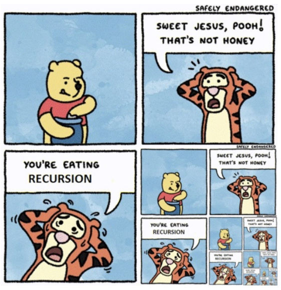
Recursion 1



AGENDA:

- ✓ What is recursion ?
- ✓ How to write recursion code ?
- ✓ How it works ?

Time and Space Complexity - Recursion 3

Advanced Batch

- Merge, Quicksort
- Trees , Heaps, Tries
- Backtracking
- DP
- Graphs

What is Recursion ?

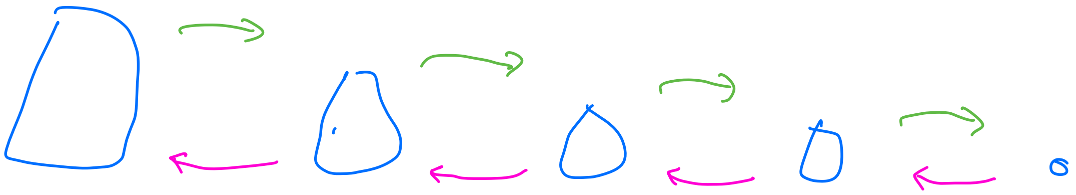
Function calling itself

Observations

1) Size keeps decreasing

2) Similar dolls

3) End doll



Solving a problem using a smaller instance of the same problem

subproblem

Example: Sum of first N natural numbers

$$\text{sum}(N) \rightarrow \underbrace{1+2+3+4\ldots\ldots+(N-2)+(N-1)}_{\text{Sum of all nos from } 1 \text{ to } (N-1)} + N$$

$$\text{sum}(N-1)$$

$$\Rightarrow \text{sum}(N) = \underbrace{\text{sum}(N-1)}_{\text{subproblem}} + N$$

Steps

1) Make an assumption

↳ Decide what your function does & trust that it will do it.

2) Main Logic

↳ Solve the big problem using a subproblem

3) Base Condition

↳ when your recursion stops

Assumption - sum(N) gives us sum of all natural no.s from 1 to N.

```
sum (int N) {

    // Base Condition
    if ( N==1 )              ← sum(1) = 1
        return 1


    // Main Logic
    return   sum(N-1) + N

}
```

Example: Factorial of N

$$fact(N) = \underbrace{1 \times 2 \times 3 \times 4 \ldots \ldots \times (N-1)}_{(N-1)!} \times N$$

$$fact(N) = \underbrace{fact(N-1)}_{\text{Subproblem}} \times N$$

Quiz 1

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

    return  fact(N-1) × N
}
```

Assumption

fact(N) given N factorial

← Base Case

← Main Logic

1! = 1

0! = 1

if N==1

1! = fact(0) × 1

↓

1 × 1 = 1

if N==0

0! = fact(-1) × 0

Example: Fibonacci Series

Golden Ratio

N=	0	1	2	3	4	5	6	7	8	9	10	11	
	1,	1,	2,	3,	5,	8,	13,	21,	34,	55,	89,	144	....

Given N, compute N^{th} fibonacci number

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

if N=0, → ans=1

$fib(0) = fib(-1) + fib(-2)$

if N=1, → ans=1

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

Assumption

fib(N) gives N^{th} fibonacci number.

