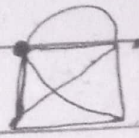
DATE: 21-03-25 DAA DAY: Friday

Exhaustive Approach

Optimization (Brute Force)

→ Traveling Salesman Problem:

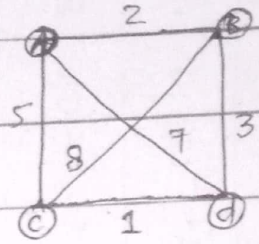
Hamiltonian circuit.



etc

. Cost Minimum ho

$\frac{1}{2}(n-1)!$



→ Knapsack Problem:

2^n → subsets pick kr sakty

. Cost / Benefit Maximum

. Constraint → Size of bag

item	Weight	Value of bag
0	0	0
{1}	7	$42
{2}	3	$12
{3}	4	$40
{4}	5	$25

→ try every combination

→ pehly (1,1) phir (2,2) phir (3 3)

* 0/1 knapSack → either you have to pick item or not.

DATE: DAY:

→ Assignment Problem

n!

→ Cost minimization

Person \ Job	1	2	3	4
1	9	2	7	8
2	6	4	3	7
3	5	8	1	8
4	7	6	9	4

→ aik person ki position fix kr k combination banaeingy or ye her bunday k liye ho ga.

→ Hungarian Method exponential to polynomial

cost matrix

9	2	7	8		7	0	5	6
6	4	3	7	→	3	1	0	4
5	8	1	8		4	7	0	7
7	6	9	4		3	2	5	0

row reduction:
→ Identify row minimum
→ Then substract it from that row

DATE: DAY:

3- Identify Column Ro minimum

4- Then subtract it from column

7	0	5	6
3	1	0	4
4	7	0	7
3	2	5	0

→

4	0	5	6
0	1	0	4
1	7	0	7
0	2	5	0

Not optimal

→ Draw horizontal or vertical lines that covers maximum zero. (no diagonal lines).

* Maximum Order of matrix lines

→ If all lines are cut then the sol/matrix will be optimal.

For Not Optimal

4	0	5	6
0	1	0	4
1	7	0	7
0	2	5	0

DATE: _____ DAY: _____

from \ to	x	y	z
a	250	400	350
b	400	600	350
c	200	400	250

Cost Matrix

250	400	350		0	150	100
400	600	350	R.R →	50	250	0
200	400	250		0	200	50

				x	y	z
0	0	100	a	0	0	100
-50	100	0	→ b	50	100	0
0	50	50	c	0	50	50

400+350+200 = 950

Q#2

90	75	75	80		15	0	0	5
35	85	55	65	→	0	50	20	30
125	95	90	105	R.R	35	5	0	15
45	110	95	115		0	65	50	70

C.C-

UC-R → −
C-C → +

DATE:

15	0	0	0
0	50	20	25
35	5	0	10
0	65	50	65

→

15	0	0	0
-5	45	15	20
30	0	-5	5
-5	60	45	60

UC-V → −
IV → +

20	0	5	0
0	45	20	20
35	0	0	5
0	60	50	60

16-04-2025 DAA

Backtracking: (DFS)

General ← NP-problem: Non-polynomial
↳ NP run time
Non-optimization Problems

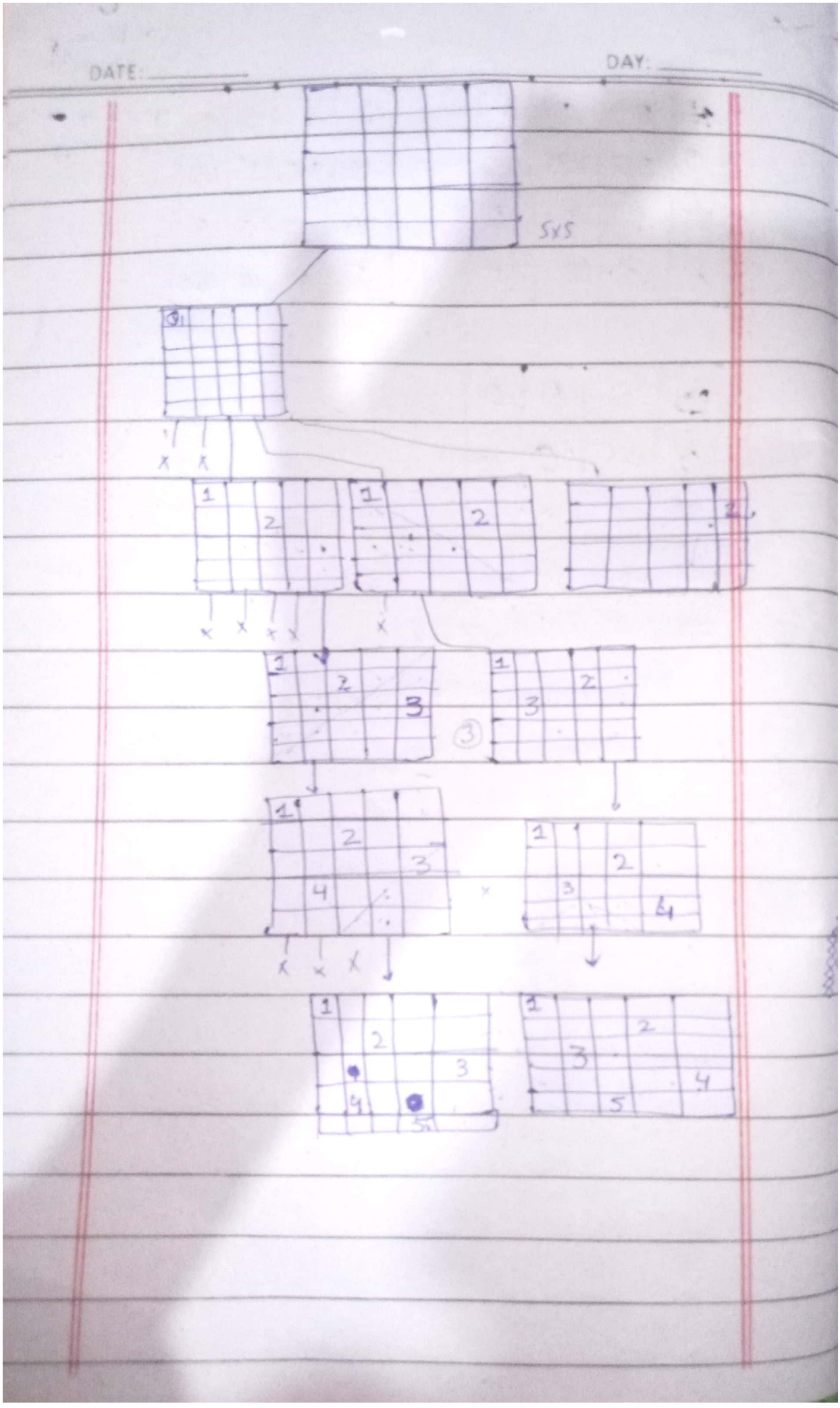
n-Queens Problem:

n = 1  1X1

n = 2  2X2 No solution exists

State-space Tree
→ Initial State
→ promising
→ Non-promising (Sol. exists nae kota)

