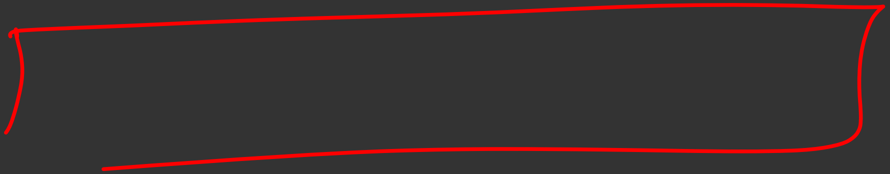
Greedy Algorithms and Techniques

- Priyansh Agarwal

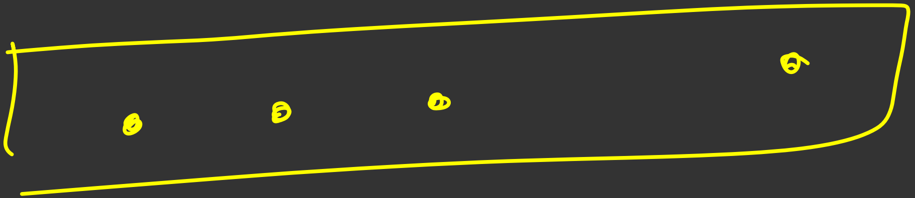
Array of integers (+ve, -ve, 0)

Select a subset whose sum is maximum



$O(\boxed{2^n})$

$n \to 10^5$



take all positive numbers

O(n)

pick up a subset whose sum is max, if there are more than 1 such subset, pick the subset with max no of elements

(+ve, -ve, zeros)

DP vs Greedy

Problem 1:

Proving → Informal

Proving → Formal → Exchange Argum

Given an array of integers find the minimum absolute difference between any two elements of the array. (1 <= N <= 1e5)

$(1 \le a[i] \le 1e9)$

Guess Work ✗

10	9	20	3	5	1

$O(n^2)$

Ans = 1

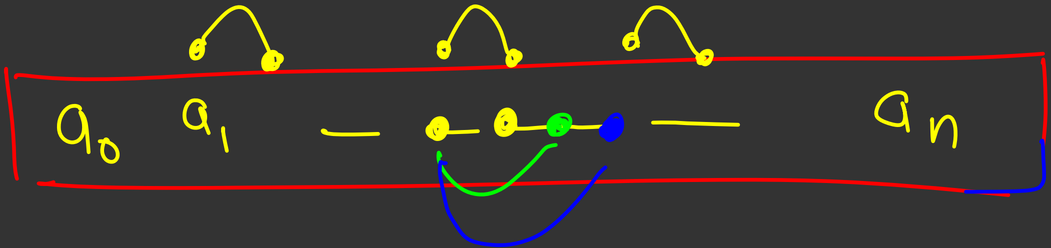
Solution : Sort and consider all adjacent pairs

→ n!

time complexity

→ n log n → sorting

n → iterating



Sort the array, consider all adjacent pairs

$[a_i \geq a_{i-1}]$ (a_5, a_6)

$a_7 \geq a_6 \geq a_5$ (a_5, a_7) #

$\longrightarrow (a_7 - a_5) \geq (a_6 - a_5)$

formal

Claim ✓ a_i a_j

if $j > i+1$ then (a_i , a_{i+1}) will have a smaller differenu than (a_i , a_j)

