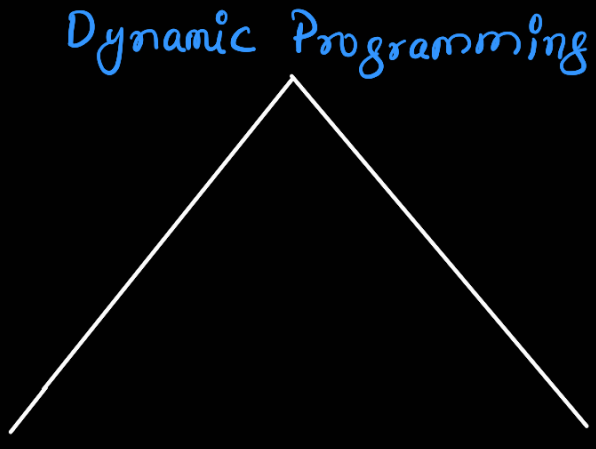
Those who cannot remember the past are condemned to repeat it

~ Dynamic programming



Memoization (Top down)
- Recursion
- \+
- a data structure to store result of computed function calls so that we avoid calling overlapping subproblems

doesn't mean necessarily, we can also write recursion to from bottom to up.

Tabulation (Bottom up)
- Going from Base case to required case.

Fibonacci problem:

	T(n)	S(n)
Brute force :	$O(2^n)$	$O(n)$ (i.e. max recursion depth)
Memoization :	$O(n)$	$O(n)+O(n)$
Tabulation :	$O(n)$	$O(n)$

we can do even more space optimization by simple observation.

for any $f(i)$ we just need previous two values i.e. $f(i) = f(i-1) + f(i-2)$

values : f(1) = f(1-1) + f(1-2)

So there is no need to store all values. we can just carry previous two values in each iteration.

$T(n)$: $O(n)$

$S(n)$: $O(1)$

```
prev1 = 0
prev2 = 1

for (i : 2 to n)
{   curr = prev1 + prev2
    prev1 = prev2
    prev2 = curr
}
```