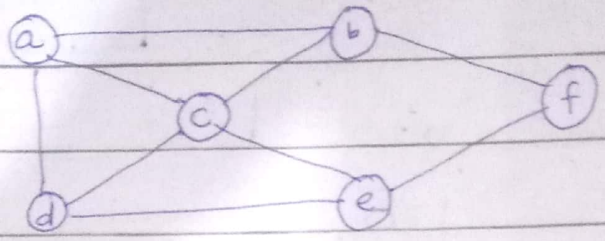
DATE: _____ DAY: _____



DATE: 18-04-25 DAY: _____

Backtracking:

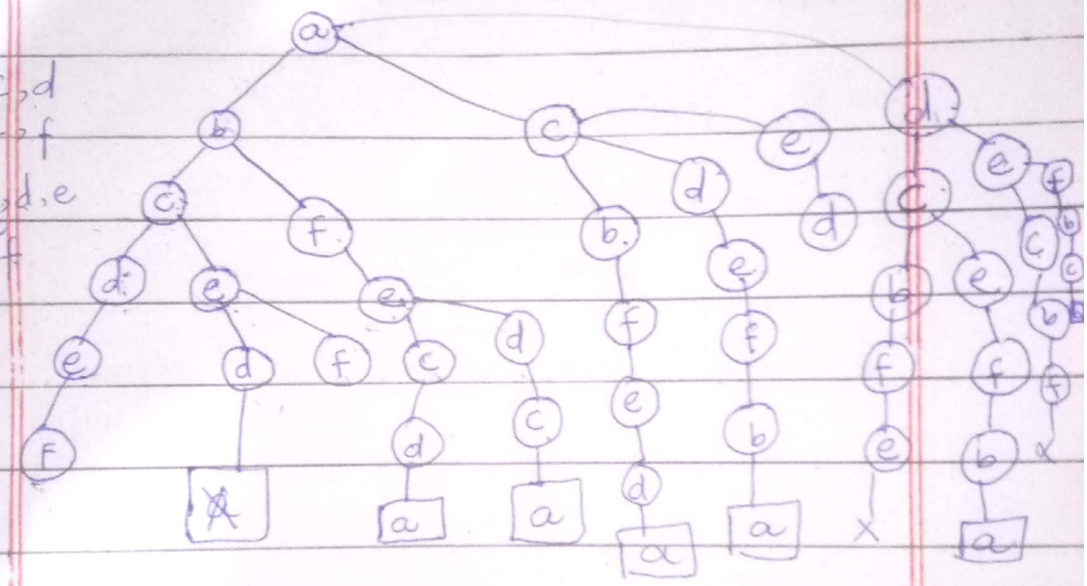
→ draw complete state space tree to solve hamiltonian circuit using back tracking



Hamiltonian Circuit:

→ Alphabetical order

a → b, c, d
b → a, c, f
c → a, b, d, e
d → a, c, e
e → c, d, f
f → b, e



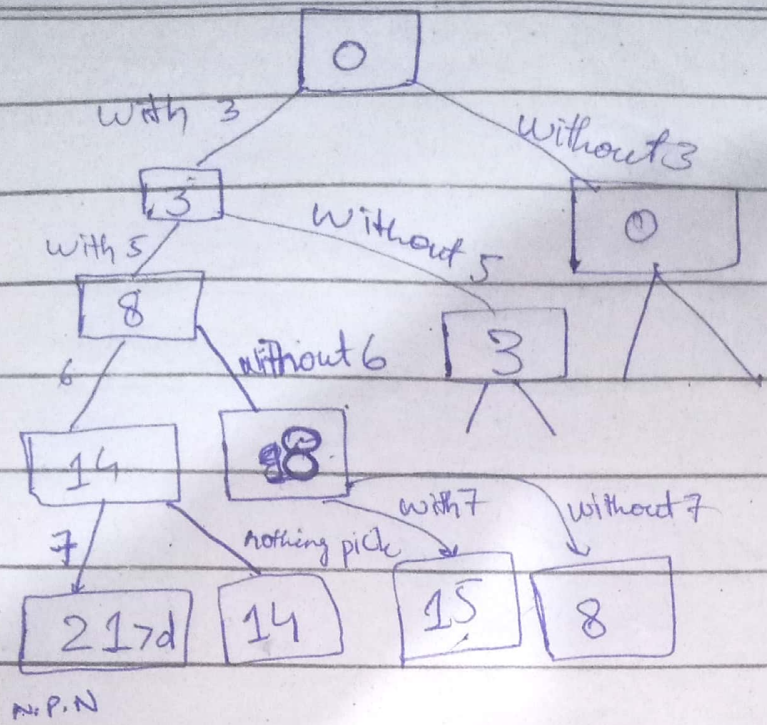
Subset Sum Problem:

Set: {3, 5, 6, 7}

d = 15

If Sum of all values > than d then don't solve. if equals than the same set is the solution.

DATE: DAY:

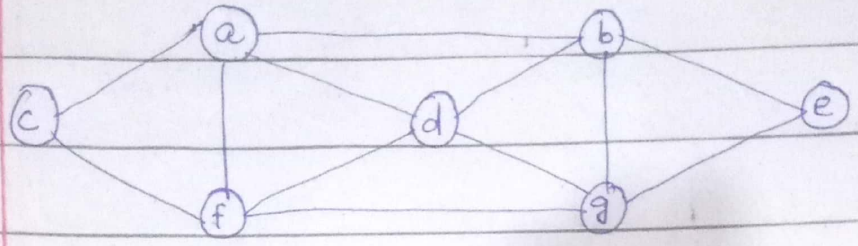


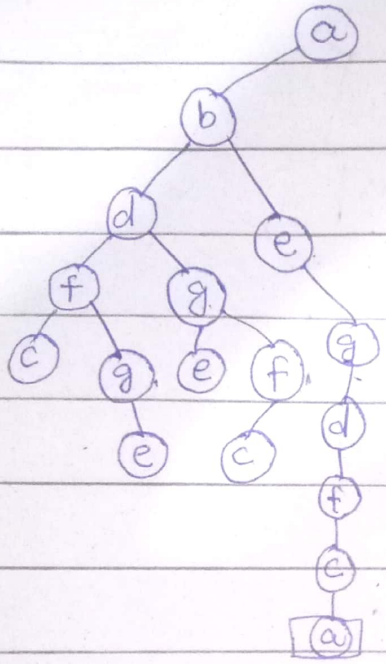
→ For knapsack problem, size of bag is constraint.

Entended theorem ≠ tight bound.

DATE: Backtracking DAY:

5





a → b, c, d, f
b → a, d, e, g
d → a, b, f, g
f → a, c, d, g
c → a, f
g → b, d, e, f
e → b, g

8.