$(\quad \leq)$

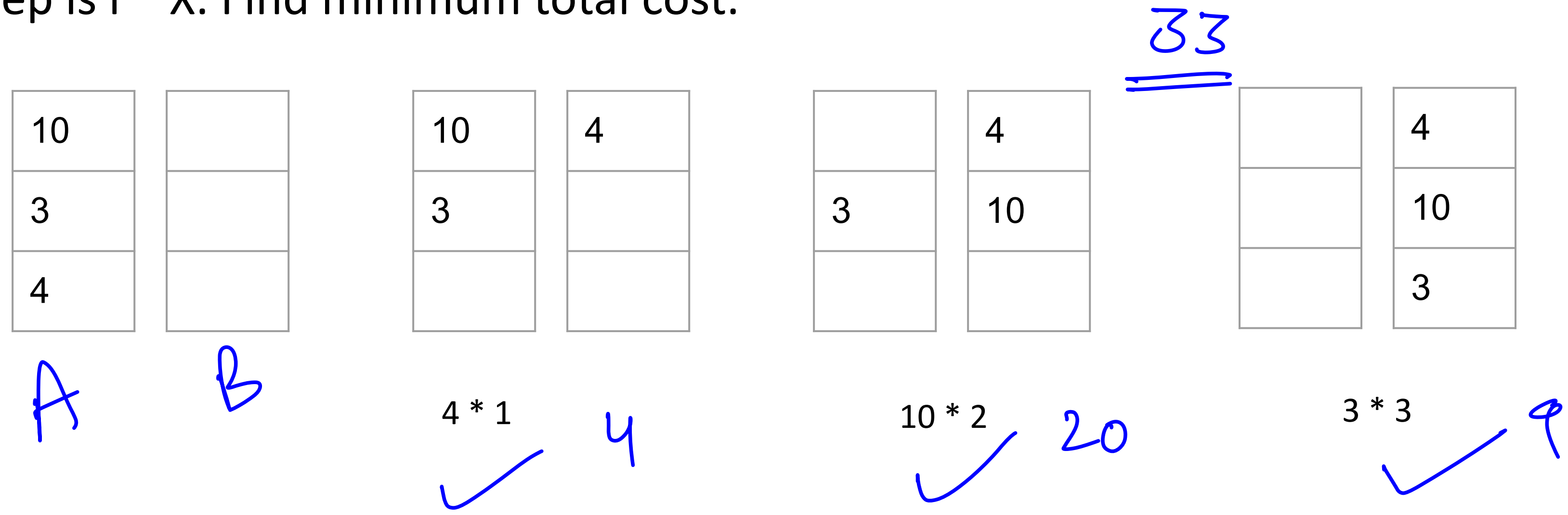
Proof: ✓

$$a_j \geq a_{j-1} \geq a_{j-2} - - - a_{i+1} \geq a_i$$

$$(a_{j+1} - a_i) \leq (a_j - a_i)$$

Problem 2:

Given an array of integers, you want to transfer every element from it to another array (initially empty). The cost of moving an element X on the i^{th} step is i * X. Find minimum total cost.



Sort array A in decreasing order and start moving elements from largest to smallest

$a_1 \geq a_2 \dots a_n$

$\hookrightarrow \boxed{a_1 \; a_2 \; - \; - \; - \; a_n}$

$\rightarrow [a_1 \cdot 1, \; a_2 \cdot 2, \; a_3 \cdot 3 \; - - - \; a_n \cdot n]$

$[(a_1 \geq a_2 \; - \; a_{i-1}) \;|\; [\geq a_n] = (d_1 - - d_{i-1})$

$$\boxed{a_1 \quad a_2 \quad - - - \quad a_n}$$ ✓

$$a_1 \cdot 1 + a_2 \cdot 2 \; - - - - \; a_n \cdot n$$

↓

minimum cost.

$$(a_1 \geq a_2 - - - \geq a_n)$$

$G \longrightarrow \boxed{a_1 \; - - \; a_i \,|\, a_{i+1} \; - - \; a_n}$

$O \longrightarrow \boxed{b_1 \; - - - - \; b_i \,|}$

$a_{i+1} \geq$ rest of element remaining

$a_1 \quad a_2 \; - \; a_n$

$a_1 - a_{i-1} \quad \boxed{(a_i - a_n)}$

There exists ~~some~~ an optimal ~~order~~ $\{\delta_i\}$ sequence → O

with sum < G (current sequence)

→ Disprove

$[a_1 > a_2 \; -- \; a_n]$

G → $| a_1 \; a_2 \; -- \; | a_i \; a_{i+1} \; ---- \; a_n |$

a_{i-1}

x

O → $| \delta_1 \; \delta_2 \; -- \; | \delta_i \; \delta_{i+1} \; -- \; \delta_n |$

