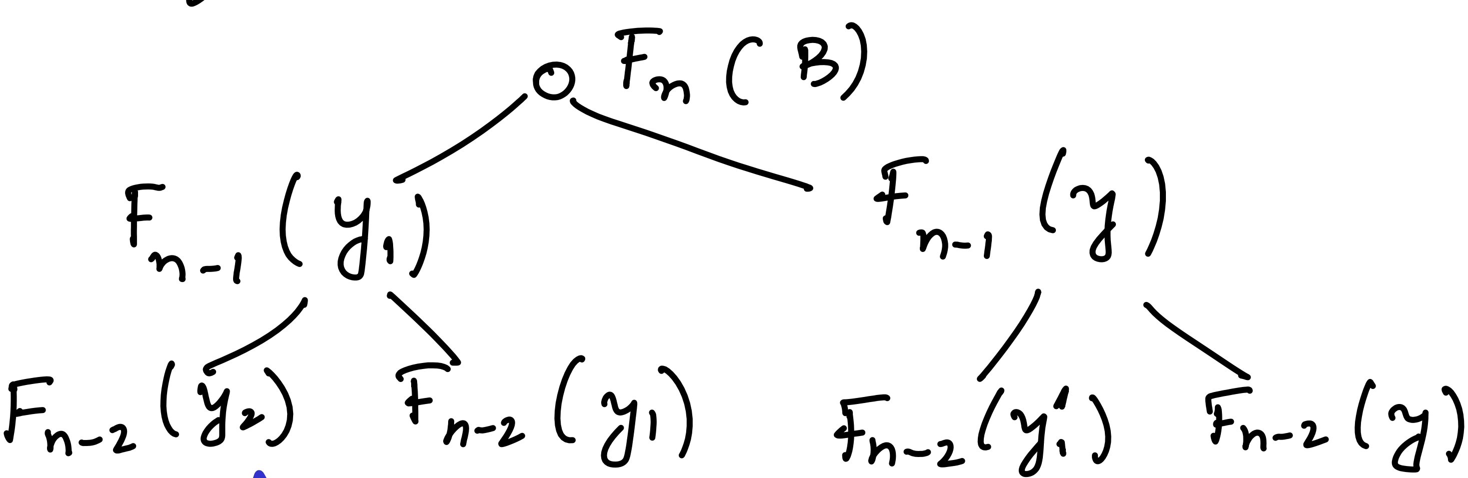
1. Writing a recurrence for the specific problem to express the optimality in terms of smaller subproblems (Optimal Substructure property)

For knapsack

$$F_i(y) = \max\left\{F_{i-1}(y - w_i) + p_i,\ F_{i-1}(y)\right\}$$

2. We must have well defined base cases for the induction



may be $y_2 = y_1'$

$\Rightarrow$ Repetition in computation

To avoid repetition, we would like to store what we have computed (in a table) so that we can re-use it that term needs to be computed under another node

Some amount of space may be required to store the values computed → especially if they will be needed in future

→ Space/Time trade off

→ An ordering of computation is essential so that the present term can be computed from prior terms

1	2	3	4	5	6	7	8	9
1,	5,	2,	8,	50,	30,	25,	11,	40
1	2	2	3 (3)	4 (4)	4 (4)	4 (4)	4 (4)	5 (8)

Problem: Find the longest monotonically increasing subsequence

$\boxed{5, 50, 25}$

not monotonically increasing

How many subsequences in a sequence of length n? $\sim 2^n$

In any sequence of length n, there is either a mono increasing subseq of length $\lfloor \sqrt{n} \rfloor$ or a " decreasing " " " $\lfloor \sqrt{n} \rfloor$

<u>Erdos - Szekeres thm</u>

S_i = as the longest monotonic subsequence ending in position i

$|S_i|$ is the length

$\max_i S_i$

S_i can be obtained from $j \leq i$ such that $x_j < x_i$ and choosing the longest S_j among them