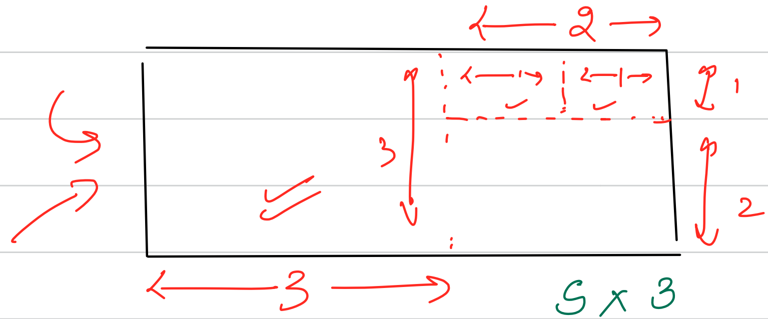
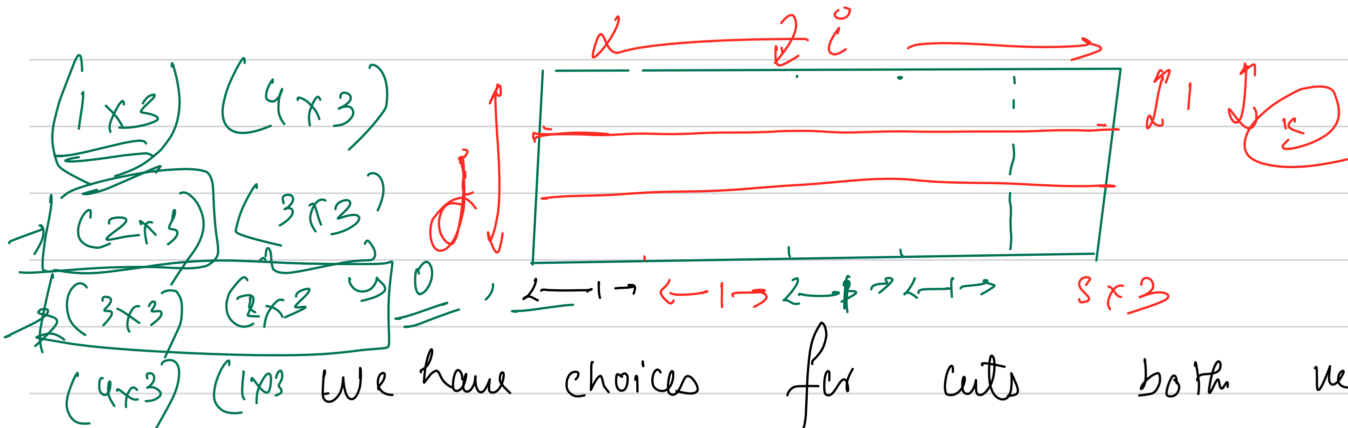
Q→ ~~Rectangle Cutty~~



Try all possible cases



We have choices for cuts both vertically & horizontally

We can add the cuts both horizontally & vertically

$$f(i, j) = \min \begin{cases} \min\left(f(i,j),\; f(i,k) + f(i, j-k)\right) & \forall k \in [1, j-1] \quad \text{cut horizontally} \\ \min\left(f(i,j),\; f(k,j) + f(i-k, j)\right) & \forall k \in [1, i-1] \quad \text{cut vertically} \end{cases}$$