δ_i is the first index where $G_i \neq O_i$

$\delta_1 = a_1 \quad b_2 = a_2 \quad --- \quad \delta_i = a_{i-1}$

$a_i \neq b_i$

$b_i < a_i$

b_1	b_2	--	b_{i-1}	b_i		a_n

b_i

a_i

$b_i \quad a_i$

$b_1 \times 1 \quad b_2 \times 2 \quad -- \quad b_{i-1} \cdot (i-1)$

$+ b_j \times i + a_i \times j +$

sum1	b_i	sum_2	a_i	sum_3
	i		j	

$j > i$

S_1 ✓

$sum1 + sum2 + sum3 + (b_j \cdot j + a_j \cdot j)$

S_2

$$\left(a_i \cdot i + b_i \cdot j\right)$$

$$\text{sum1} + \text{sum2} + \text{sum3} + \qquad + $$

$$\left(\left(b_j \le a_i \;\&\&\; j > i\right)\right) \qquad S_2 < S_1$$

$$S_1 \to b_i \cdot i + a_i \cdot j \qquad S_2 < S_1$$

$$S_2 \to a_i \cdot i + b_i \cdot j$$

$$\left(S_2 - S_1\right) \qquad a_i \cdot (i-j) + b_i(j-i)$$

$$= a_i(i-j) - b_i(i-j)$$

$$A \to B \qquad = \underbrace{(a_i - b_i)}_{+ve}\underbrace{(i-j)}_{-ve} \longrightarrow -ve$$

Proving → [Your current solution = A]

→ there exists a better solution than A called O → which is the optimal solution such that O gives you a better answer than A

→ Disprove this statement

Exchange Argument

Problem 3:

Given an array of positive integers, select 3 elements from it such that they can form a triangle and perimeter of the triangle is maximized. (1 <= N <= 1e5)

10	6	2	4	3	1

1 2 3 4 6 10

Choosing 6, 4, ~~2~~ 3 will give the best biggest perimeter

(1)

$a+b > c$ ✓ $b+c > a$ ✓ $c+a > b$ ✓

$$\boxed{\text{if } (a \leq b \leq c \;\&\&\; a+b > c)}$$

$$\boxed{1 \leq a, b, c}$$

$b+c > a$

$$\underbrace{b \geq a \quad c \geq 1}_{\boxed{b+c > a}}$$

$$\underbrace{\boxed{c \geq b} \text{ and } a \geq 1}_{\boxed{c+a > b}}$$

Make a triangle by selecting 3 elements and maximize the perimeter

sort the array

$$\boxed{a_1 \quad a_2 \quad — \quad — \; — \quad a_n}$$

$$a_1 \geq a_2 \quad —— \; ———— \quad a_n$$

✓ $a_i \le a_j \le a_k$ ✓ $i < j < k$

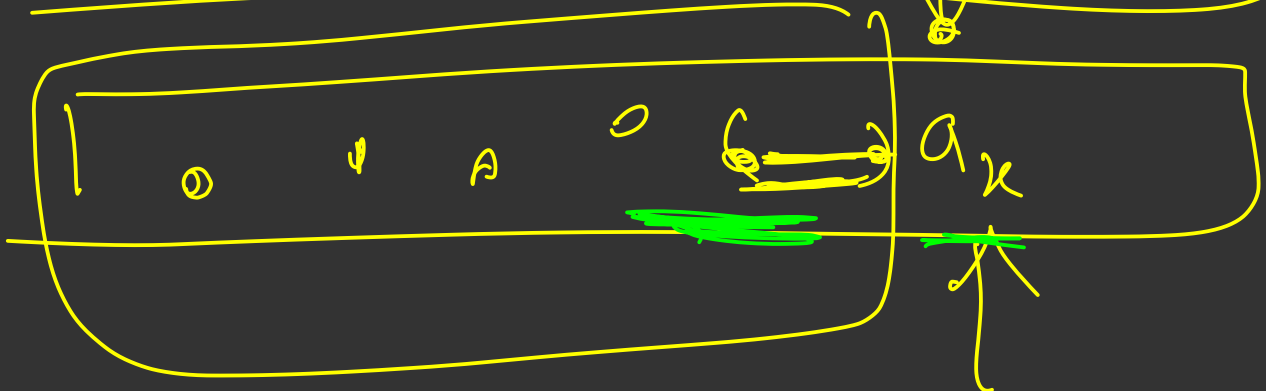
$i < j < k$

such that $a_i + a_j > a_k$

$j = k - 1$

$i = k - 2$

$a_{k-1} + a_{k-2} > a_k$ ✓



k

$k-2 \quad k-1$

All 3 consecutive elements and see the highest k for which

$$\boxed{a_{k-2} + a_{k-1} > a_k}$$

$O(n^3)$

```
for (i — (0 — n) )
    for (j — (i+1 — n)
        for (k — (j+1 — n)
```