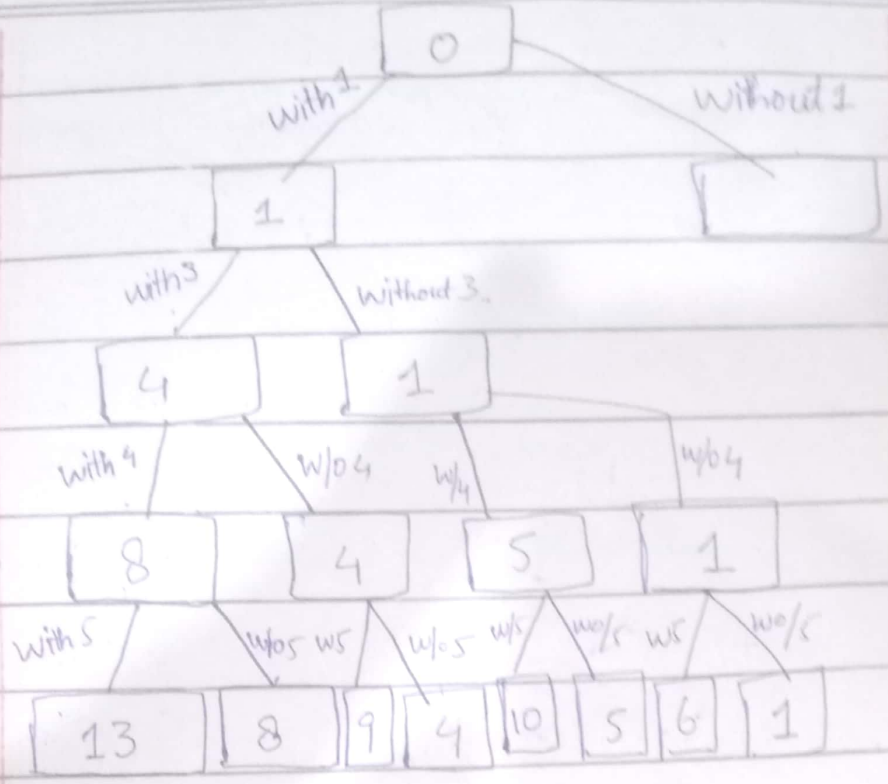
Subset Sum d = 11.

A = {1, 3, 4, 5}

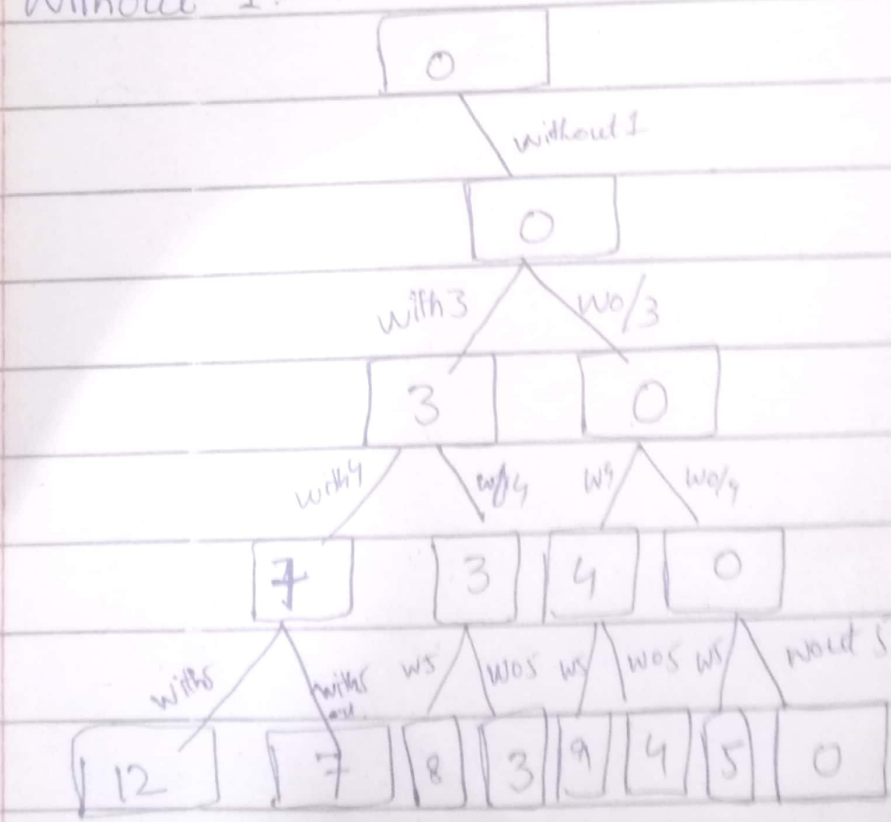
7-a Generate all permutations of {1, 2, 3, 4} by backtracking:

8 = set = {1, 3, 4, 5}

DATE: DAY:



Without 1:

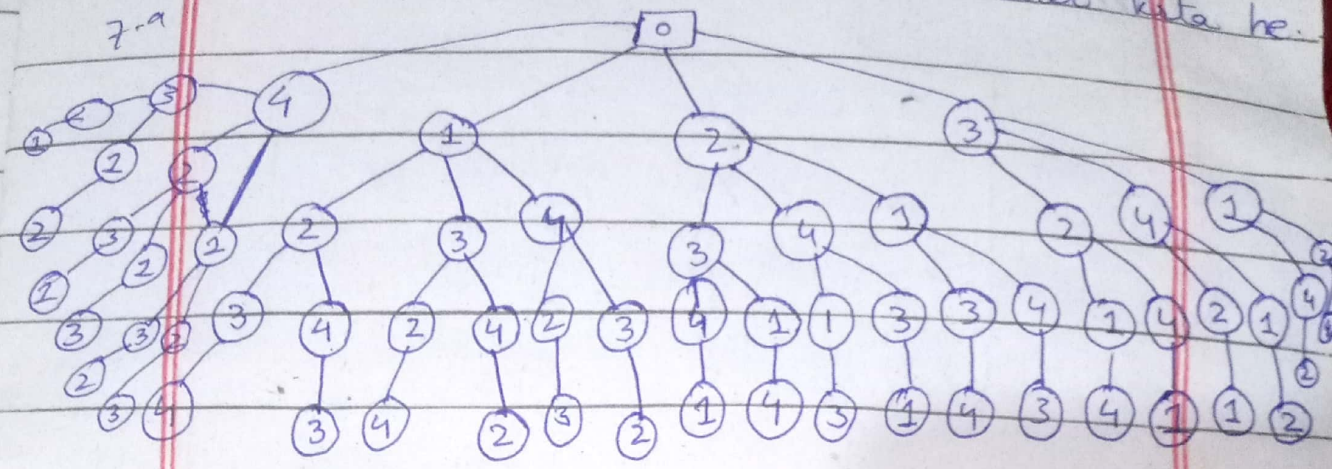


Sol. not exist for d = 11

DATE: DAY:

→ Permutation mein order matter karta he.