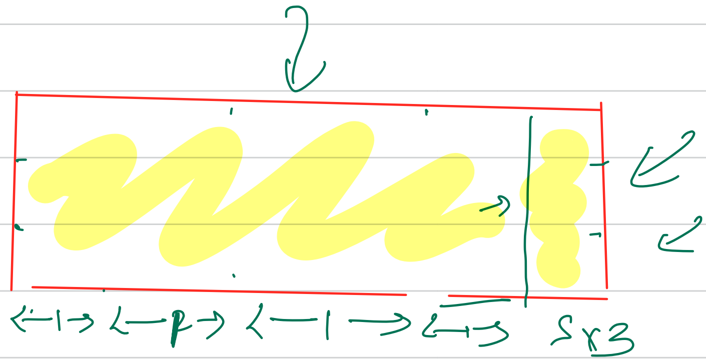
$(i \times j)$

1x3

5x3 → 4x3

3x2 1x3

ans = 9 ~~7~~ 3



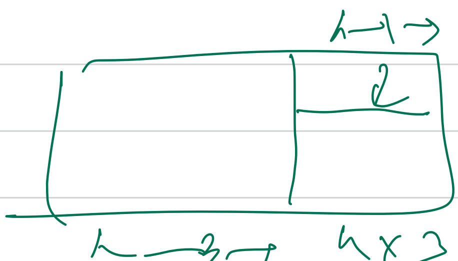
1 + 5 + 3

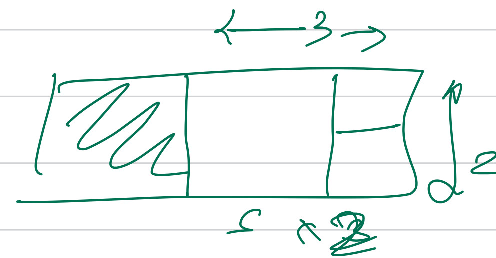
3 + 3 + 1

0 + 2 + 1

4x3

3





Q:- Money Sums

2 2

4 2 5 2