$O(n \log n)$

$O(n)$

Problem 4: Activity Selection Link

Ig ✓ ✓

Given N event with start and end times, find the maximum number of events you can attend such that no two selected events overlap.
(1 <= N <= 1e5), (1 <= start times <= end times <= 1e9)

1 - 2	3 - 7	5 - 6	8 - 9	7 - 10	10 - 11

	10			✓ 20	
🟩 1 - 2	3 - 7	🟩 5 - 6	🟩 8 - 9	7 - 10	🟩 10 - 11
5	✓	2	3		4

Sort the events based on end

1-2
3-7
5-6
8-9
7-10
10-11

→

1 — 2
(5 - 6)
3 - 7 ✗
8 - 9
7 - 10 ✗
10 - 11

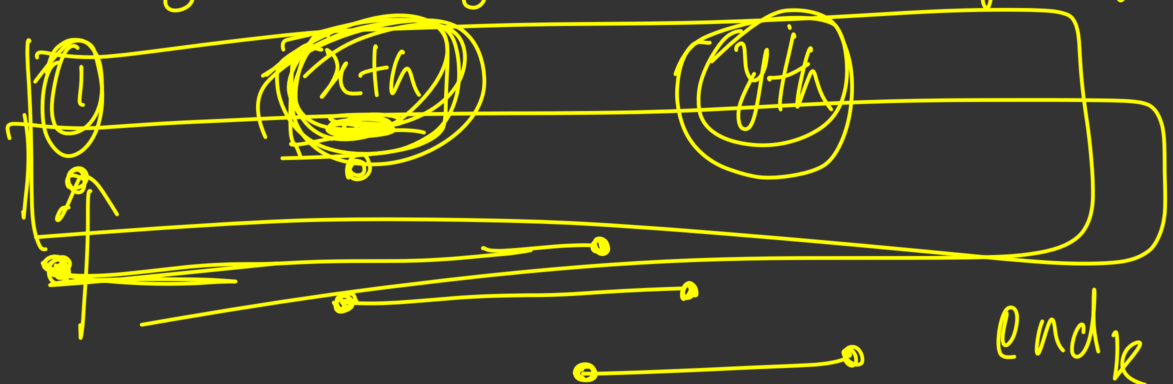
(1-2)
(5-6)
(8-9)
∅
(10-11)

