GREEDY

Good Evening

To do list

Greedy Properties

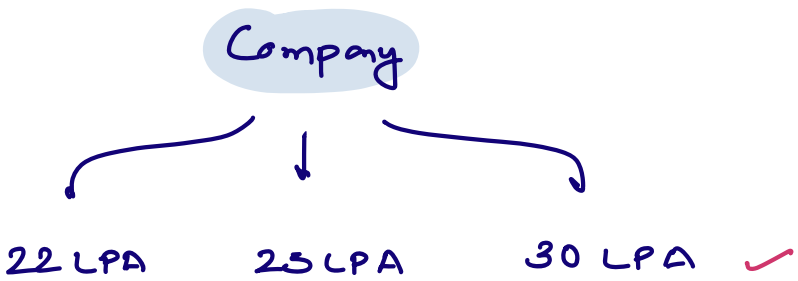
01. Free cars

02. Distribute candy

03. Activity Selection

Greedy → Solving the problem by using the best possible option at that time

→ Used for optimisation of min/max related problems.



→ remote?

→ work culture

→ timings?

→ Work-life balance?

Free Cars

01 There is a limited time sale going on for toys.

A[i] → sale end time for i^{th} toy

B[i] → Beauty of i^{th} toy

Time starts with T = 0 & it takes 1 unit of time to buy one toy & toy can only be bought if T < A[i].

Buy toys such that sum of beauty of toys is maximised.

A[i] =	3	1	3	2	3
	0	1	2	3	4

B[i] =	6	5	3	1	9

Idea 1 → Pick the toy with maximum beauty first

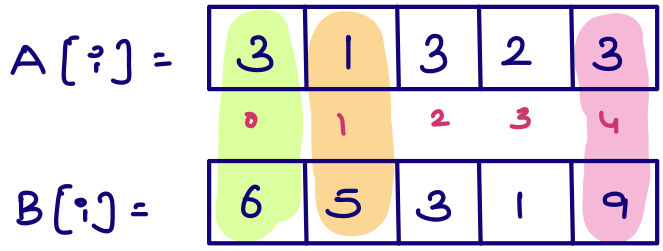
T = ∅

→ 4

→ ~~2~~

→ 3

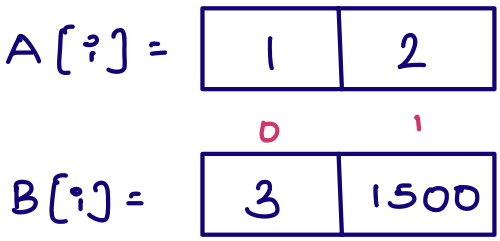
toy →	beauty
4 →	9
0 →	6
2 →	3
beauty =	18



T=∅ toy → beauty

✗	1 → 5
✗	4 → 9
3	0 → 6

beauty = 20



T=∅ toy → beauty

✗	0 → 3
2	1 → 1500

1503

* Idea → Buy everything → ascending order of time

A[] = { 1 3 3 3 5 5 5 8 }
0 1 2 3 4 5 6 7
B[] = { 5 2 7 1 4 3 8 1 }

T = ∅ ✗ ✗ ✗ ✗ 5 ans = 0 + 5 + 2 + 7 + 4 + 3 + 8 + 1

= 28

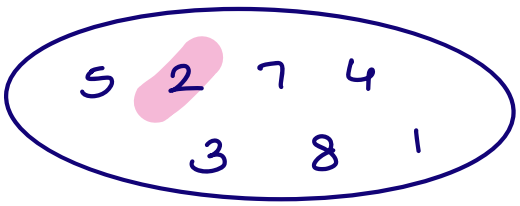
* Correcting the incorrect step that you have taken in

post

* Using min-heap

A[] = { 1 3 3 3 5 5 5 8 }
       0 1 2 3 4 5 6 7
B[] = { 5 2 7 1 4 3 8 1 }

T = 0 1 2 3 4 5



minheap

⇒ 5+7+4+3+8+1

= 28

* Pseudo code

01. Sort the array on the basis of sale ending time

02. Minheap heap;
T=0
for (i=0; i<n; i++){
  if (T < A[i]){
    heap.insert(B[i]);
    T++;
  }
  else{
    if (B[i] > heap.peek()){
      extractmin();
      heap.insert(B[i]);
    }
  }
}

TC: O(nlogn)
SC: O(n)

Ans = Remove all elements from heap & add them.