```
fib( int  N) {
      if (N==0 or N==1)
           return 1

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

Base Case

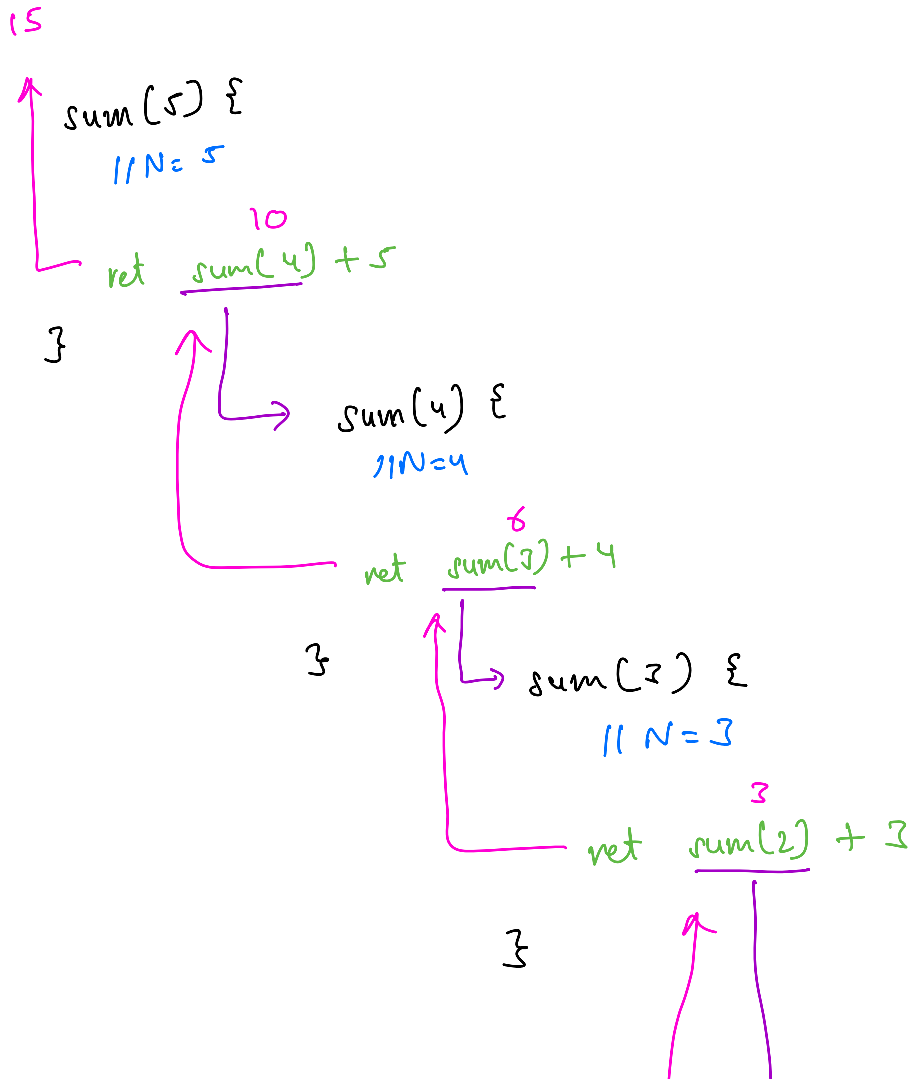
Main Logic

Break till 10:10 PM

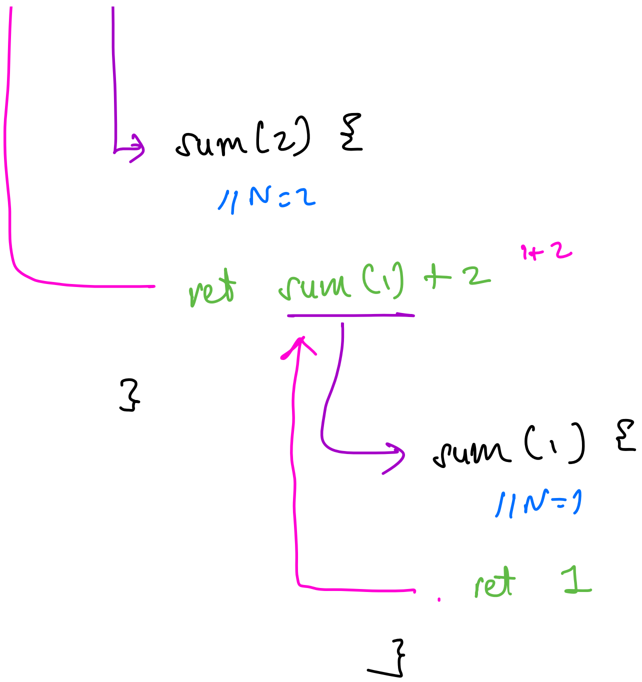
Dry Run - sum(N)