4

$$start_i \le end_i$$

$$end_i \le end_{i+1} \le end_{i+1} - - \le end_n$$

1st segment you can always pick

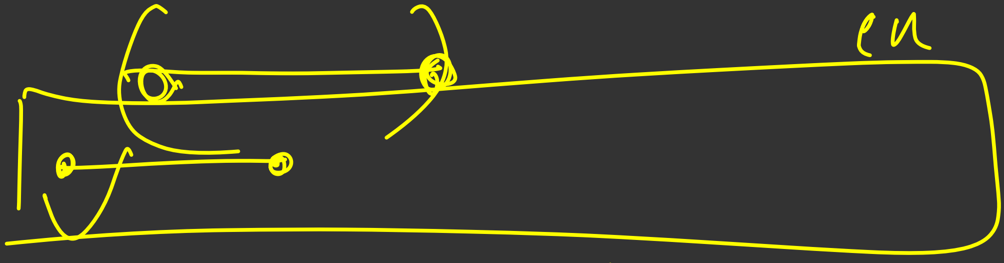


$end_k \geq end_1$

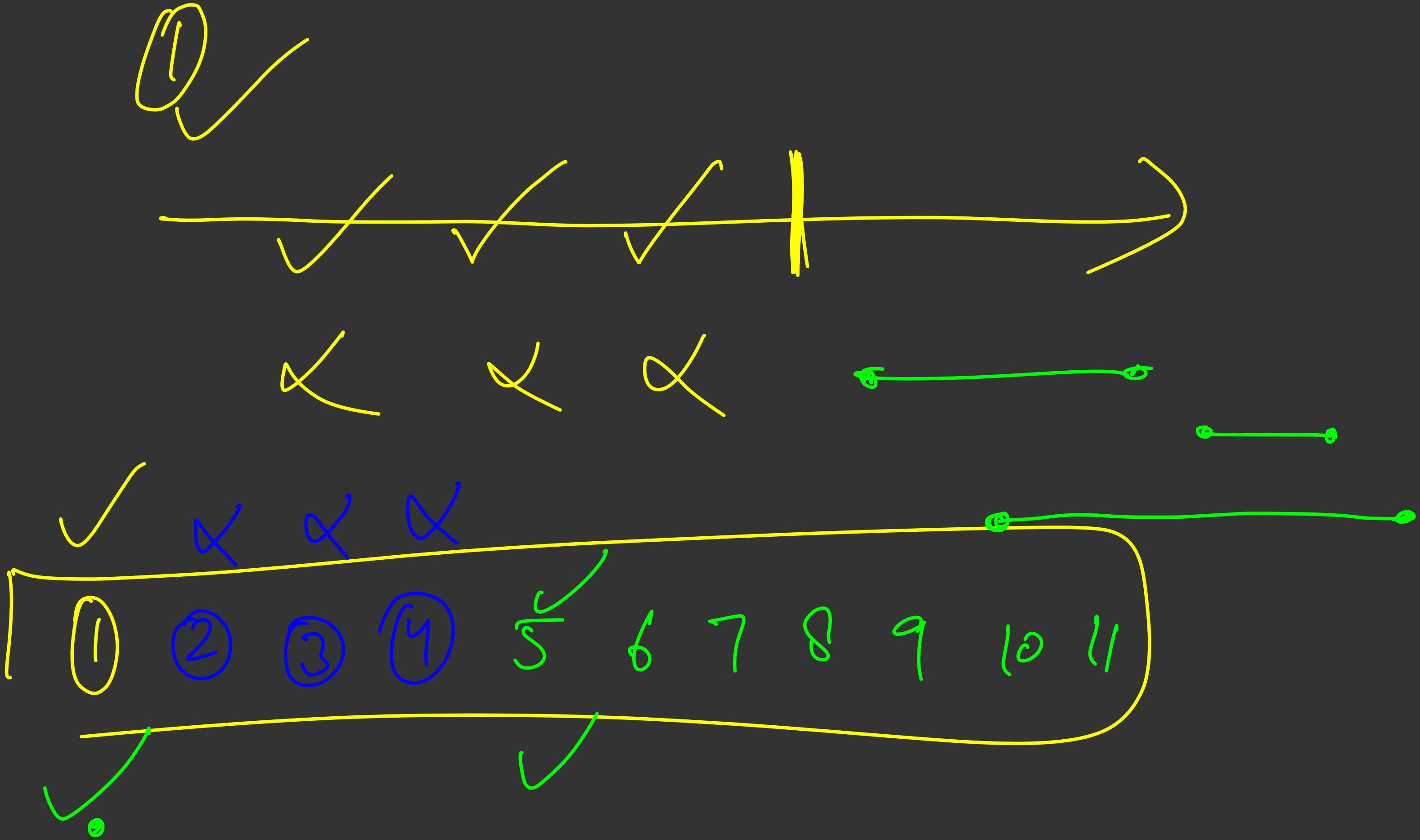
$end_k \geq end_1$

$end_1 \geq start_y$

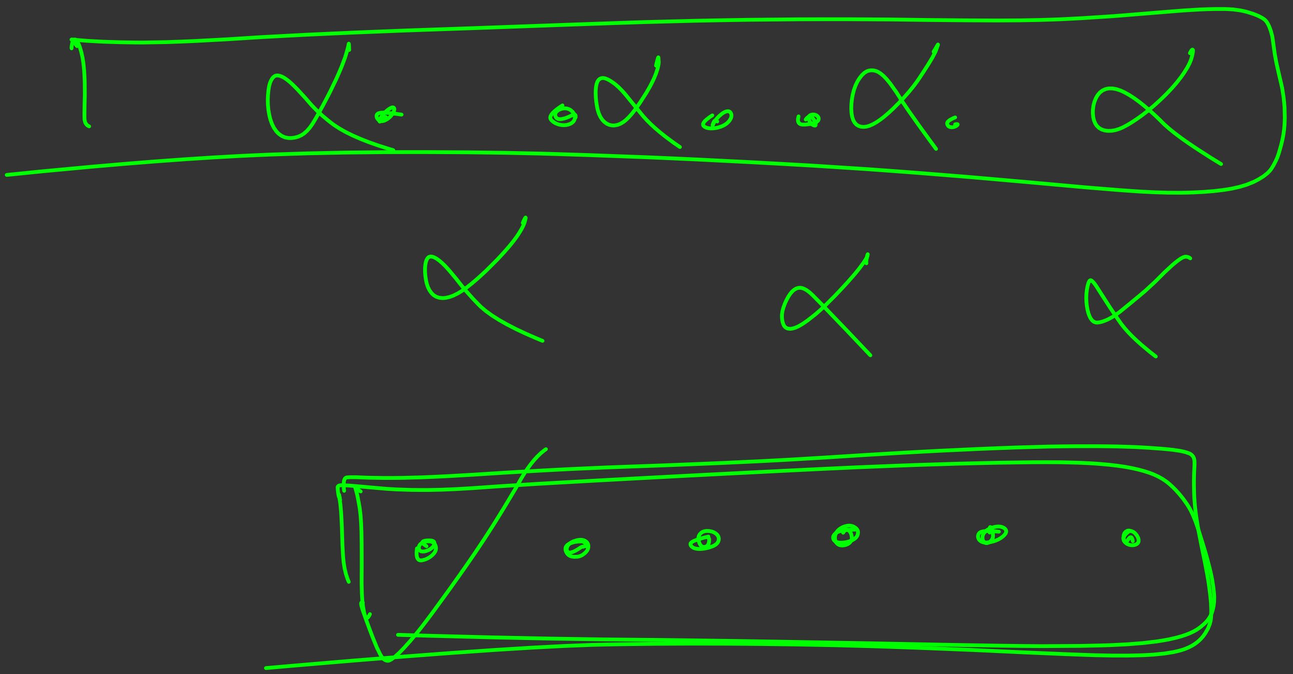
$end_k \geq start_y$



