Design splitwise

start @ 9.05pm

↳ Machine coding interview

- Overview
- Requirements
- Clarify the requirements
- Class diagram
- Schema design
- Code

Overview

↳ Expense tracking + sharing application

A B C D

↳ Dinner — Paid 2000

A	500
B	500
C	500
D	500

Goa trip — A B C D

(7 days)

E1 Hotels A 10000 →

E2 Dinner B 1000 →

E3 Water C 2500 →

⋮

Weekend Party — 4000/-

	Who paid How much?	Who owes what amount?
Suyash	— 2000 paid	1000
Biswojeet	— 800 paid	1000
Chetan	— 1200 paid	1000
Nikul	— 0	1000
	PAID	Had to pay

=> Suyash should get 1000 Rs back
=> Biswojeet should pay 200 Rs more
=> Chetan should get 200 Rs
=> Nikul should pay 1000 Rs

Total amount paid = Total amount had to pay

Settle up

=> Biswojeet Pays 200 to Chetan
=> Nikul pays 1000 to Suyash

Expense

Amount

< Paid

Had to pay

2000 (food + Drinks)

	Paid	Had to pay
Nikeil	1000	400
Saket	1000	600
Jasmeet		600
Rush		400
	2000	2000
	Paid	Had to Pay

User Paid < User, amount >

Nikeil 1000
Saket 1000

User Had to Pay < User, amount >

Nikeil, 400
Saket, 600
Jasmeet, 600
Rush, 400

How much amount every user has paid extra

=> paid amount − Had to Pay amount

Nikeil = 1000 − 400 = +600
Saket = 1000 − 600 = +400
Jasmeet = 0 − 600 = −600
Rush = 0 − 400 = −400

Note: If amount paid by any user

i) >0 : They will get amount

ii) <0 : They have to pay

way1
- T_1 = Jasmeet → Nikil 600
- T_2 = Rush → Saket 400

If we perform all transactions then every user will be zero

⇒ Minimum no. of transactions to settle up all expenses

way2
- T_1 → Jasmeet → Saket 400
- T_2 → Jasmeet → Nikil 200
- T_3 → Rush → Nikil 400

Problem Statement

Assume there are n expenses that happened in a group. If someone clicks on **settle up** button then it shows a list of transactions which if performed will settle all the user of that group

Who should pay whom & how much amount.

Expense

amount

→ who paid => [Person, amount]

→ read to pay => [Person, amount]

Exp 1 (ABCD)

Who Paid A: 1000 B: 1000

who had to Pay A: 500 B: 500

C: 500 D: 500

Exp2

Who Paid A: 3000

who had to pay A: 1000 B: 200

C: 800 D: 1000

Exp3

Who Paid D: 800 C: 500

Who had to pay A: 500 B: 100

C: 200 D: 500

Exp4 Who Paid D: 1000

had to Pay: A: 250 B: 250

C: 250 D: 250

for every user:

amount paid extra = 0

for every expense {

amount_paid_extra += whopaid[user]

amount_paid_extra -= readtopay[user]

}

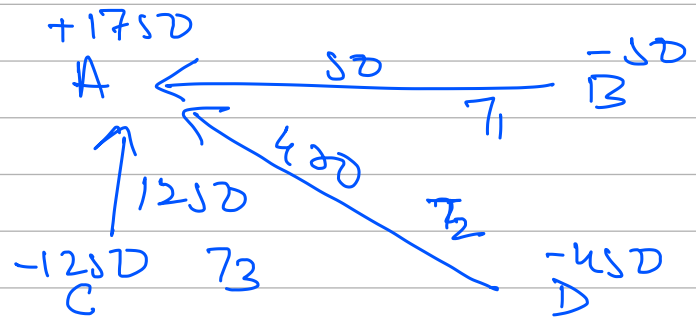
A: 1000 - 500 + 3000 - 1500 + 0 - 500 + 0 - 250

= +1750

B = 1000 - 500 + 0 - 200 + 0 - 100 + 0 - 250 = -50

C : −1250

D : −450



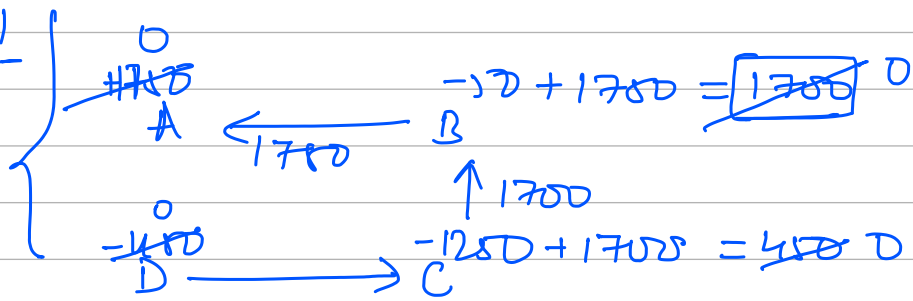
NOTE

⇒ Minimum # of transactions ⇒ NP Hard Problem

↙

No polynomial time solution

Sol 1



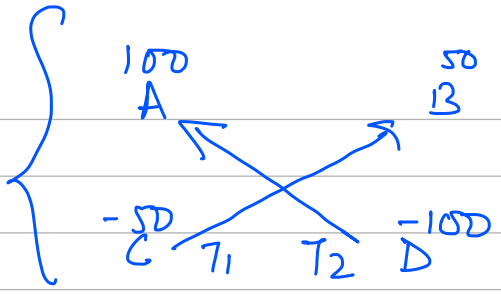
T_1 = B → A = 1750

T_2 = C → B = 1700

T_3 = D → C = 450

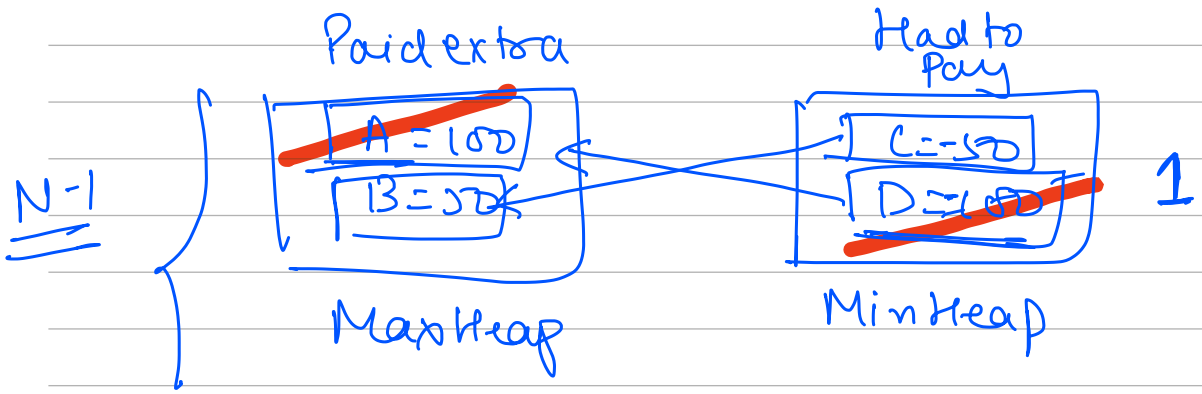
# of transaction = N users − 1

eg



SCC = 3

Who had paid extra ← Who has to pay



A → D 100
B → C 50

M → [log(N)] = [N(log N)]

eg

Paid
A = +1500
C = +700

Had to Pay
B = -400
D = -1200
E = -600

D → A 1200
E → C 600
B → A 300
B → C 100