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

        return   sum (N-1)  + N

   }
```

N=5
↓
Ans = 15



```
15
sum(5) {
   // N= 5
               10
   ret  sum(4) + 5
}
          sum(4) {
           // N=4
                      6
          ret  sum(3) + 4
          }
                  sum(3) {
                    // N=3
                                3
                  ret  sum(2) + 3
                  }
```

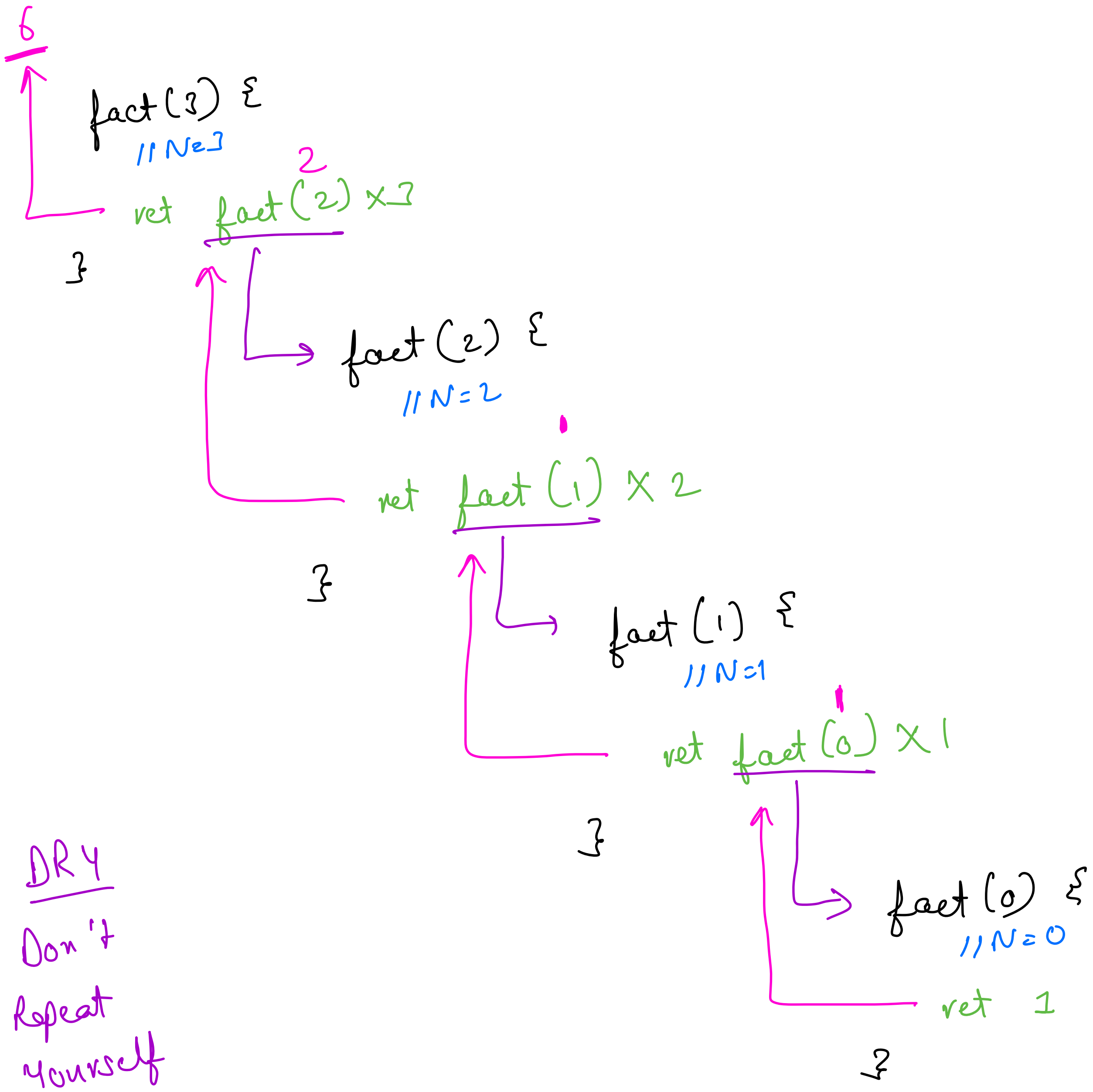