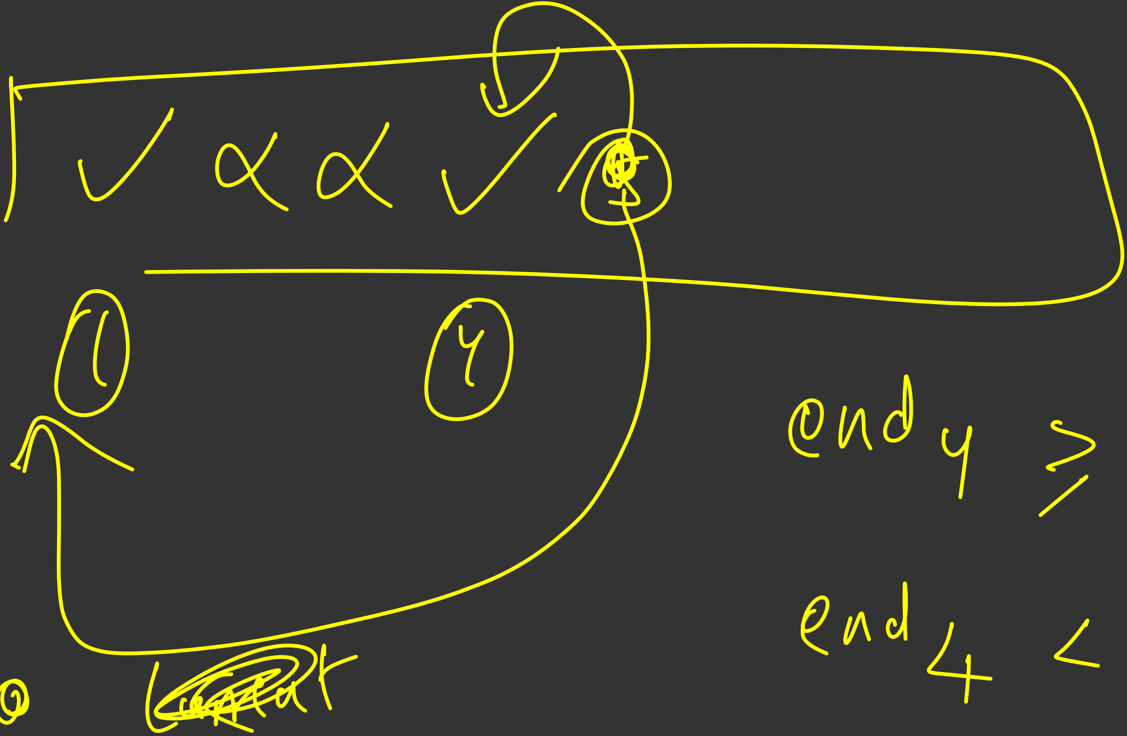
$\boxed{end_x \geq start_y}$


1
1
2
3
4
5
6
7
8
9
10
11

①



1
5
8
5
10



$end_4 \geq end_1$

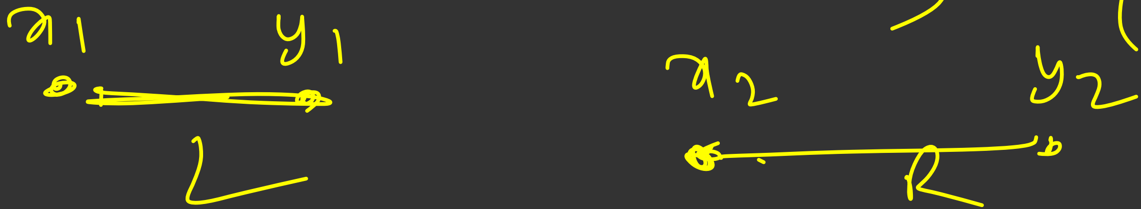
$end_4 < start_5$

Current segment -> {-Inf, -Inf}
Sort based on end point

```
for(i=0; i<n; i++)
{   if (intersecting(seg(i), current))
        continue
    ans++
    current = seg(i) }
```

$(x_2 \le y_1)$

$(x_1 \le x_2)$



Bonus: Weighted Activity Selection ✓

Given N event with start, end times and profits, find the maximum number of events you can attend such that no two selected events overlap.

(1 <= N <= 1e5), (1 <= start times <= end times <= 1e9) (1 <= profits <= 1e9)

