DSA SERIES

- Learn Coding

Topic to be Covered today

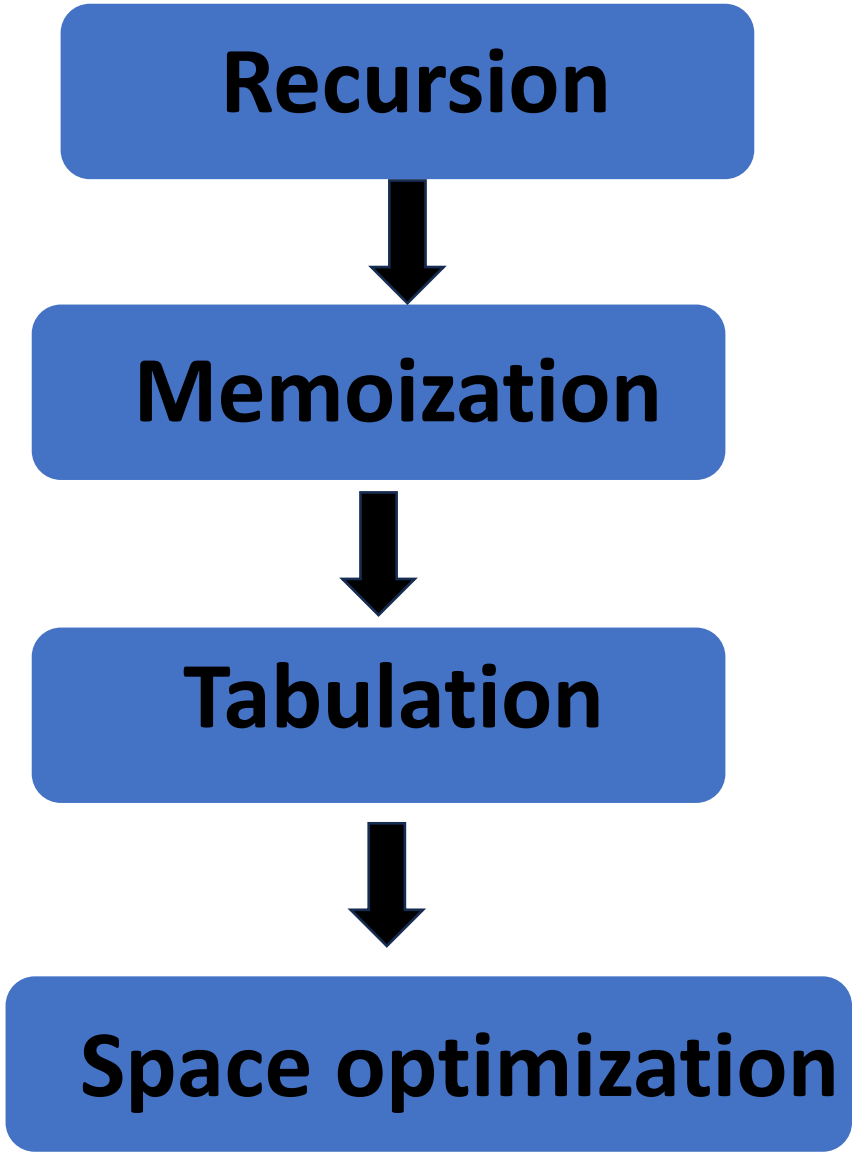
Dynamic Programming

Dp is a technique for solving complex problems by breaking them down into simple subproblems,solving each subproblems once and storing their results to avoid redundant computations .

It is mainly used when :

- The problem can be broken into overlapping subproblems.
- It has an optimal substructure.

(If the optimal solution of the main solution of the main problem can be constructed from the optimal solutions
of its subproblems.)


Recursion
Memoization
Tabulation
Space optimization

Example : Fibonacci Number

Formula : $F(n) = F(n-1) + F(n-2)$

Base cases : $F(0) = 0$, $F(1) = 1$

Recursion

```
int fibo(int n){

if(n<=1) return n;

return fibo(n-1) + fibo(n-2);

}
```

Memoization

```
int fibo(int n , vector<int> &dp){

if(n<=1) return n;

if(dp[n] != -1) return dp[n];

return dp[n]=fibo(n-1,dp) + fibo(n-2 ,dp);

}
```

Tabulation

```
int fibo(int n){
vector<int> dp(n+1);
dp[0] = 0;
dp[1] = 1 ;
for(int i =2;  i<=n ; i++){
dp[i] = dp[i-1] + dp[i-2];
}
return dp[n];
}
```

Space Optimisation

```
int fibo(int n) {

int prev2 = 0 , prev1 = 1;

for(int i =2;i<=n;i++){

int curr = prev2 + prev1;

prev2 = prev1;

prev1 = curr;

}

return prev1;

}
```

Learn coding

THANK YOU