```
sum(2) {
  //N=2
  ret sum(1) + 2      1+2
}
    sum(1) {
      //N=1
      ret 1
    }
```

Dry Run - Factorial

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

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

N=3
↓
Ans = 6



```
6
fact(3) {
  // N=3
                 2
  ret  fact(2) x3
}
        fact (2) {
          // N=2
                        1
          ret  fact (1) X 2
        }
                 fact (1) {
                   // N=1
                                1
                   ret  fact (0) X 1
                 }
                          fact (0) {
                            // N=0
                            ret  1
                          }
```

DRY

Don't
Repeat
Yourself

Q1 Given a number N, print all numbers from 1 to N in increasing order using recursion.

incPrint (5) → $\underbrace{1, 2, 3, 4,}_{}$ 5

incPrint (4)
print (5)

incPrint (N)
↳ incPrint (N-1)
print (N)

```
incPrint ( N)  {
        if(N==0)  {
                return;
        }
        incPrint (N-1)
        print (N)
}
```

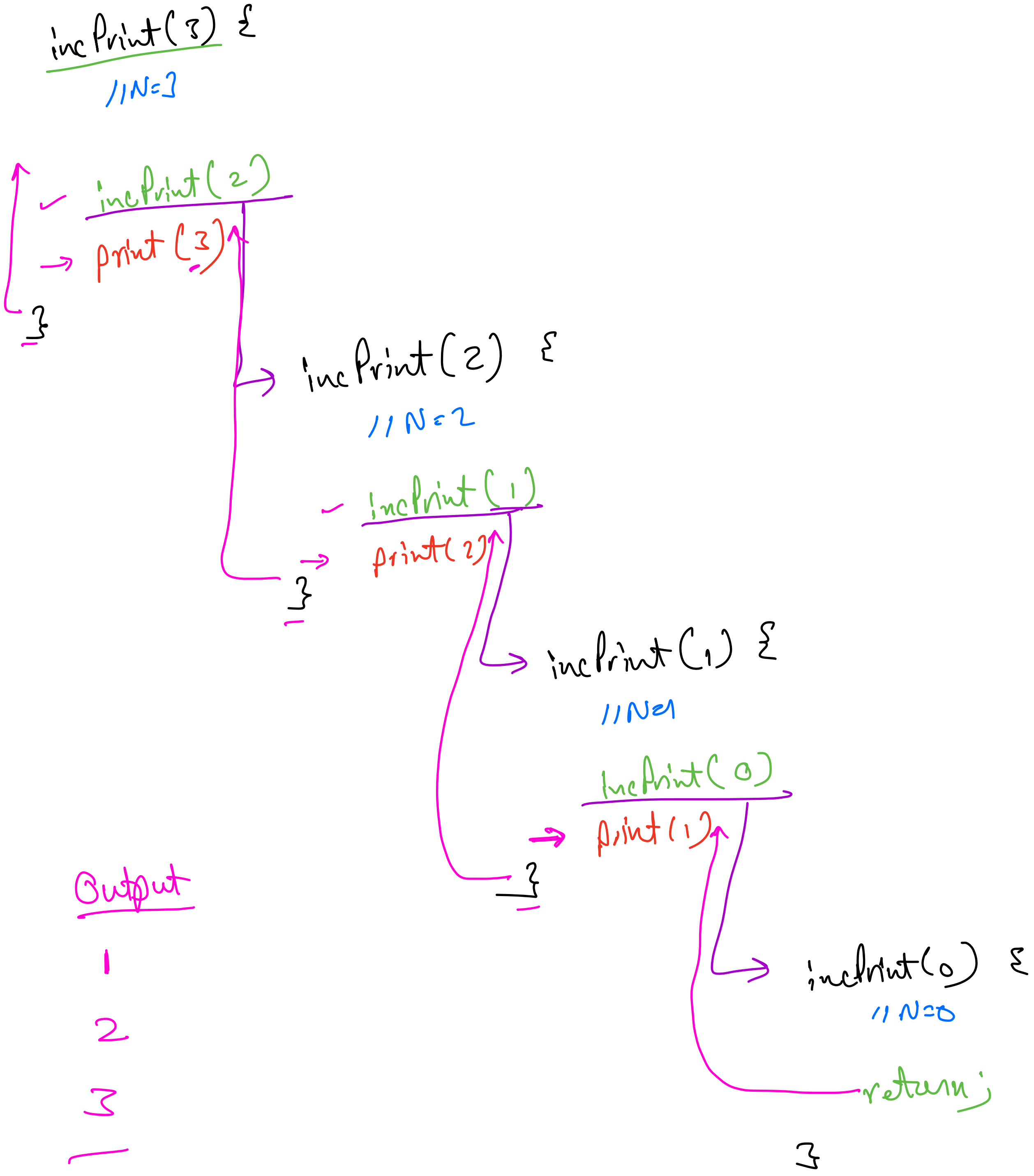
Assumption

incPrint(N) will print nums from 1 to N.

Base Case

Main Logic

KISS — Keep it simple, stupid.