Chocolate Distribution

Given N students marks, assign chocolate to all students in such a way that :-

01. Each student get atleast 1 chocolate.

02. If (ar[i] > ar[i-1]) → chocolates assigned to i^{th} student should be more than $(i-1)^{th}$ student

03. If (ar[i] > ar[i+1]) → chocolates assigned to i^{th} student should be more than $(i+1)^{th}$ student

Calculate min no. of chocolates to assign to all N students.

Eg :- { 1, 5, 2, 1, 6 }

choc = 1, 1, 1, 1, 1 ✗

choc = 1, 5, 2, 1, 6 → 15 chocolates

choc = { 1, 3, 2, 1, 2 } Ans = 9

Eg :- { 3, 100, 60 }

choc = { 3, 100, 60 } - 163 chocolates

choc = { 1 2 1 } - 4 Ans

Idea

Greedy

If (ar[i] > ar[i-1]) ar[i] = ar[i-1] + 1

if (ar[i] > ar[i+1]) ar[i] = ar[i+1] + 1

arr[] = { 3 6 2 8 10 }

left = { 1 2 1 2 3 }

right = { 1 2 1 1 1 }

chocolates = 1 2 1 2 3 = 9 chocolates

arr[] = { 3 6 9 11 7 }

left = { 1 2 3 4 1 }

right = { 1 1 1 2 1 }

chocolates = 1 2 3 4 1 = 11 chocolates

arr [] = { 3 6 9 11 7 2 }

left = { 1 2 3 4 1 1 }

right = { 1 1 1 3 2 1 }

---

chocolates = 1 2 3 4 2 1 = 13 chocolate

arr [] = { 4, 4, 4, 4, 4 }

Chocolate = 1 1 1 1 1 → 5 chocolates

arr [] = { 1, 6, 3, 1, 10, 12, 20, 5, 2 }

left = { 1 2 1 1 2 3 4 1 1 }

right = { 1 3 2 1 1 1 3 2 1 }

---

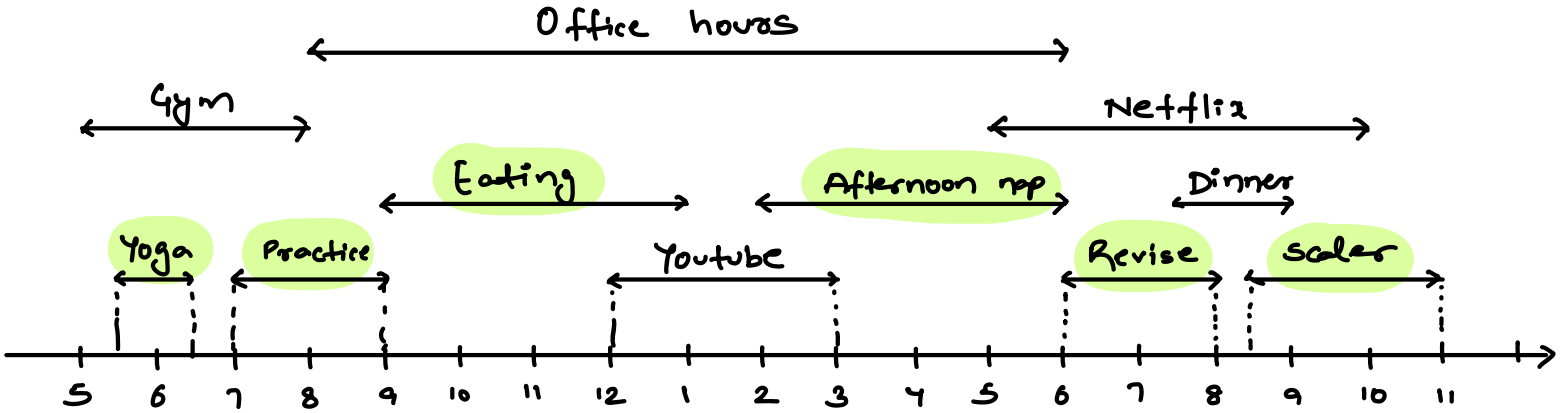
chocolates = 1 3 2 1 2 3 4 2 1

= 19 chocolates

10:22 → 10:32 PM

Activity Selection

Ans = 6



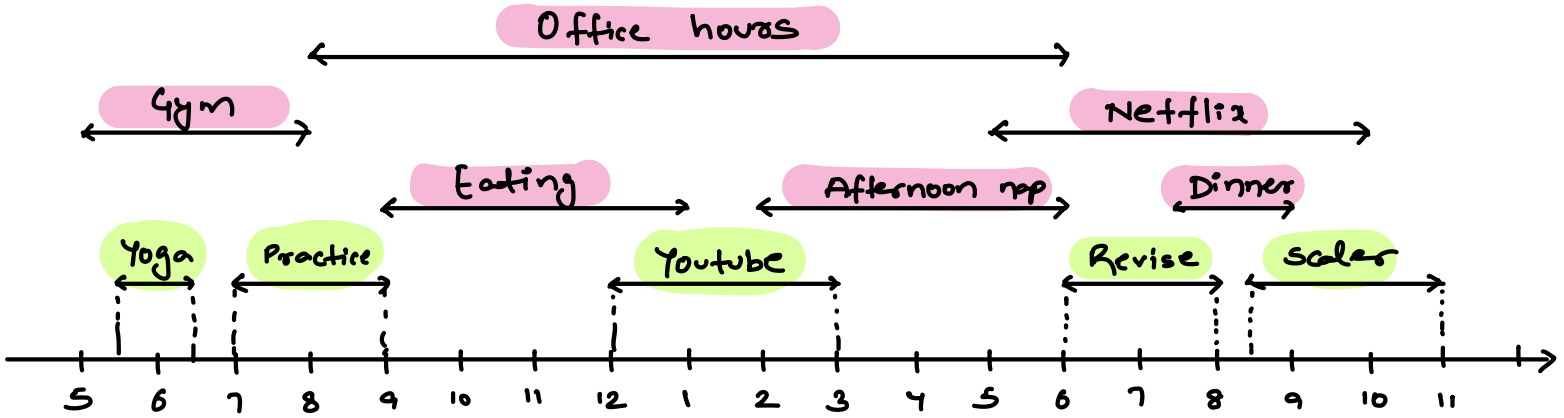
Q **Max tasks** which we can do.

Note 1 :- Once we start, we need to finish the task

Note 2 :- At any point, we can't do overlapping tasks.

