T - 0 S $\xrightarrow{500}$ N $\xrightarrow{500}$ V 24 $\Rightarrow$ ⊘

S $\xrightarrow{500}$ Q $\xrightarrow{500}$

(diagram with circular arrows connecting S, N, V, Q)

S $\xrightarrow{500}$ X B $\xrightarrow{500}$ M $\Rightarrow$ ⊥

S $\xrightarrow{6}$ M

S $\xrightarrow{500}$ M

→ One of the Most asked Interview Question

→ LLD (machine coding)

↓

Design Splitwise

⇒ Steps for a machine coding round ?

⇒ Overview

⇒ requirement gathering [might not be required for machine coding]

⇒ Class diagram

=> schema design

=> code

=> done

i) User can register and update their profiles.

ii) User can add friends

iii) User can create groups.

iv) we can add/remove friends from group.

v) User can add expenses.

vi) for an expense, we can add the following details

    i) description

    ii) currency

    iii) amount

    iv) date of expense

    v) who or (who all) are paying
       and how much

vi) Split strategy (equal or by specific amounts)

vii) add the people who were supposed to pay

vii) Group will have a list of expenses.

viii) User's can settle up expenses, while setting up, we should show minimum possible no. of transactions to fulfil all debt.

⇒ input ⇒ APIs

⇒ persistence ⇒ yes | DB (SQL)

⇒ Minimum Transaction Settlement [Simplify Debt]

⇒ given a no of transactions in a group of people, find the minimum no. of transactions required

⇒ NP-HARD

$$[ax^3 + by^2 + cz]$$ =) O/p

$$[x, y, z]$$ → polynomial

TC or SOM =) polynomial

⇒ group ⇒ A B C D

Expense 1 =)   who paid ⇒ A = 1000   B = 1000

who had to ⇒ A = 500 B = 500 C = 500 D = 500

Expense 2 ⇒   who paid ⇒ A = 3000

who had to ⇒ A = 1000 B = 200 C = 800 D = 1000

Expense 3 =)   who paid ⇒ C = 500   D = 800

who had to ⇒ A = 500 B = 100 C = 200 D = 500

Expense 4 ⇒   who paid ⇒ D = 1000

who had to pay ⇒ A = 250 B = 250 C = 250 D = 250

total (who paid) = = total (who had to pay)

7300   7300

⇒ find the total outstanding amount for
each person :

	paid	debt

1) A = 4000 — 2250 ⇒ (1750)

ii) B = 1000 — 1050 ⇒ - 50

iii) C = 500 — 1750 ⇒ -1250

iv) D = 1800 — 2250 ⇒ — 450

$$0$$

⇒ basic sol^n

1750 + 1750

-50 + (-1750) = 1800

1750
+ 1750
  2500
—3500

-450

-300

-1250

-1250 — 1800
= -3050

go clockwise

O 1750

(A) ──────────1750──────────→ (B) −50

3500

−450
(D) ←──────────────────── (C) −1250 = −1800
                                        ────────
                                        − 3050
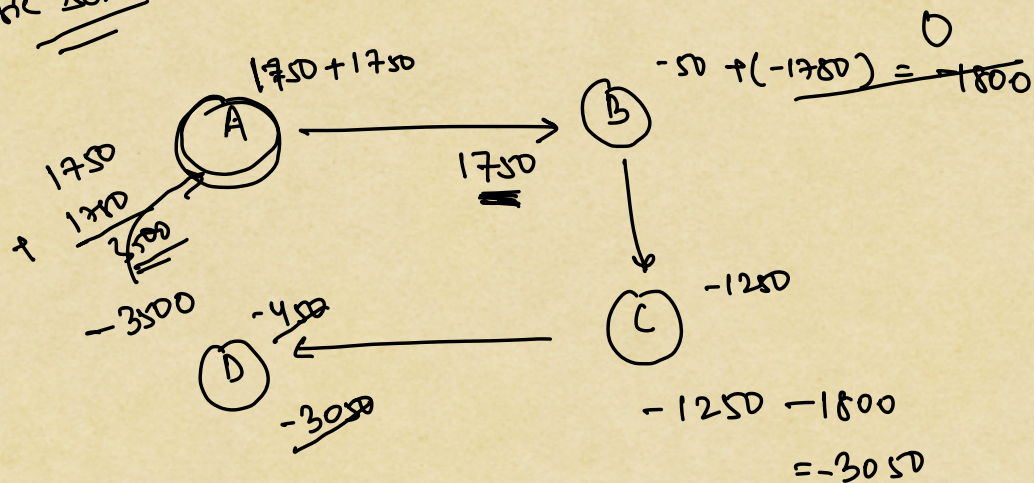
=> −50 −1750
=> (−1800)
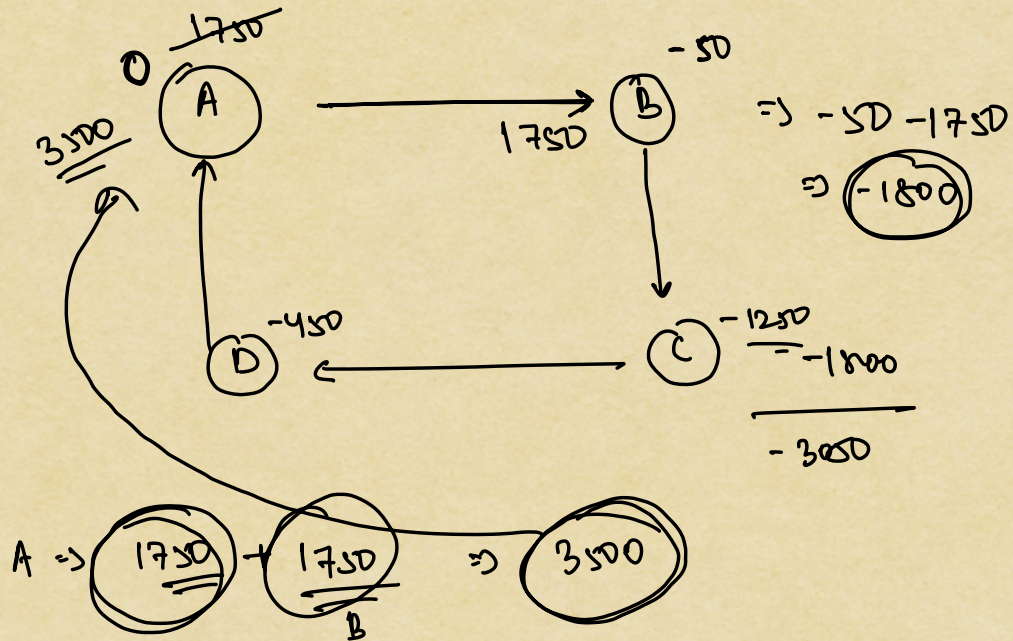
A => (1750 + 1750) => (3500)
              B