```
incPrint(3) {
    //N=3

    incPrint(2)
    print(3)
}

incPrint(2) {
    //N=2

    incPrint(1)
    print(2)
}

incPrint(1) {
    //N=1

    incPrint(0)
    print(1)
}

incPrint(0) {
    //N=0

    return;
}
```

Output

1

2

3

Q2 Given a number N, print all numbers from N to 1 in decreasing order using recursion.

decPrint(5) → 5, 4, 3, 2, 1

print(5)

decPrint(N-1)

TODO

One or two line change from the prev problem

Q3 Given a string, check if it is palindrome using a recursive function.

aba malayalam

madam dad

racecar mom

---

a b c d d c b a

↑ ↑

j i

0 n-1

```
if ( s[i] == s[j] )
    i → i+1
    j → j-1
else
    return false
```

```
                                                 0        n-1
                                                 ↑         ↑
bool    isPalindrome ( string  s, int i, int j) {
                                                          Base
        if ( i >= j)                         ←            Case
              return   true


      if ( s[i] == s[j])
           return    isPalindrome (s, i+1, j-1)
                                                           Main
      else                                            ←    Logic
          return   false

}
```

Assumption

isPalindrome (s, i, j)

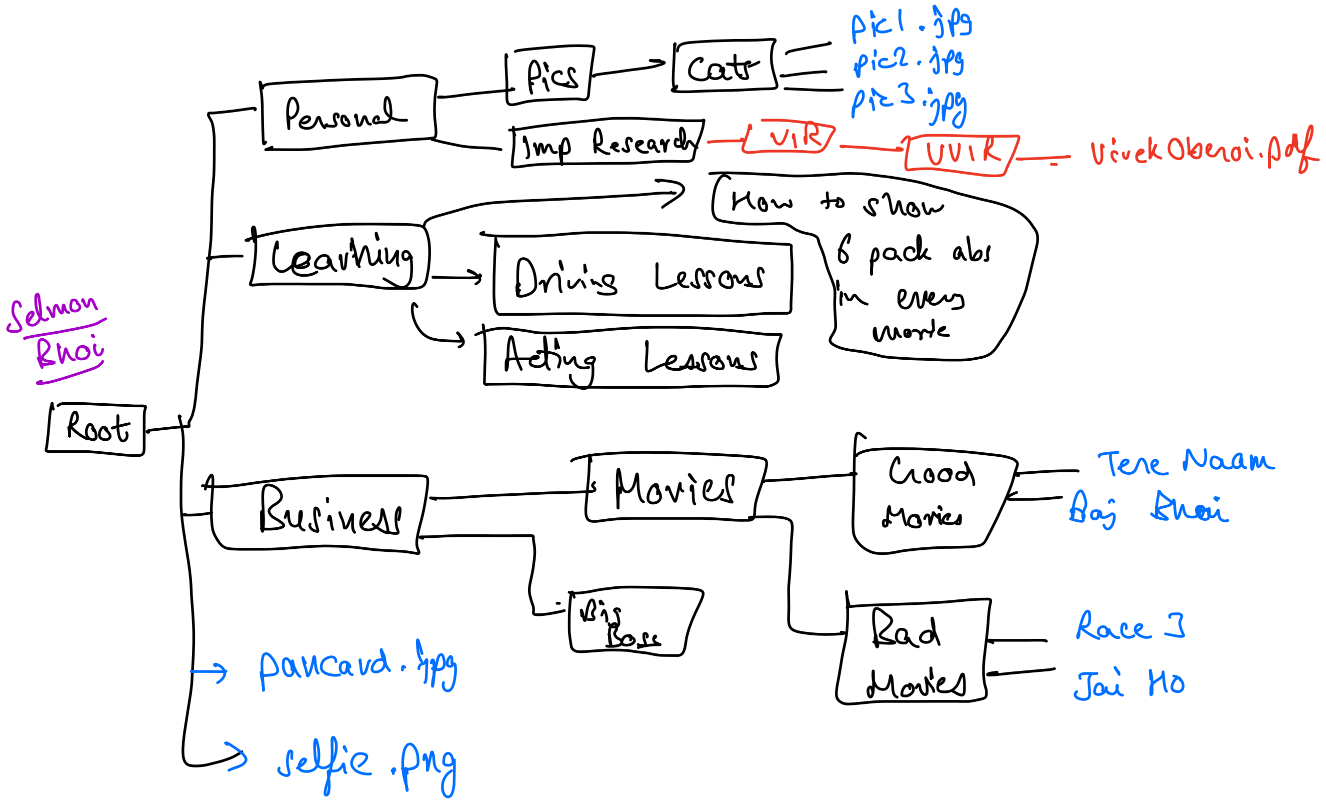