To be covered after Dynamic Programming ✓ Greedy

Dynamic Programming + Binary Search

Custom Comparator

Google →

set <int> s

vector

~~map~~

set < int , int> s

x y

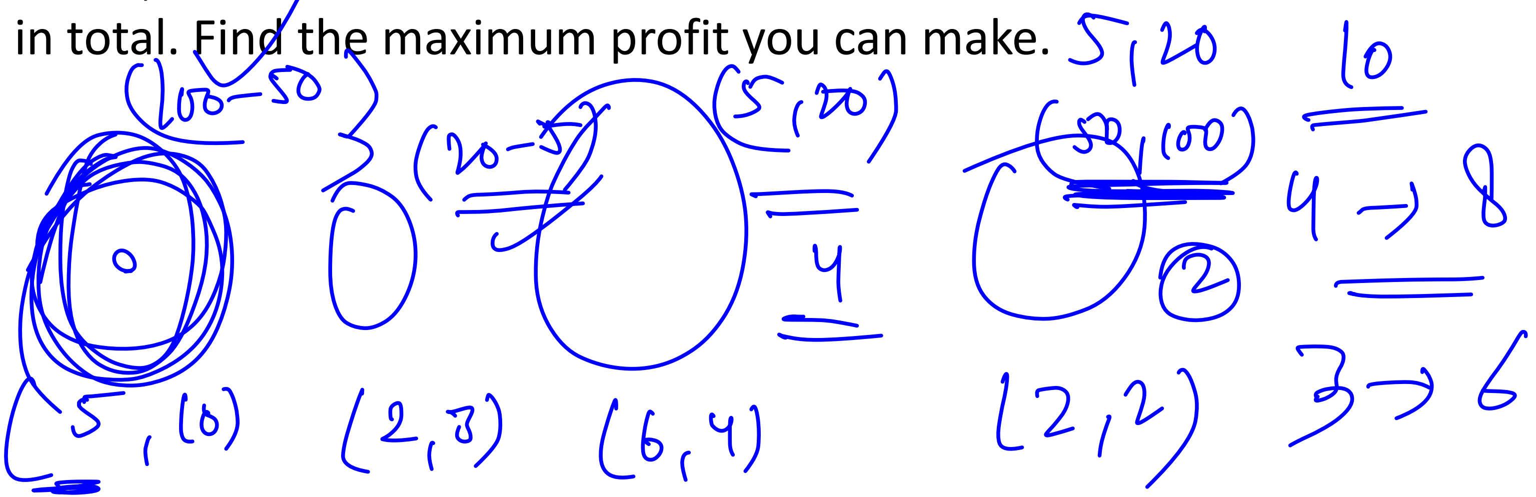
$x \neq y$

rate profit / unit

Problem 5: ~~Fractional~~ Knapsack y_i/x_i

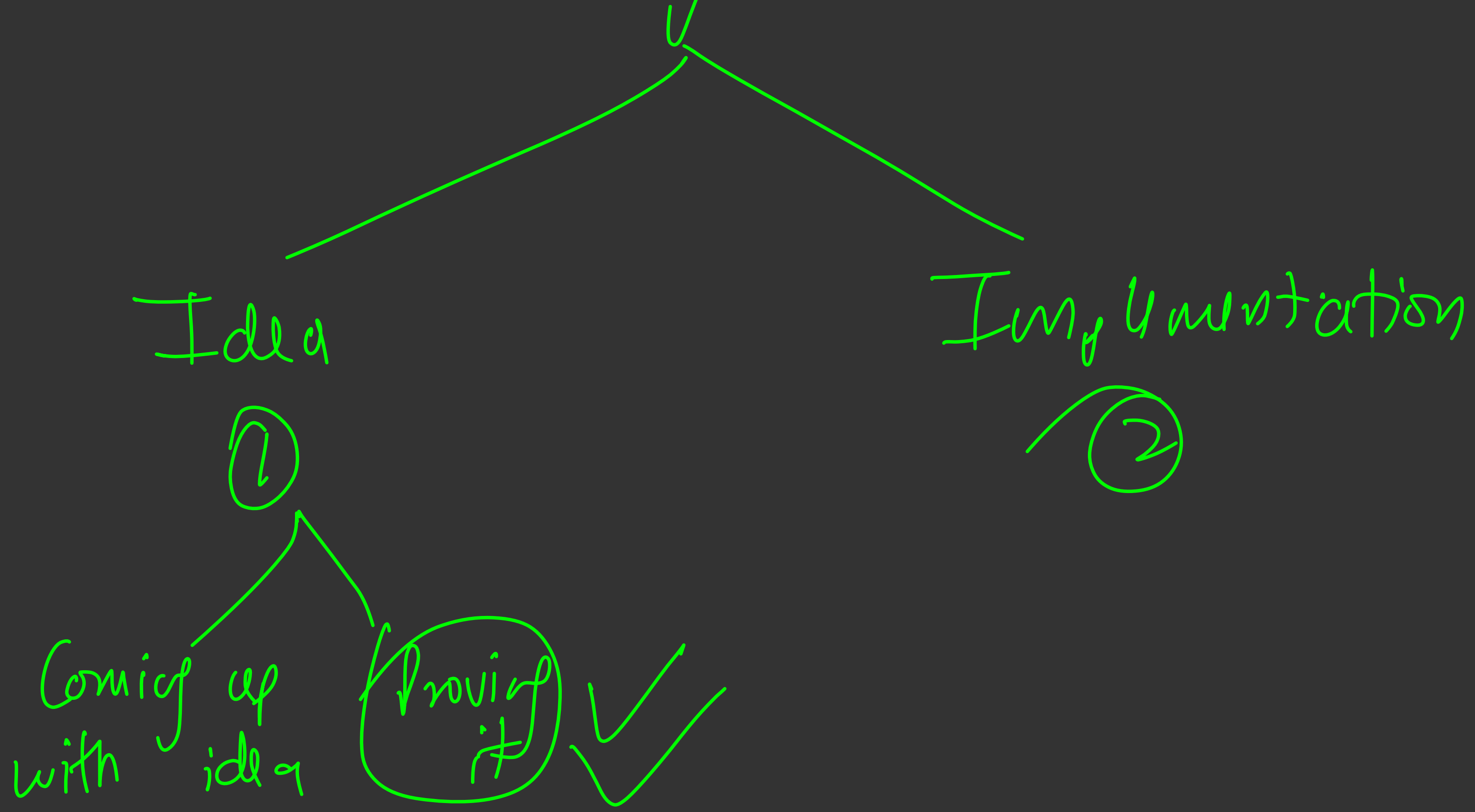
(profit - unit)

Given N items with each item having 2 parameters X_i and Y_i which means picking X_i units of item i will give you a profit of Y_i points. You are not required to pick all X_i units of an item. You are also given a value D. You can pick up to D units in total. Find the maximum profit you can make.



```
Sort ( items based on yi/xi ) → decreasing order
D
for(i=0, i<n ; i++)
{
        max_units = min( xi, D)
        ans += (yi/xi) · max_units
        D -= max_units }
cout << ans << endl;
```

Greedy Problems



Problem 6: Link

Homework