(4, 6, 2) 9, 11, 7, 5

4, 6, 2, 9, 11, 7, 5, 8, 13

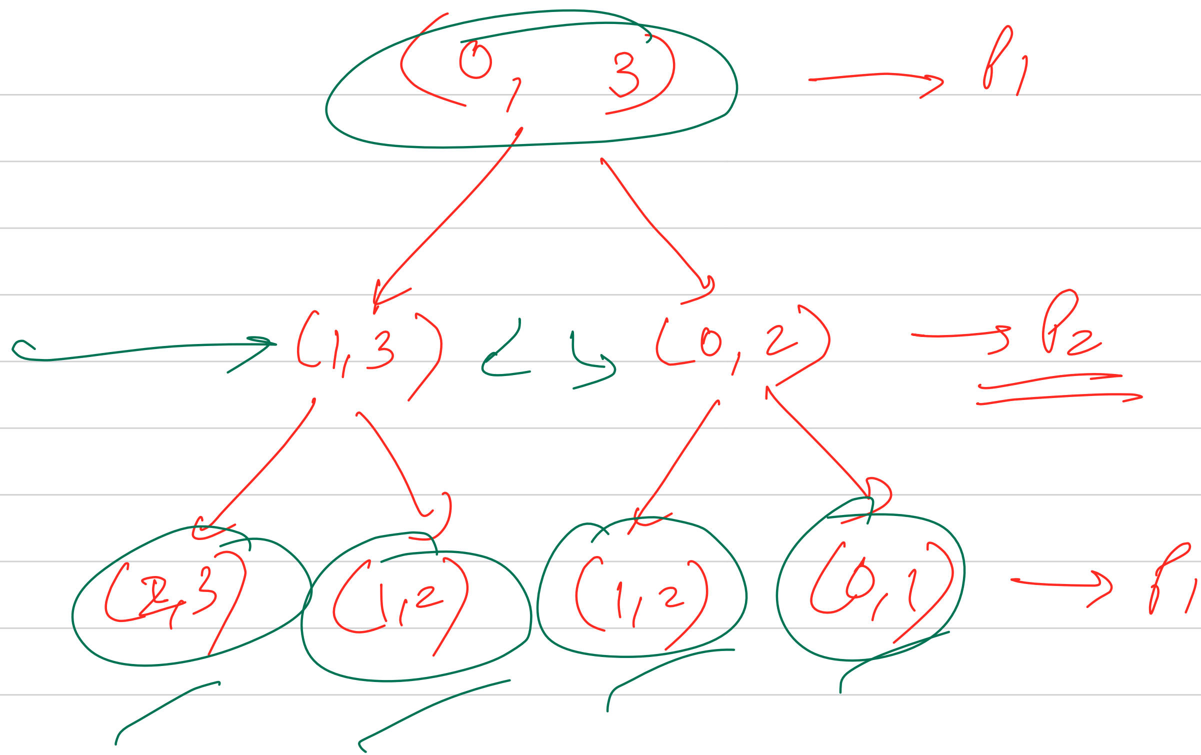
9

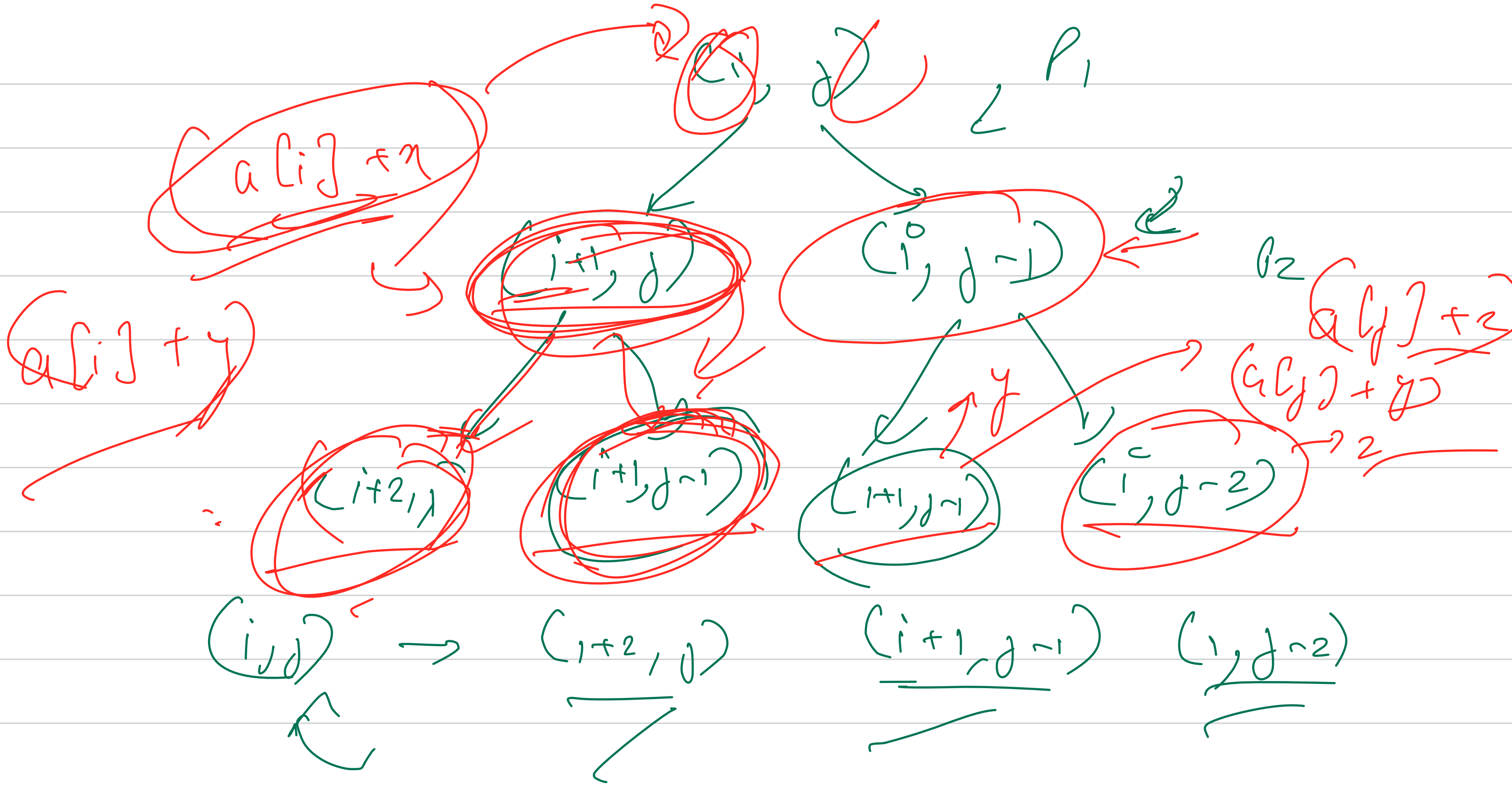
~~4~~, 5, 1, 3 $\underline{\underline{P_1}}$

[5, 1, 3] [4, 5, 1] $\underline{\underline{P_2}}$

(0, 3)

(1, 3) (0, 2)