7-a



30-04-25

Branch & Bound

↳ objective function

→ Best fit

Live Nods

maximization → Upper bound.

minimization → Lower bound

Assignment Problem:

→ Cost minimization ⇒ objective

* Optimization problem

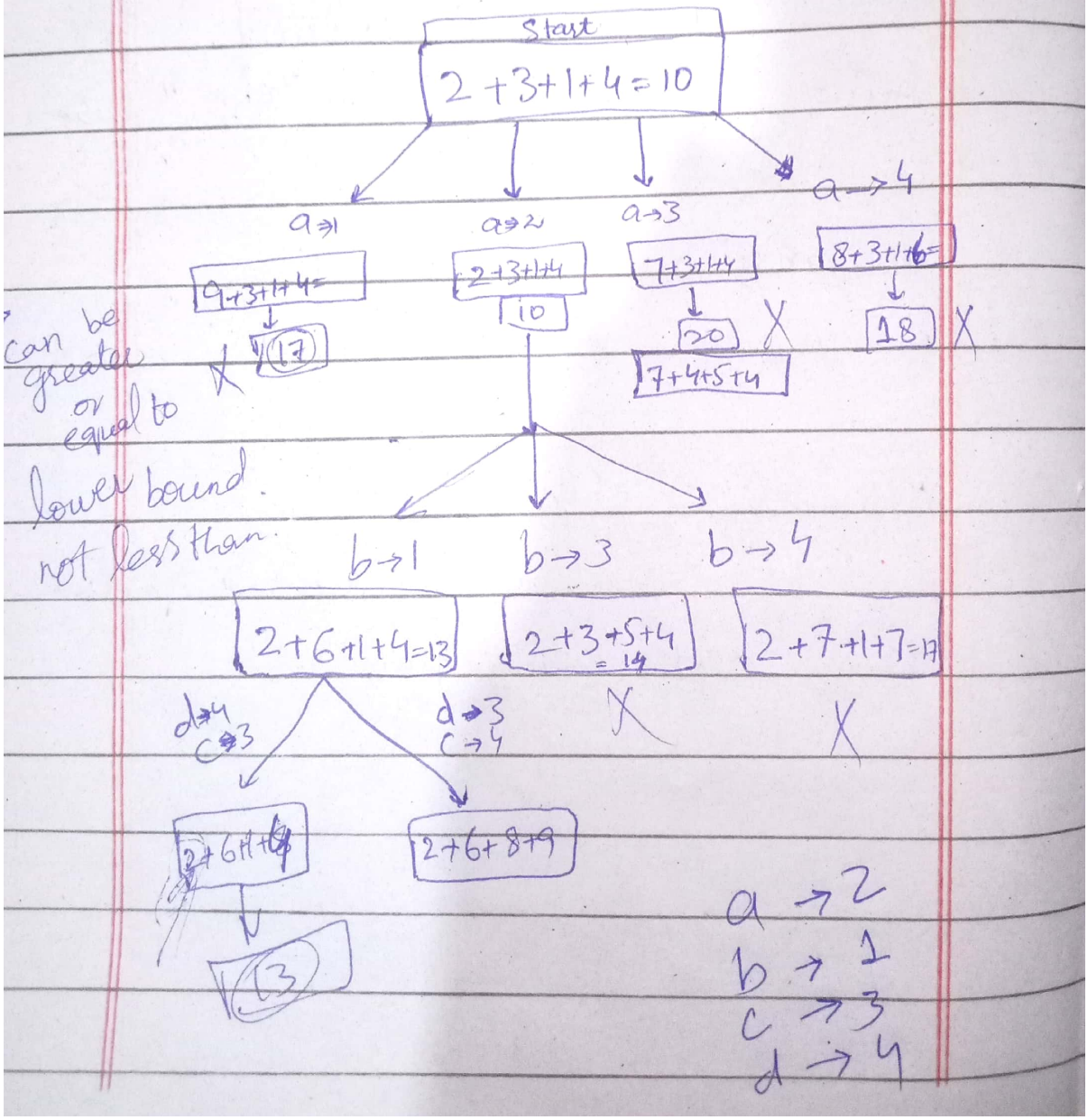
DATE: DAY:

Assignment Problem:

	Job →		
Person ↓	9	2	7	8
	6	4	3	7
	5	8	1	8
	7	6	9	4

Row wise

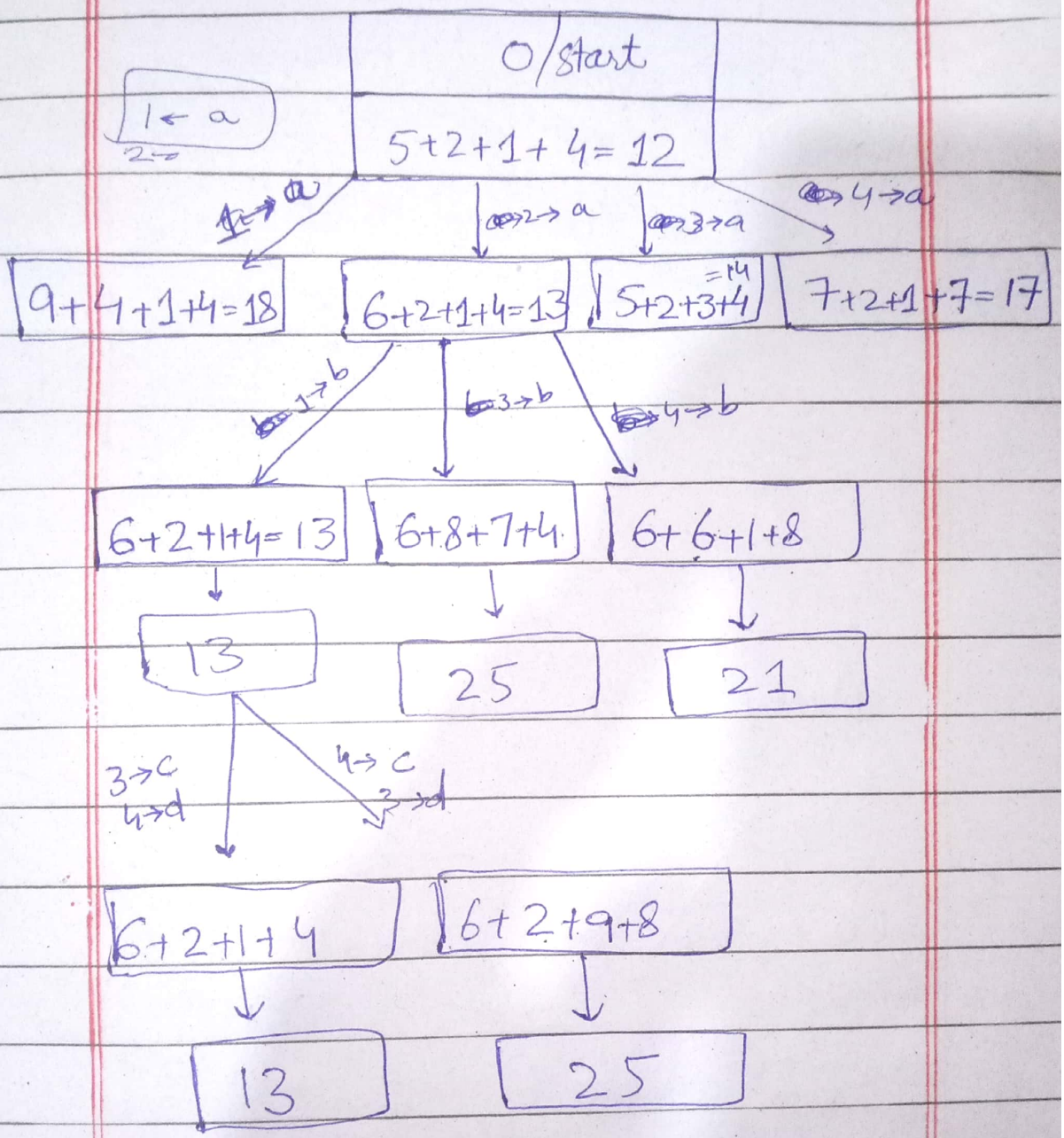
$2 + 3 + 1 + 4 = 10$ → lower bound



DATE: Column Wise DAY:

Lower bound:

$5+2+1++4 = 12$



a = persons → jobs

1 → b

2 → a

3 → c

4 → d

DATE: Knapsack Problem

Cost Maximization
↳ Upper bound.

weight	value	value/weight
4	40	10
7	42	6
5	25	5
3	12	4

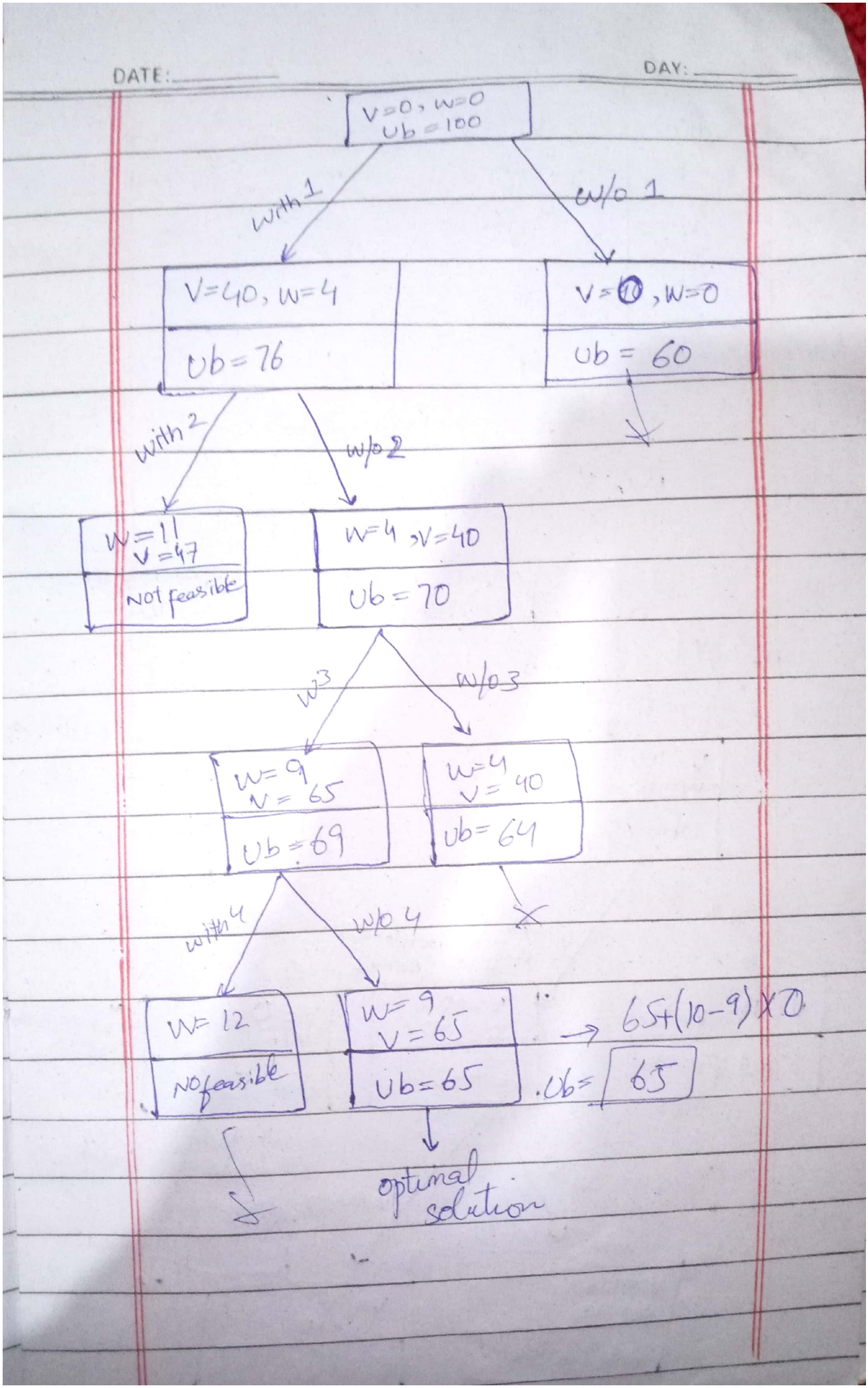
Knapsack's capacity = 10

fractional → greedy → value/weight
like energy.

$Ub = v + (W - w)(v_{i+1}/w_{i+1})$
↳ next item fraction

v → value of item
W → Size of knapsack
w → weight of item.

DATE: DAY:

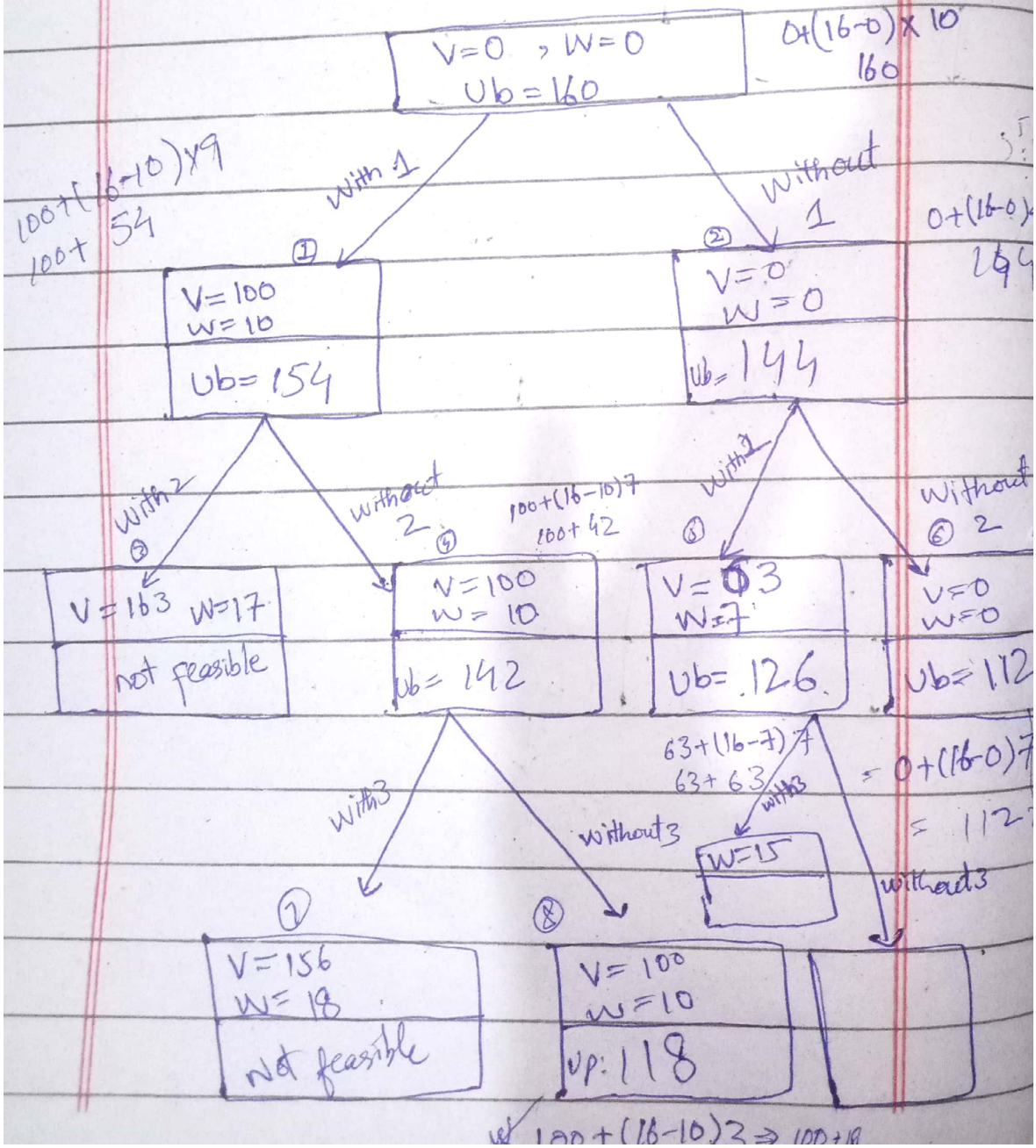

V=0, w=0
Ub = 100
with 1
w/o 1
V=40, w=4
Ub=76
V=0, w=0
Ub = 60
with 2
w/o 2
w=11
v=47
Not feasible
w=4, v=40
Ub = 70
w/ 3
w/o 3
w=9
v=65
Ub=69
w=4
v=40
Ub=64
with 4
w/o 4
w=12
Nofeasible
w=9
v=65
Ub=65
→ 65+(10-9)x0
Ub= 65
optimal solution

DATE: DAY:

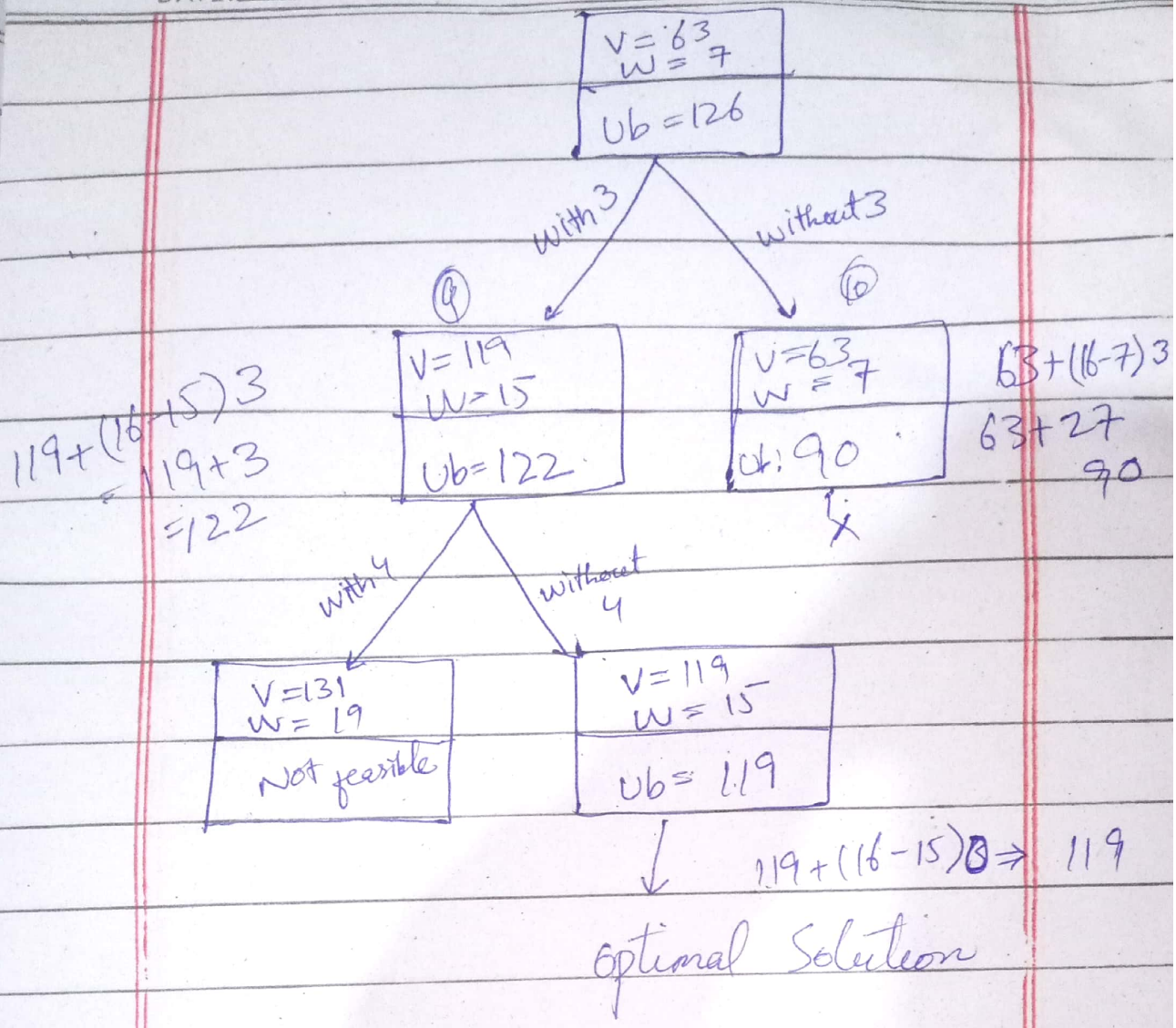
Solve this Problem.

item	weight	value	value/weight
1	10	100	10
2	7	63	9
3	8	56	7
4	4	12	3

W = 16



DATE: DAY:



without 1, with 2, with 3, without 4

weights 7+8 → 15

Values: 2, 3 → 63+56 ⇒ 119

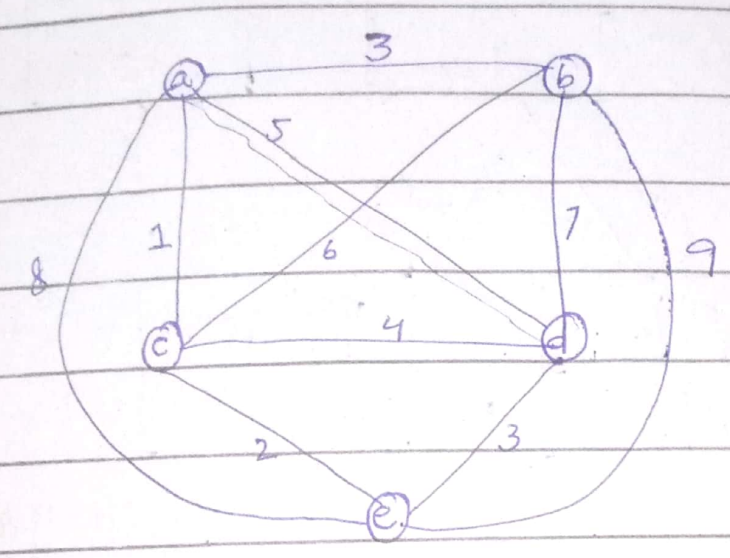
DATE: 02-05-2025 DAY: Friday

Branch and Bound

Travelling Salesperson Problem:

