Longest Common Subsequence

String 1 → b d

String 2 → a b c d

Length → 2 (bd)

task → Tuesday

i → row number

J → column number

Tabulation

		0	1	2	3	4
		0	a	b	c	d
0	0	0	0	0	0	0
1	b	0	0	1	1	1
2	d	0	0	1	1	2

m = 3

n = 5

Length of LCS

return → LCS[m-1][n-1]

String of LCS :- bd

LCS → Table name

```
if string1[i] == string2[j]:
    LCS[i,j] = 1 + LCS[i-1][j-1]
else:
    LCS[i,j] = max(LCS[i-1][j], LCS[i][j-1])
```

String1 : A G G T A B
0 1 2 3 4 5

String2 : G X T X A Y B
0 1 2 3 4 5 6

Longest common subsequence

↳ GTAB ←

Expected Output

Length = 4

(m+1)

		0	G	X	T	X	A	Y	B
	0	0	0	0	0	0	0	0	0
	A	0	0	0	0	0	1	1	1
	G	0	1	1	1	1	1	1	1
	G	0	1	1	1	1	1	1	1
(m+1)	T	0	1	1	2	2	2	2	2
	A	0	1	1	2	2	3	3	3
	B	0	1	1	2	2	3	3	4

GTAB

0 1

X = BD

Y = A B C D
0 1 2 3

i=0 or j=0
↳ dtable = 0

i=1 or j=1

X[i-1] == Y[j-1]

max capacity

$m = 8$

$n = 4$

0-1 Knapsack

$P = \{1, 2, 5, 6\}$ (1 2 3 4)

$\omega = \{2, 3, 4, 5\}$

$x_i = \{0/1, 0/1, 0/1, 0/1\}$

$2^4 = 16$

Optimization

↳ $\max \sum P_i x_i$

Constraint $\sum \omega_i x_i \leq m$

1	1	1	1
1	1	1	0
1	1	0	1
1	1	0	0
1	0	1	1
1	0	1	0
1	0	0	1
1	0	0	0
0	1	1	1
0	1	1	0
0	1	0	1
0	1	0	0
0	0	1	1
0	0	1	0
0	0	0	1
0	0	0	0

$n = 4$ ⇓ 16 Possible Combination

n Objects

↳ 2^n

$\Theta(2^n)$

↳ Exponential

↓

Polynomial

Recursive

P = {1, 2, 5, 6} (0 1 2 3)

wt = {2, 3, 4, 5} (0 1 2 3) — 9

m = 8 capacity (max)

KnapSack(m, wt, P, n):

if wt[n-1] > m:

* Skip that object n = 4

return KnapSack(m, wt, P, n-1)

↳ Recursion

else:

* Skip that object
* include that object

} max

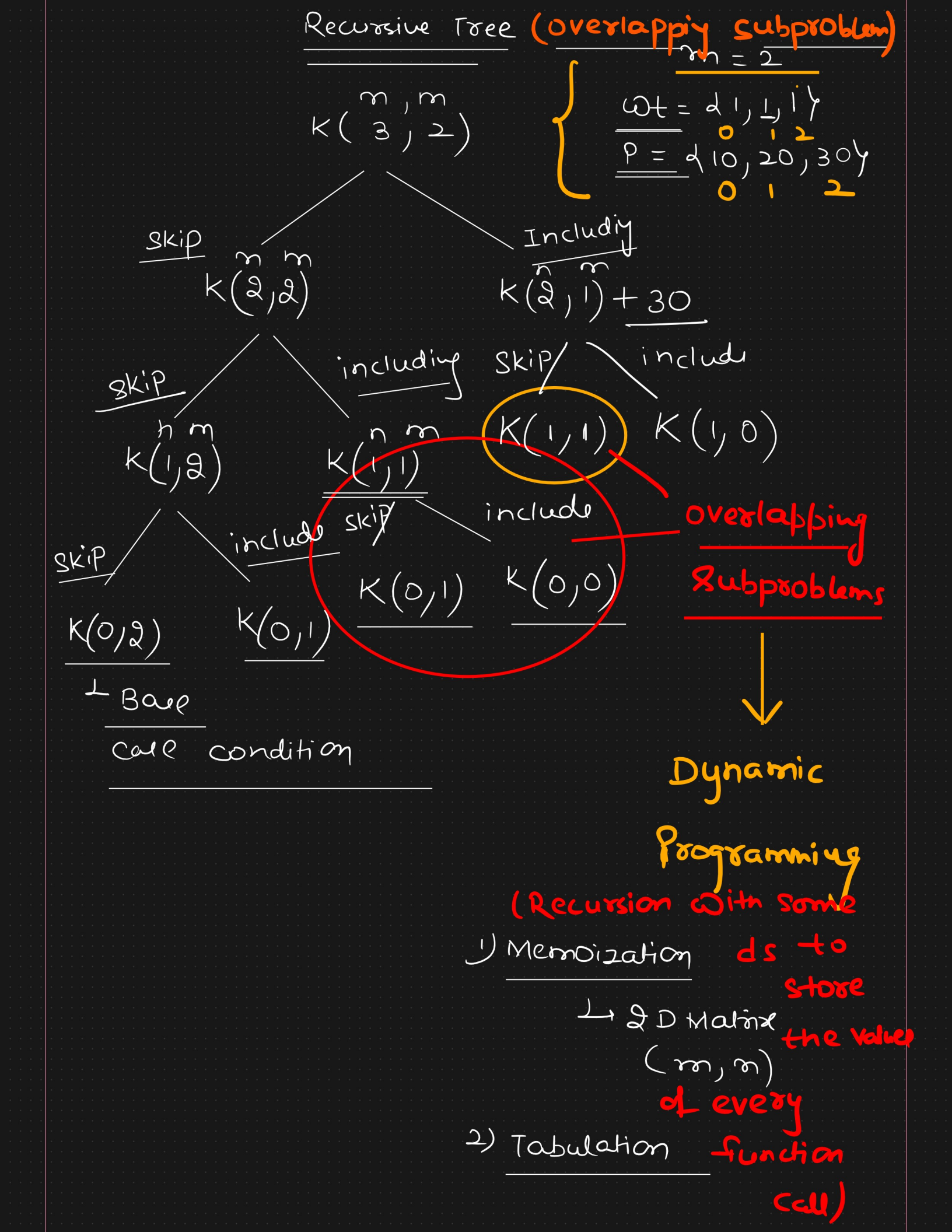
return max(KnapSack(m, wt, P, n-1), (skip element)

P[n-1] + KnapSack(m - wt[n-1], wt, P, n-1))

Recursion

1

including the element


Recursive Tree (overlapping subproblem)
m = 2
wt = {1, 1, 1}
0 1 2
P = {10, 20, 30}
0 1 2
K(3, 2)
Skip
Including
K(2, 2)
K(2, 1) + 30
Skip
including
Skip
including
K(1, 2)
K(1, 1)
K(1, 1)
K(1, 0)
Skip
include
Skip
include
overlapping subproblems
K(0, 2)
K(0, 1)
K(0, 1)
K(0, 0)
Base case condition
Dynamic Programming
(Recursion with some ds to store the value of every function call)
1) Memoization
2D Matrix (m, n)
2) Tabulation