Maximum Ice Cream Bars

I/p: costs = {1, 3, 4, 2, 1}

coins = $\boxed{7}$

o/p: ⇒ $1 + 3 + 2 + 1 = \boxed{7}$ → cost

→ O/p → $\boxed{4}$

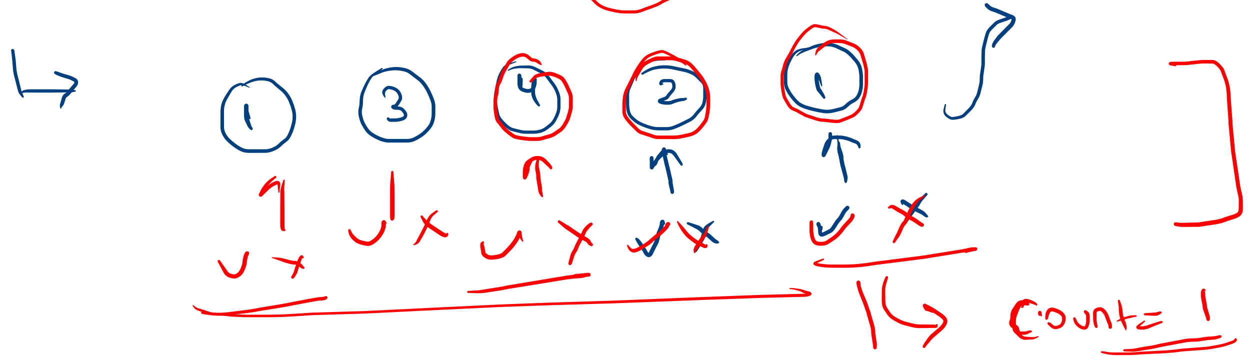
{ maximum no. of Ice Cream, Boy can Buy

I/P: costs = {1, 3, 4, 2, 1}

coins = 7

sol: → Sometimes, we use our 100% Brain × DP TLE

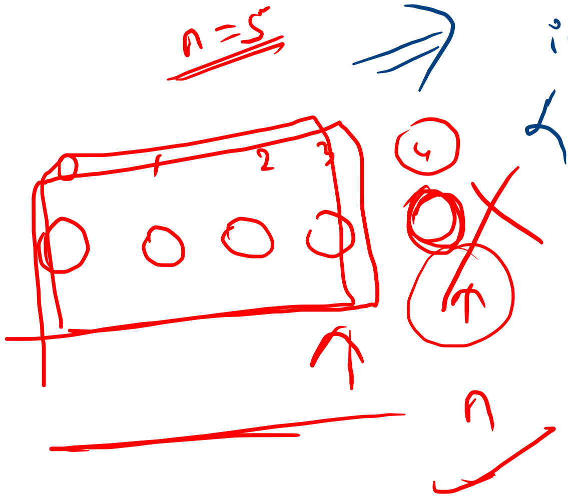
And, Start using DP.



Recursion

count = 1

Coins -= Cost[i-1]



```
int fun( vector<int> costs, int n, int coins)
{
    if (n==0 || coins==0)
        return 0;
    if ( cost[n-1] <= coins)
    {
        return max( 1+fun( costs, n-1, coins - cost[n-1])
                    fun( costs, n-1, coins) )
    }
    else
    {
        return fun(costs, n-1, coins)
    }
}
```

Base Cond'

$O(N^2)$ X

$O(n^2)$

memo
&
Top

Greedy

I/p: costs = {1, 3, 4, 2, 1}

coins = 7

o/p: → Common sense,

↳ Start Picking the ice-creams which has lower → greedy

Cost and increase the count

→ How to find the increasing order Costs

↳ Sort the costs[] array

↳ {1, 1, 2, 3, 4} 5/6 → 4

→ ↑ ↑ ↑ ↑

Count = ~~1~~ ~~2~~ ~~3~~ 4

~~7~~ ~~6~~ ~~5~~ ~~3~~ 0.

I/p: costs = { 1, 3, 4, 2, 1 }

coins = 7