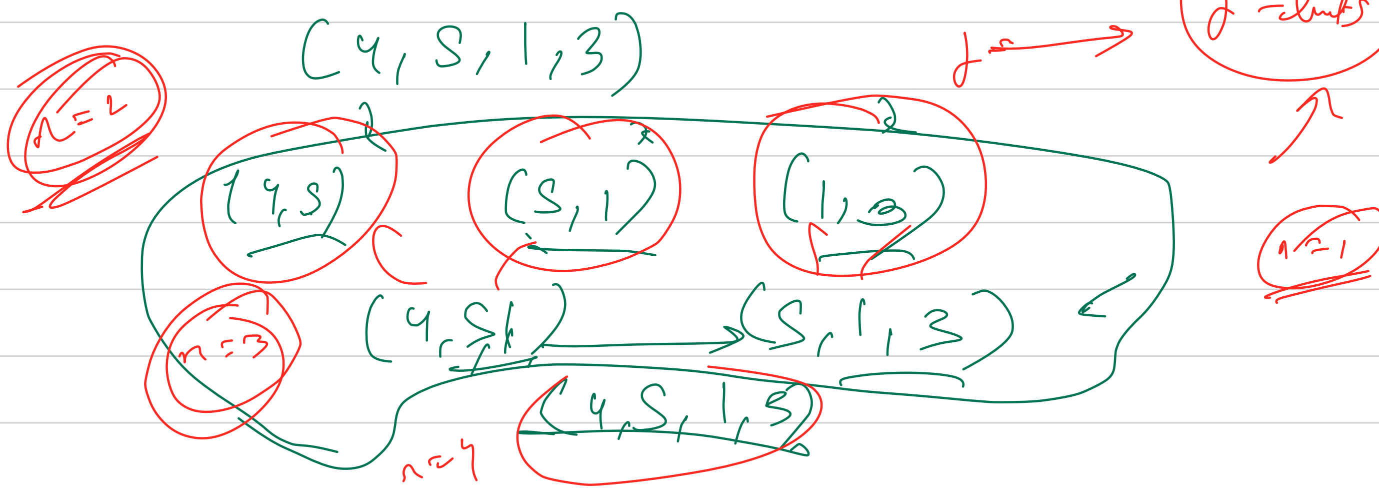

(0,3)
P1
(1,3)
(0,2)
P2
(2,3)
(1,2)
(1,2)
(0,1)
P1



(i, j)
P1
a[i] + x
(i+1, j)
(i, j-1)
P2
a[j] + z
a[i] + y
a[j] + y
(i+2, j)
(i+1, j-1)
(i+1, j-1)
(i, j-2)
(i, j)
(i+2, j)
(i+1, j-1)
(i, j-2)

$$f(i,j) = \max\Big(\min\big(a[i] + f(i+2,j),\ a[i] + f(i+1,j-1)\big),\ \min\big(a[j] + f(i,j-2),\ a[j] + f(i+1,j-1)\big)\Big)$$

j → j -elements



{1, 2, 3, 4, 5} => 15

$\sum_{i=1}^{n} i \rightarrow$ is odd ~~X~~

sum/2

→ ans → Sum of elements is equal to sum/2

of subset

Knapsack

$$f(i, j) = f(i-1, j) + f(i-1, j-i)$$

$j \to$ sum

$i \to$ no. of elements obser

count no. ways

every elmt has 2 choice

add

avoid

1 2 2 2 2 2 1 2

7, 3, 5, 3, 6, 2, 9, 8

dp $f(i)$ = length of longest inc subseq ending at i

7	3	5	3	6	2	9	8	
1	1	2	1	3	1	4	4	ans

7, 9

3, 9

3, 6

3, 5, 6 → 3

3, 6

3, 5, 9

3, 9

3, 5, 6, 9

2, 9

4

for every elmts, iterate from $0 \to i-1$

$f(i) = 1 + f(j) \quad \forall j \in [0, i-1]$ &&

$a[j] < a[i]$

$LIS(0, n-1) = \max f(i) \quad \forall i \in [0, n-1]$

TC $\rightarrow O(n^2) \rightarrow$ Binary Search

Segment Tree

$O(n \log_2 n)$

→ Sort the array based endTime

→ for any i^{th} job → we can either pick it
or leave it

pick → check all $(0 - i-1)$ jobs

endTime < startTime

2 4 4
3 6 6
6 8 2
5 7 3

2 4 4 → 4
3 6 6 → 4, 6 → 6
5 7 3 → 6, 6 → 8
6 8 2 → 8, 6 → 8

Binary Search

Unsorted

J1 2 4 4 → 4
J2 3 6 6 → 6
J3 5 7 3 → 7
J4 6 8 2 → 7, 6 → 7

n log n

n^2