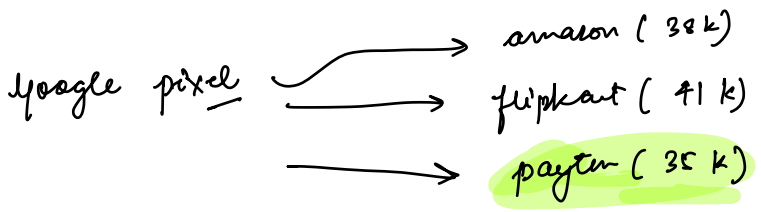


20 books ⟶ 10 books

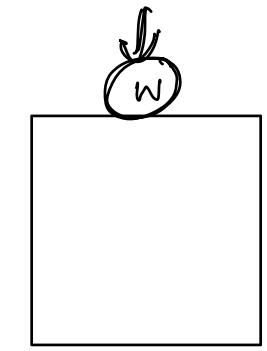
thinner

LLD 10 books

ropes ⟶ minimise cost

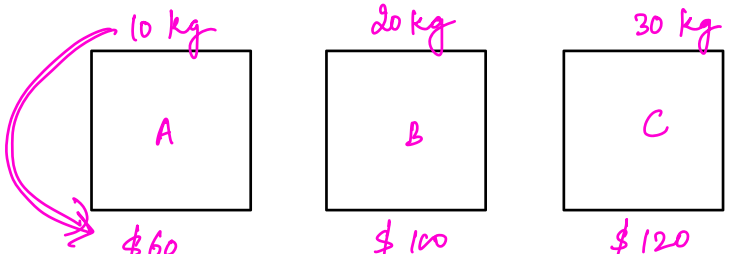
array a[i]+a[j] two largest values

1) Fractional Knapsack



N items → weight
→ value

fractionally



maximise total value

W = 50

$60 for total 10 kg

$60/10 = $6/kg

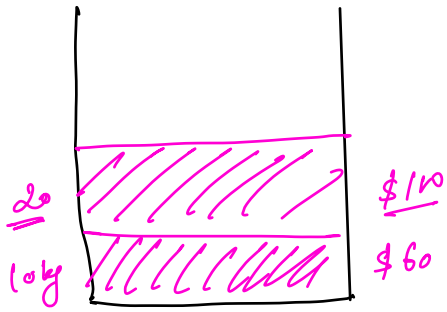
$100 → $5/kg

$120 → $4/kg

A	10 kg	$60
B	20 kg	$100
C	20 kg	$80
		$240

0/1 Knapsack. ⇒ either take it completely or leave it

can't be solved by greedy approach



Job scheduling

N tasks/jobs → reward, deadline

one task can be done only once

A job will take only one complete day.

you can't do 2 jobs simult.

you can do a job either on the deadline day or before it

maximise reward

Tasks	Deadline	reward
✓ A	3	100
B	1	19
✓ C	2	27
D	1	25
✓ E	3	30

100, 27, 25

100 A, 27 C, 30 E

B.F Try all possible diff combinations ≤ 2^n ($^nC_{max\ no.\ of\ days}$ ✗ max)

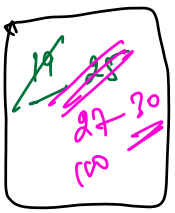
maximise reward + maintain deadlines

sort the data acc to deadline

day=0 day=1 2 3

1 1 2 3 3

19 25 27 100 30

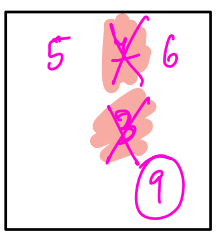


100+27+30

deadline	3	1	3	2	3
rewards	6	5	3	1	9

	1	2	3	3	3
	5	1	6	3	9

day=0 1 2 3

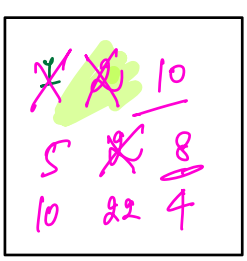


minheap

deadline	1	1	3	3	3	4	6	6	6
reward	1	2	10	2	22	4	10	5	8

day=0 1 2 3 4 5 6

ans = 59



minheap

1) sort the data acc to deadline

day = 0

for(i = 0 ⟶ n)

{

if(deadline[i] > day)

{ day++;

insert (reward[i]) in minheap

}

else {

if(root of minheap < reward[i])

{

extract min()

insert in minheap

}

}

}

T.C: nlogn

+ nlog(max deadline)

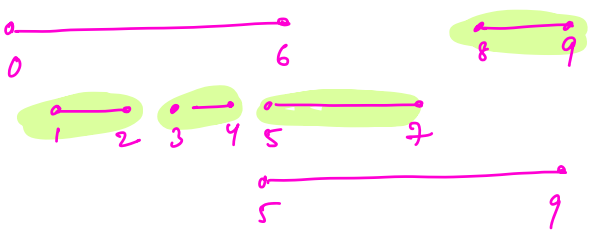
Activity selection problem :

N activities → start end

1 activity at a time

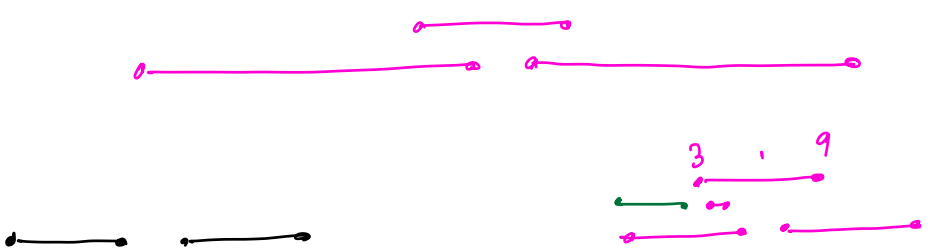
{ maximum no of activities you can perform }

	a_1	a_2	a_3	a_4	a_5	a_6
start	1	3	0	8	5	5
end	2	4	6	9	7	9



(1) sort acc to start time ✗ greedy

(2) sort acc to duration ✗



start	8	12	1	3	4	5	13
end	11	14	5	4	5	6	19

1) sort acc to end

3	1	4	5	8	12	13
4	5	5	6	11	14	19

2) Traverse while maintaining the cond.

Distribute candies

N students — marks[i]

minimise the chocolates

student → left neighb → higher no of candies
↳ right neigh → higher no of ca.

Atleast one chocolate to every studs