objective → cost minimization

↓ Lower bound



a → 1, 3, 5, 8

b → 3, 6, 7, 9

c → 1, 2, 4, 6

d → 3, 4, 5, 7

e → 2, 3, 8, 9

$lb \rightarrow \frac{(1+3)+(3+6)+(1+2)+(3+4)+(2+3)}{2}$

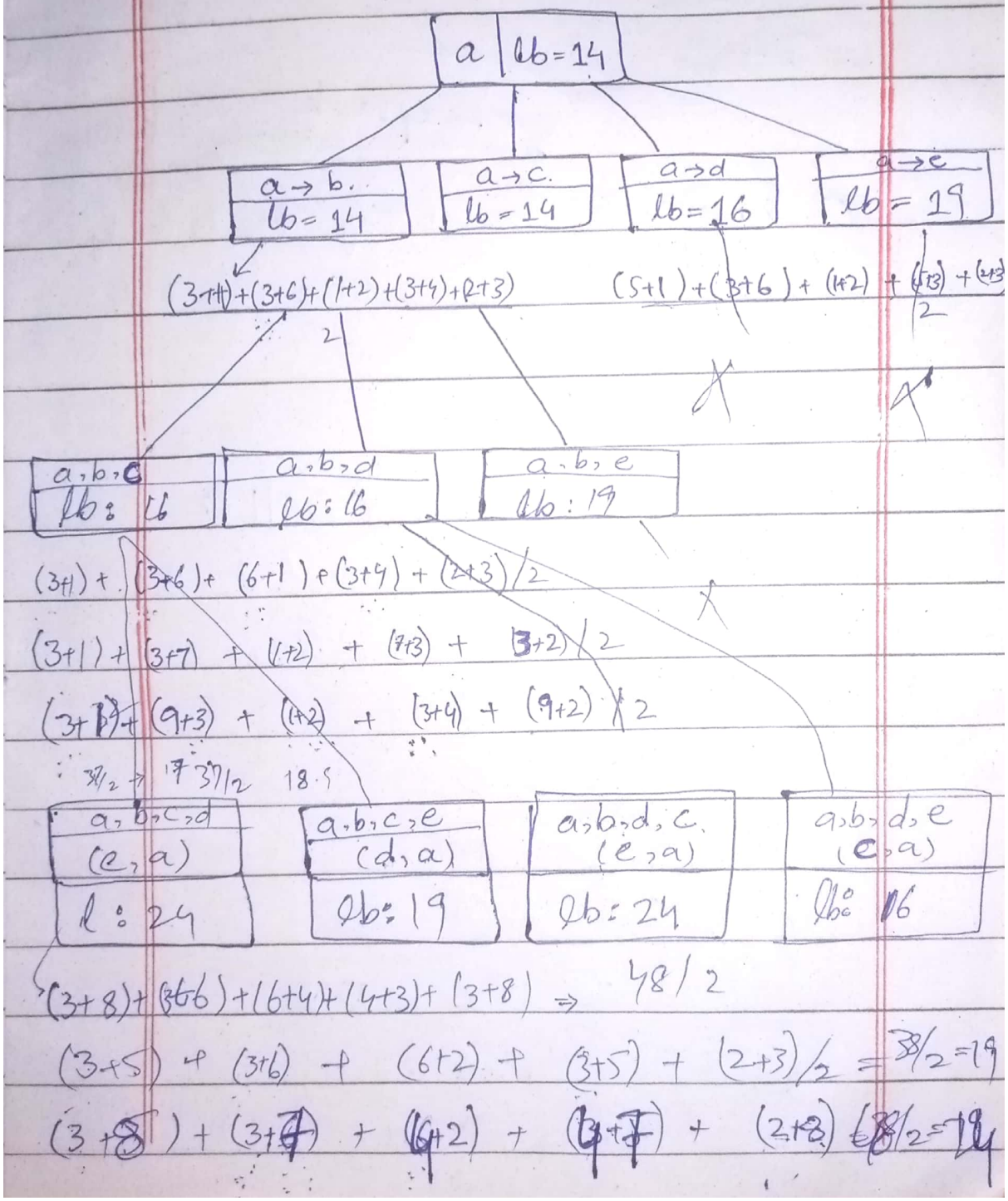
$lb \rightarrow 28/2 = 14$

DATE: DAY:

For constraint:

$$lb = \frac{(1+5)+(3+6)+(1+2)+(3+5)+(2+3)}{2}$$

$$= 31/2$$



DATE: DAY:

(3+1) + (3+7) + (2+1) + (7+3) + (3+2)

32/2 = 16

Greedy Technique

→ greedy choice (makes its first choice first/before solving any subproblem.)

* Huffman/ knapsack
* Kruskal
* prims

* knapsack assignment

01 knapsack → greedy approach se solve nae hota.

encoding

00 → a

01 → l

10 → i

11

00011 0

a l i

6

	freq	
a	2	0
l	1	1
i	1	01

0 1 0 1 0

decode a l a l a

a l i a

(3+2)

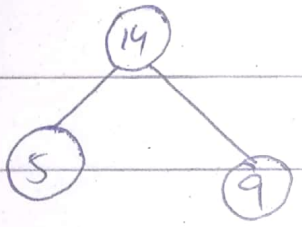
Prefix Codes

Huffman Encoding runtime (nlgn)