will check if

substring s[i j]

is palindromic

```
a   b   c   b   a
        ↑ ↑
        i j
```

Q4 Given a directory structure & some utility functions, search a file.



Utility functions

1. getAllDirectories (directoryName)

   Returns all directories inside it as a list → List [string]

2. getAllFiles (directoryName)

   Returns all files inside it as a list → List [String]

Assumption

search (D, F) returns true if the file F is present somewhere inside it.

```
bool    search( dir , fileName) {
        // Check files
        List <string> files = getAllfiles ( dir)
        for( i=0; i< files. size ; i++) {
                if( files [i] == fileName)
                        return True
        }

        // Check inside the folders
        List<string> folders = getAllDirectories ( dir)
        for( i=0; i< folders.size; i++) {
                if ( search (folders [i] , fileName ) )
                        return True
        }

        return false

}
```

Doubts

Thank You

$$z = \overset{\text{real}}{a} + i\overset{\text{imag}}{b}$$ ← self

$$x = c + id$$ ← x

$$res = a+c + i(b+d)$$

$$(a+ib)(c+id)$$

$$\Rightarrow ac + iad + ibc + i^2 bd$$

$$= (ac + i^2 bd) + i(ad+bc)$$

$$= \underbrace{(ac - bd)}_{\text{Real}} + i\underbrace{(ad+bc)}_{\text{Imag}}$$

Good Night

Thank You

Monday