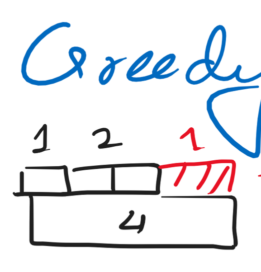
* Square Root using Binary Search
int n → range $[\underset{s}{0} - \underset{e}{\overset{36}{n}}]$

Find mid → $m \times m > n$
ans = -1
$e = m - 1$
mid → $\boxed{m \times m} < n$
Store the answer & move to right
ans = mid
$s = m + 1$
Once you get the answer, return mid.

* <u>Introduction to Greedy Algorithms:</u>

* Padding & Greedy Alignment!

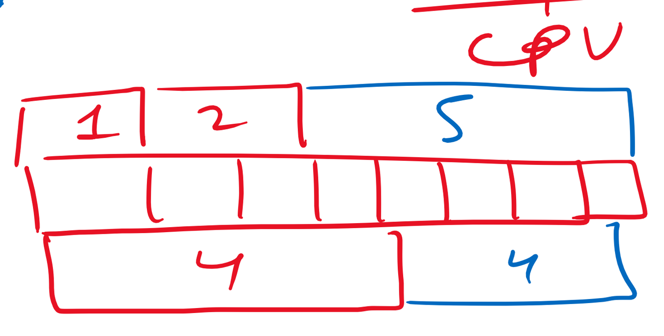
c
s
i



→ padding
↓
compiler
CPU

1 → 5

c
s 8
d → 8
i 4



8 bytes

= 24

* Minimum Number of Coins
* Minimum Cost of Ropes
* Activity Selection Problem
* Maximum Absolute Difference is minimum. (Chocolate Distribution)
* Job Scheduling Problem
* Huffman Encoding
* Policemen & Thieves
* Nikunj & Donuts
* Fractional Knapsack
* 0 - 1 Knapsack
* Min Cost to Climb Stairs

LeetCode / GFG / Coding Ninjas

Min no of Coins: →

Denomination of Coins

coins ⇒ {1, 2, 5, 10, 20, 50, ~~100~~, ~~500~~, ~~1000~~, ~~2000~~}

V = 91   90+1   50, 20, 20, 1   1, 2, 3, 4

91 - 50 = 41   >=

V = 31   41 - 20 = 21

20, 10, 1   21 - 20 = 1

3   1 - 1 = 0 → Stop

Activity Selection Problem: →

Given a set of activities with their start & finish times, select the maximum number of activities that can be completed by a single person, assuming that a person can work on only one activity at a given particular time.

Activity	Start Time	Finish Time	Sort - Finish Time	O/P
A1	5	7	A3 1-4	A3 1-4
A2	8	9	A6 3-5	
A3	1	4	A5 0-6	A1 5-7
A4	5	9	A1 5-7	
A5	0	6	A4 5-9	A2 8-9
A6	3	5	A2 8-9	

3

Steps ① We sort all the activities according to their finish time.
② We start by selecting the activity which finishes earliest. For each activity, if the finish time is after or equal to the current selected activity, we select it.
③ The final selected activities are those which can be completed without overlapping.

Minimum Cost of Connecting n Ropes:

{4, 2, 3, 6} → O/P = $\boxed{29}$

Sort → (2, 3, 4, 6)   2+3 = 5
n log n

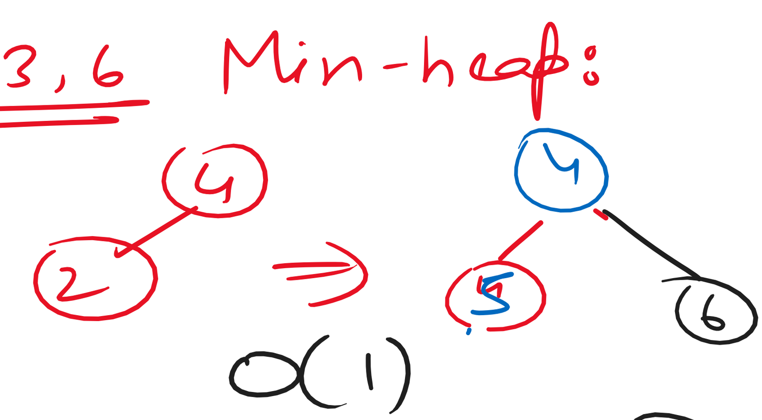
Sort → (5, 4, 6)   (4, 5, 6)   4+5 = 9
n log n

Sort → (9, 6)
n log n   (6, 9)   6+9 = 15

(very big TC)

Cost
5
+
5 ↓
14
+
15
= 29

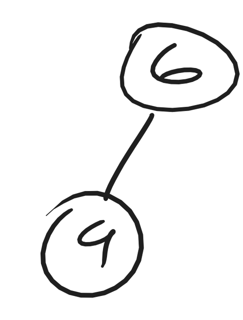
4, 2, 3, 6   Min-heap:



O(1)

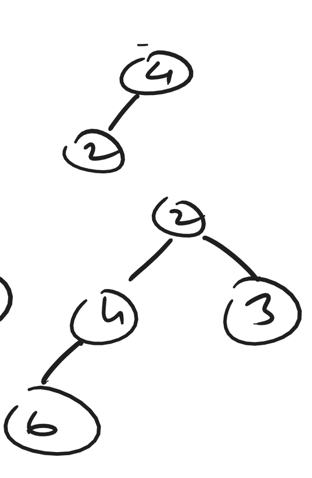
C = 5
+
9
+
15
= 29

pop →
2+3 = 5
4+5 = 9
6+9 = 15



cost = 0
arr = 4, 2, 3, 6 → mHeap

First = 2, 4
Second = 3, 5
merged = 2+3
= 5
4+5
= 9



4, 5, 6
9, 6
15

cost = 5, 9, 15
= 29

28th Feb - 2025   (9520)

* Chocolate Distribution Problem

---

* Graphs → Types, Definition
* Graph Representation
  → Adjacency List
  → Adjacency Matrix

* Graph Traversals: →
  → BFS Traversal
  → DFS Traversal

**** Dynamic Programming: Space
Recursion, Memoisation, Tabulation, Optimisation