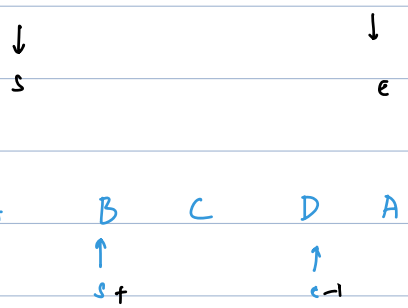
return true





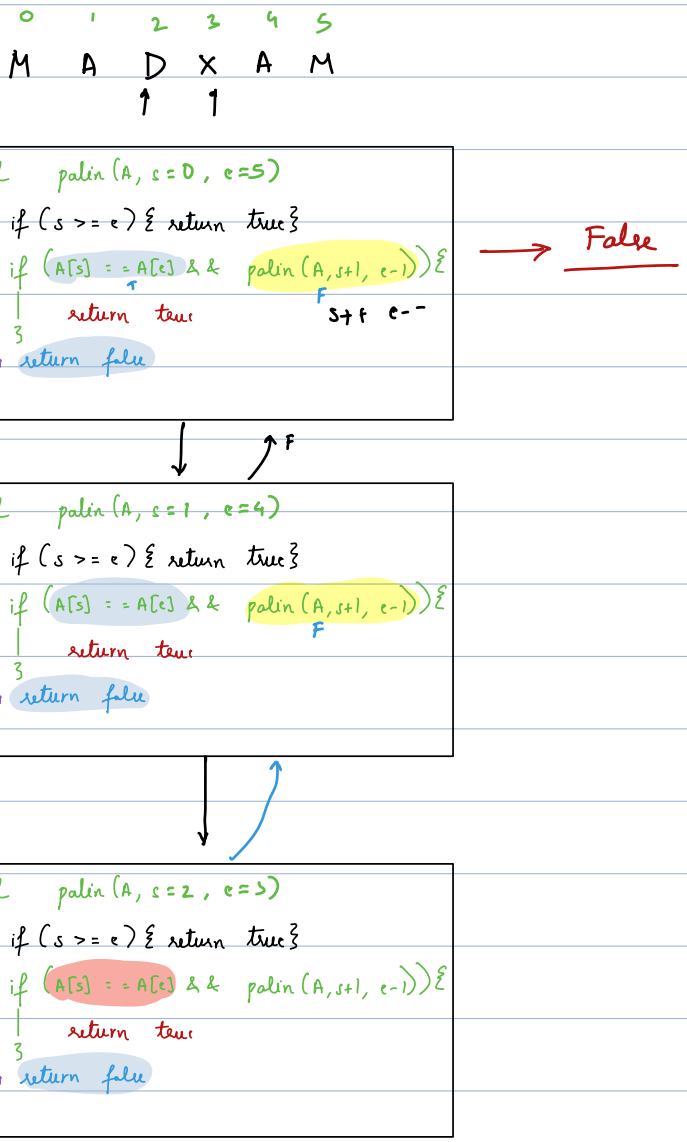
if (A[s] != A[e])

return false



Assumption: palin(A, s, e) checks if string A[s:e] is palindrome or not?

```
bool palin(A, s, e) {
1   if (s >= e) { return true }
2   if (A[s] == A[e] && palin(A, s+1, e-1)) {
3       return true
    }
4   return false
}
```



PS. on Sunday

5-6 problems

10 → >> 1 = 5

String, Arrays

2	10	% 2 = 0
2	5	% 2 = 1
2	2	% 2 = 0
	1	% 2 = 1

0101

0101

rev (0101) → 1010