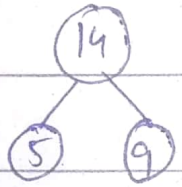
	a	b	c	d	e	f
freq	45	13	12	16	9	5

↗ higher order arrange

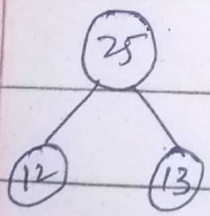
f	e	c	b	d	a
5	9	12	13	16	45

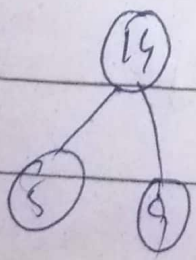


12 13

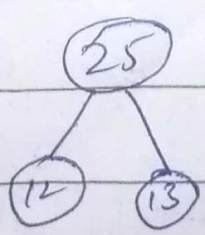


16 45



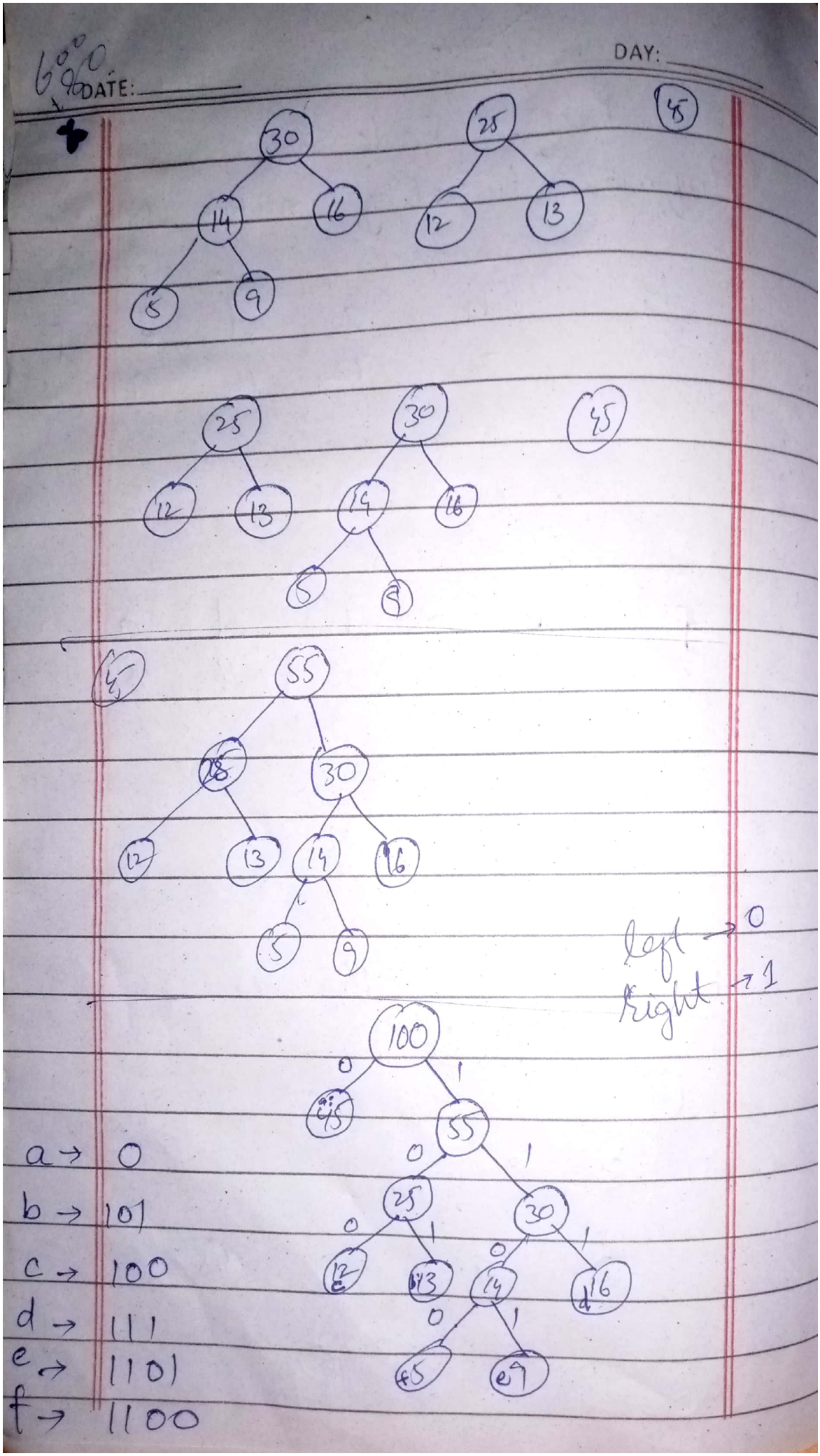


16



45

DATE: DAY:



a → 0

b → 101

c → 100

d → 111

e → 1101

f → 1100

DATE: DAY:

Fade

→ 1100 0 111101

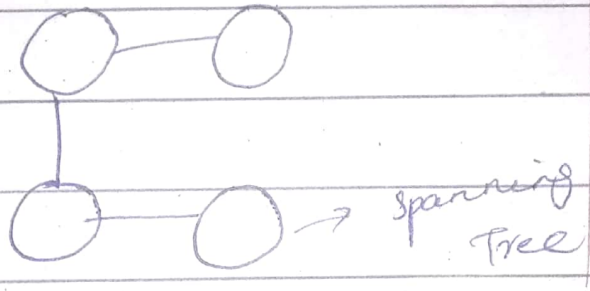
fade

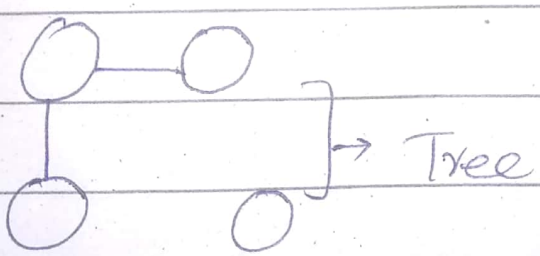
Spaning Tree: → non-optimization

→ all nodes should be reachable.

→ connected nodes.

→ NO cycle should be there





DFS, BFS.

ager minimum cost nikalni ho tu ↓ optimization

Graph by minimum cost only spanning tree nikalty hain: by

→ Kruskal

→ Prims

DATE: 14-05-25 Dynamic Programming DAY:

→ Coin-row problem: (based on recursive)

→ Maximization

→ Constraints → neighbors couldn't be picked up

Fibonacci Numbers:

Bottom-up / Tabular

$F(n) = 0 \quad , n = 0$

$f(n) = c_1 \quad , \quad n = 1$

$f(n) = \max [c_n + f(n-2) \; , \; f(n-1)]$

[5, 1, 2, 10, 6, 2]

n	0	1	2	3	4	5	6
c ~~f(n)~~		5	1	2	10	6	2
f(n)	0	5	5	7	15	15	17

When n=2 means 2 Coins

f[1]

$f(n=2) = \max [1 + f[0], 5]$

$= \max [1, 5]$

When n=3 f[1] f[2]

$f(n=3) = \max [2+5, 5]$

$= \max [7, 5]$

DAY:

DATE:

When $n=4$

$$f(n=4) = \max[10 + f(2), f(3)]$$

$$= \max[10 + 5, 7]$$

$$= 15$$

When $n=5$

$$f(n=5) = \max[6 + f(3), f(4)]$$

$$= \max[6+7, 15]$$

$$= \max[13, 15]$$

$$= 15$$

When $n=6$.

$$f(n=6) = \max[2 + f(4), f(5)]$$

$$= \max[2+15, 15]$$

$$= \max[17, 15]$$

$$= 17$$

For coins number backtrack the problem

C_1, C_4, C_6

DATE: DAY:

Coin Denomination Problem:

→ Minimization

→ Bottom-Up / Tabular

amount $n = 6$

coins denomination $\Rightarrow$ (1), 3, 4

$f(n)$ = no. of coins to give certain change.

0	1	2	3	4	5	6
0	1	2	1	1	2	2

$n - d_j$

$$f(n) = \min\{f[n-d_j]\} + 1$$

$$f(1) = \min\{f(1-1)\} + 1 = \min[f(0)] + 1$$
$$= 0 + 1 = 1$$

$$f(2) = \min[f(2-1)] + 1 = \min[f(1)] + 1$$
$$= \min[1] + 1 \Rightarrow 2$$

$$f(3) = \min[f(3-1), f(3-3)] + 1 = \min[f(2), f(0)] + 1$$
$$= \min[2, 0] + 1 \Rightarrow 0 + 1 = 1$$

DATE: DAY:

$f(4) = \min[f(4-1), f(4-3), f(4-4)] + 1$

$= \min[f(3), f(1), f(0)] + 1$

$= \min(1 + 1 + 0) + 1$

$= 0 + 1 = 1$

$f(5) = \min[f(5-1), f[5-3], f[5-4]] + 1$

$= \min[f(4), f(2), f(1)] + 1$

$= \min(1, 2, 1) + 1$

$= 1 + 1$

$= 2$

$f(6) = \min[f(6-1), f(6-3), f(6-4)] + 1$

$= \min(f(5), f(3), f(2)] + 1$

$= \min(2, 1, 2) + 1$

$= 1 + 1$

$= 2$

For Coins → Backtrack.