B = 50 + 1750 => 1800
        ────────   ═══
        took A

C = 1250 + 1800 => 3050

D = 450 + 3050 => 3500
         ────
          C

Practical
sol^n

⇒ find the total outstanding amount for
each person↓

           paid   debt

1)   A = 4000 — 2250 ⇒ (1750)

2)   B = 1000 — 1050 ⇒   — 50

3)   C = 500 — 1750 ⇒ —1250

4)   D = 1800 — 2250 ⇒ —450
                              _____
                                 0
                              _____

idea ⇒ lets settle the biggest lendee to
        the biggest lendor

        A = 1750 ⟶ biggest lendor
        B = —50
        C = —1250 ⟶ biggest borrower/lendee
        D = —450.

$T1 \Rightarrow$ $C \rightarrow A \Rightarrow 1250$

$A = \cancel{1750}\ 500$

$B = -50$

$C = 0$

$D = -450.$

$T2 \Rightarrow$ $D \longrightarrow A \Rightarrow 450$

$A = 50$

$B = -50$

$C = 0$

$D = 0$

$T3 \Rightarrow$ $B \rightarrow A \Rightarrow 50$

$A = B = C = D = 50.$

$A = 500$ $E = \cancel{1000}\ 0$

$B = \cancel{6000}\ \cancel{3000}\ 500$ $f = \cancel{-3000}\ 0$
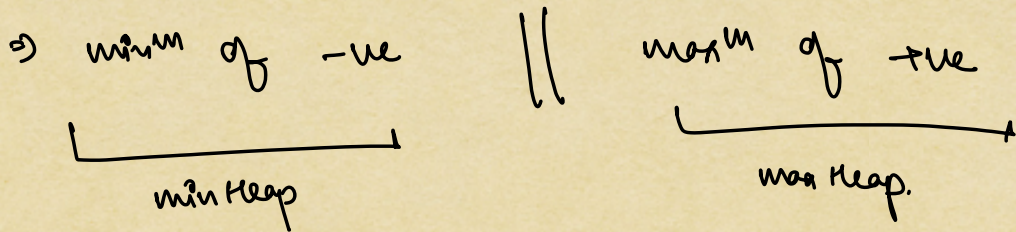
$C = 750$ $G = \cancel{-2500}\ 0$

$D = \cancel{-1750}\ -750.$ $H = -1000$

T1 => F → B = 3000.

T2 => h → B = 2500

T3 => D → F => 1000

=> minimum of -ve || maximum of +ve

‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾ ‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾

min Heap   max Heap.

Steps

=> find outstanding balances for each people

=> Store all borrowers into min Heap

=> " " lenders " max Heap

=> settle largest borrower (minimum of min Heap)

& largest lendor (maximum of max heap)

=> update the balance, return back to heaps

=> keep doing until all are at 0

(I) ⇒ all lecture content ⇒ LLD 1|2|3

(II) ⇒ psp for LLD ⇒ 85%

LLD 1|2

(III) mock LLD-2

(IV) contest LLD 3