Dynamic Programming :→

* Fibonacci Series → 1D DP Algo Flow

→ Recursive Approach :→

fib(n) = fib(n-1) + fib(n-2)

Top Down ↑ Complex Recursive Calls → $O(2^n) + O(n)$

→ Memoization Approach :→ $O(n) + O(n)$

* dp[] array (all -1) values) → Initialization
* Known values are not calculated again.
* if (dp[n] != -1) return dp[n];

→ Tabulation Approach :→ $O(n) + O(n)$

Known Values are Stored :

dp[0] = 0; dp[1] = 1;

now loop from 2 → n

Bottom Up

for (i → 2 to n) {

dp[i] = dp[i-1] + dp[i-2];

}

dp | 0 | 1 | 2 | 3 | 5 | 8 |

* Fibonacci : Space Optimization :→

Question :→ Can we eliminate the array?

O(n) → O(1) (Just vars)

n = 35 ?

int p2 = 0; (n = 6)

int p1 = 1;

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

int curr = p1 + p2;

p2 = p1;

p1 = curr;

}

return p1;

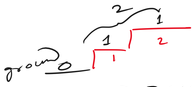
p2	p1	curr
0	1	1
1	1	2
1	2	3
2	3	5
3	5	8

(8)

TC → O(n)

SC → O(1)

2 vars p2, p1



f(2) = f(1) + f(0)

= 1 + 1

2 → 2 ways

1 + 1 | 2

ways(3) → 1 + 1 + 1 | 1 + 2 | 2 + 1

f(3) = f(2) + f(1)

2 + 1

= 3

ways(4) → 1 + 1 + 1 + 1 | 1 + 1 + 2 | 1 + 2 + 1 | 2 + 1 + 1 | 2 + 2

f(4) = f(3) + f(2)

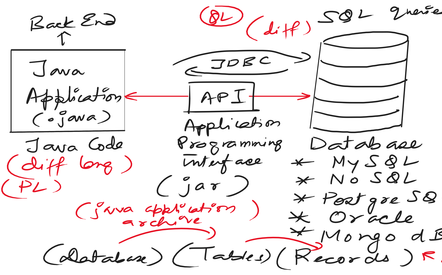
= 3 + 2

= 5

450 DSA → Topic Wise

Logic → Prapinata → Top 100

JDBC



Sawan (BOST)

Third-party applications

* jolibo →
* make my trip
* ease my trip
* yatra.com
* sastasafar.com

del boy ↑ function

API → (connects on) send receive data

react owner fro api (fun)

API-key (pay)

Airlines Services

Indigo Sawan

Flight

Boeing F9736

22 Seats

Vacant

Request → GET POST PUT PATCH

DSL

Components of JDBC :→

Jar (mysql) Driver

Java App

java.sql

Driver Manager

CRUD

Mongo | Oracle | MySQL | NoSQL

(I) Connection

(II) Driver → type of DB

(III) Statement, Prepared Statement

(IV) → executeQuery | execute Update

(V) Result Set → Stores the results

(VI) SQL Exception → All Exceptions