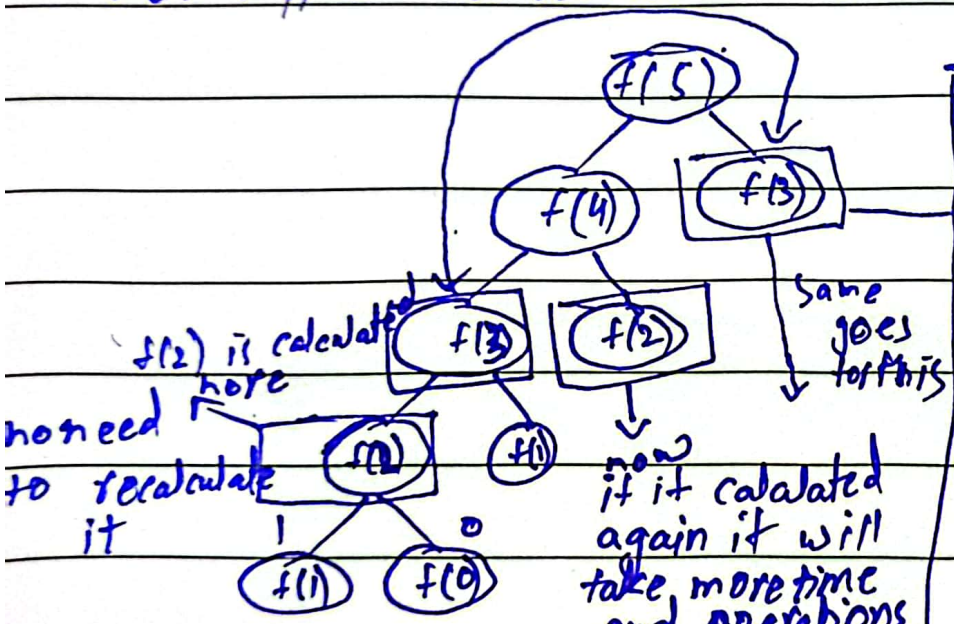
DATE 19 DAY | 10 MONTH | 24 YEAR

Dynamic Programming

for fibbonaaci numbers

Lets suppose for recursive fibonaadi number.



Solution

use memoization

create array of n+1 size.

f(2) f(3) f(4) f(5)

-1	-1	1	2	3	5
0	1	2	3	4	5

use these values

If f(2) is calculate store value of f(2) here

declare array dp[n+1]

```
f(n)
{ if (n<=1)
    return n;
  if (dp[n] != -1) → return dp[n];
  return dp[n] = f(n-1)+f(n-2);
}
```

T.C → O(N)

S.C → O(N) + O(N)

array

Recursion → Tabulation (Bottom up)

Base case + 0 required

Tabulation

change base case to loop format & reverse the case

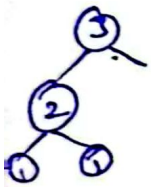
if(n<=1) → i=2

```
for (i=2; i<n; i++)
{ dp[i] = dp[i-1] + dp[i-2]; }
```

2024/10/19 16:57

prev2 → dp[0] = 0 ; prev → dp[1] = 1

for(i=2 → n)
{ curr[i]
dp[i] = dp[i-1] + dp[i-2]
prev2 = prev;
prev = current; }



Types of recursion problem
1) count the numbers of way / Try all possible way
2) Figure out best way

Shortcut for all problems in dynamic programming
1) Try to represent in term of index
(2) Do all possible stuff on that index according to problem statement
(3) sum up all stuffs → if question say count all ways
if question ask minimum → min (of all stuff)
for find max → max of all stuff)

Example stairs problem

```
if (index)
if (index == 0)
    return 1;
```