Greedy

Parameter → Lesser duration task

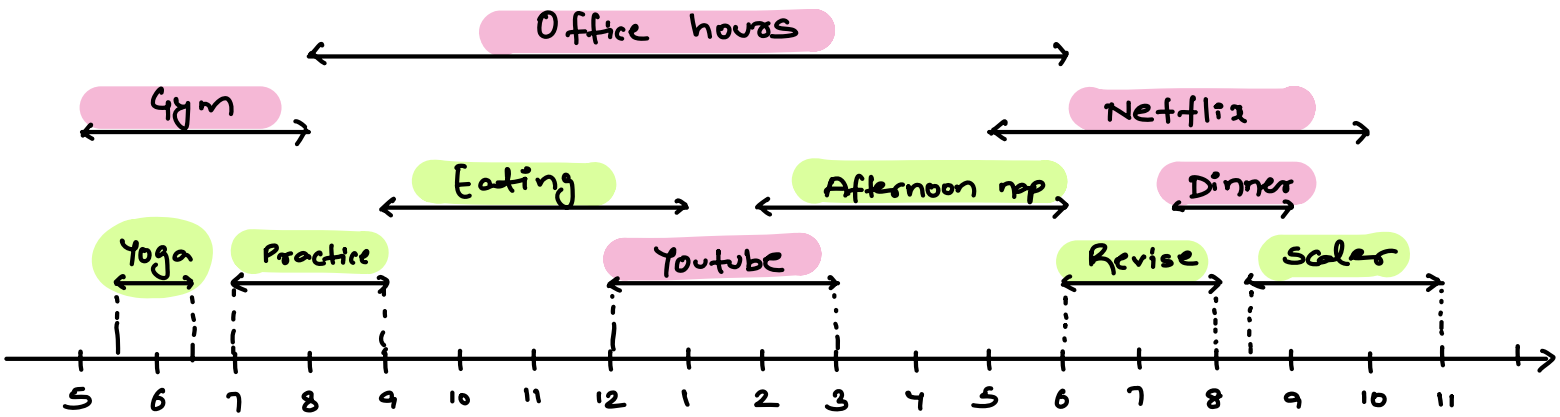


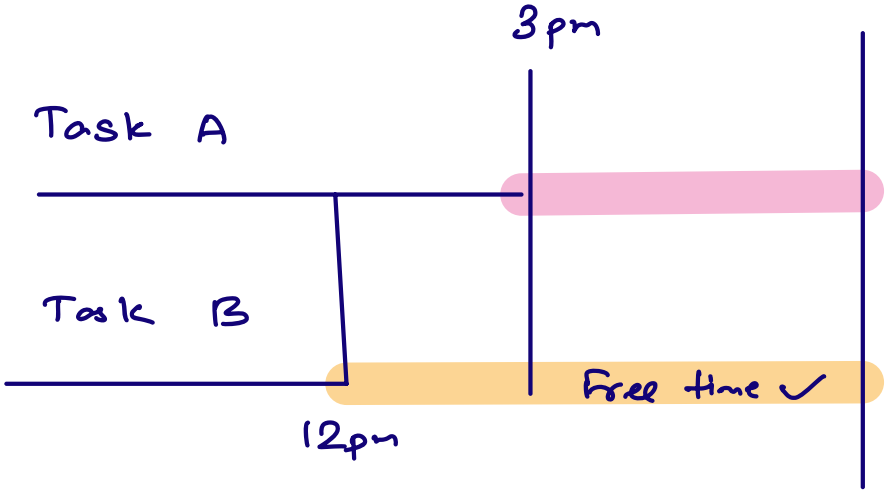
01 Yoga
02 Practice
03. Revision
04. Scaler
05. Youtube

} ✗

02. Parameter → Start time ✗

03. Parameter → Minimum end time





Obs

Choose greedily for the earliest finishing time

↓

For other task, i'm going to have more time

	st	end		sort on the basis of end time →		st	end	count	end
0	5	8	✓		0	5:30	6:30	1	6:30
1	5:30	6:30	✓		1	5	8		
2	7	9	✓		2	7	9	2	9
3	9	13	✓		3	9	13	3	13
4	8	18	✓		4	12	15		
5	12	15	✓		5	8	18		
6	14	18	✓		6	14	18	4	18
7	18	20	✓		7	18	20	5	20
8	19:30	21	✓		8	19:30	21		
9	20:30	23			9	17	22		
10	17	22	✓		10	20:30	23	6	23

Ans = 6

* Sort the array on basis of end time

* if (st of curr task $\geq$ end of previous task)

    task = task + 1

    end time = end of curr task;

}

* Metting Rooms - I, II & III
* Minimum platform
* Job Scheduling
* Activity Selection
* Candy distribution
* Minimum no. of arrows to burst balloon
* Gas station