Today's Agenda:-

3 case studies
- TTT
- Parking Lot
- BMS

1) Overview
   - Know
   - Don't know

2) Clarify the given set of requirements.

3) Class Diagram

4) Schema Design.

5) Code

Splitwise :- Expense sharing app".

List<Txn>

Expense A, B, C, D

Total amount ₹ 5000

Paid by A : ₹ 2000
B : ₹ 3000

Had to pay A : 1000
B : 2000
C : 500
D : 1500

2000 − 1000 | 3000 − 2000 | 0 − 500

A | B | C

+1000 | +1000 | −500

0 − 1500

D

−1500

A paid 1000 extra
B " " extra
C paid 500 lesser
D paid 1500 lesser

Total amount extra = Total amount lesser

2000 = 2000

{ C → A 500
D → A 500
D → B 1000 } to settle up

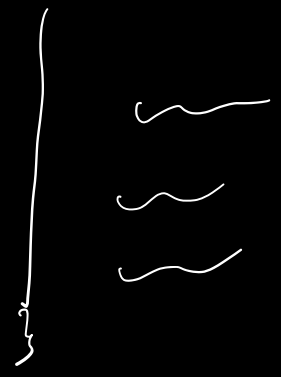
Users who paid extra should get back money from people who paid lesser.

After settle up, nobody owes anything.

Settle Up Algorithm.

Problem Statement :- There are N no of expenses that we want to settle up, we want to return a minimum list of transactions required.

```
List<Txn> SettleUp ( List<Expense> ) {
```



Group

A, B, C, D

$$\sum \text{paid} = \sum \text{had to pay}$$

0 — 1750

1000 — 2250

E1 who paid : A : 1000 B : 1000

had To pay : A : 500 B : 500 C : 500 D : 500

Map < user, amount > paid

had To Pay

E2 who paid : A : 3000 —

had To pay : A : 1000 B : 200 C : 800 D : 1000

E3 who paid : A : 500 B : 800

had To pay : A : 500 B : 100 C : 200 D : 500

E4 who paid : D : 1000

had To pay : A : 250 B : 250 C : 250 D : 250

↳ Compact the data into simpler form

Q for each user, find the balance ⊕ → extra ⊖ → lesser

A : paid : 4500

had to pay : 2250

} overall A paid extra

4500 − 2250

+ 2250

=

B : +750 C : −1750 D : −1250

```
for each user
    paid = 0, hadToPay = 0
    for every expense
        paid += paid[user]
        hadToPay += hadToPay[user]
    balance = paid − hadToPay
```

A :	+2250
B :	+750
C :	−1750
D :	−1250

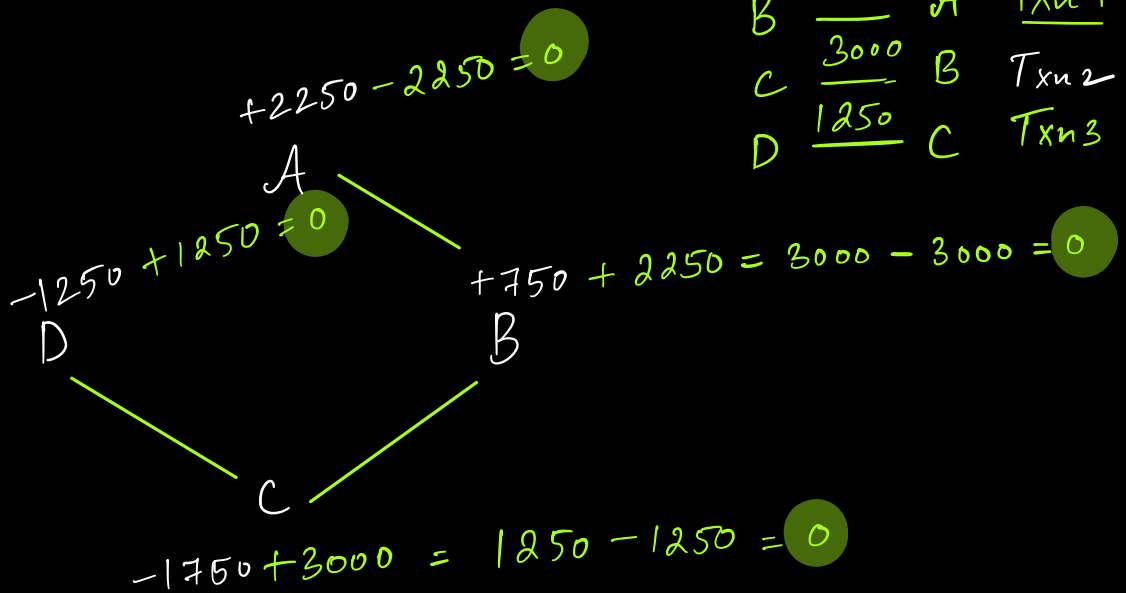
looking at these balances,
I need to devise an
algo to find min no
of txn to settle up
all the expenses.

} Greedy approach.

Solution 1: Go clockwise | anticlockwise & try settling up pair of users.

A +2250 extra
B +750 extra

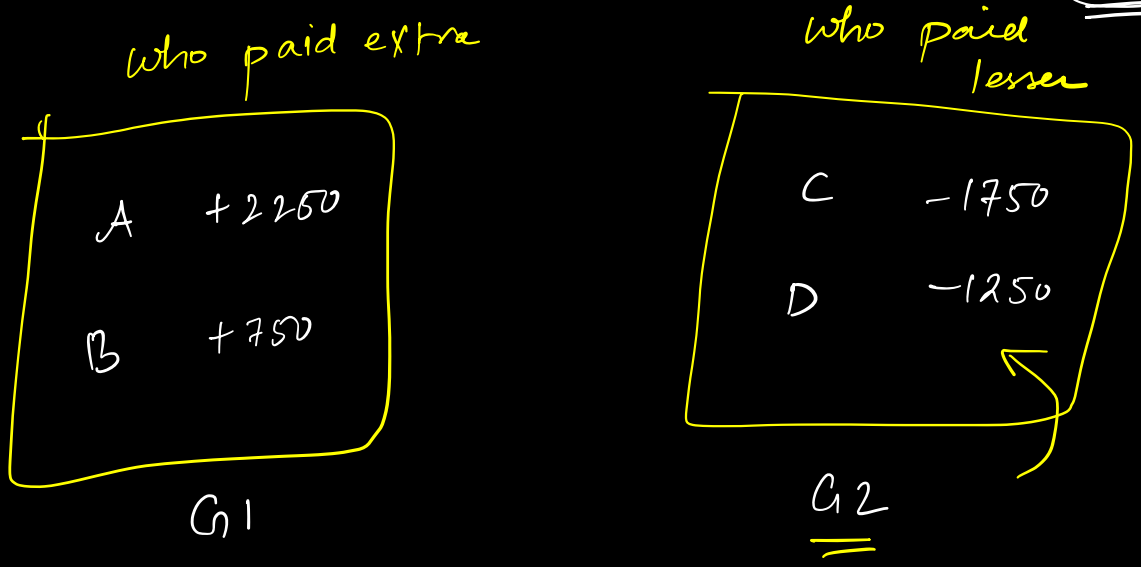
B 2250 A Txn 1
C 3000 B Txn 2
D 1250 C Txn 3



Is this soln practical ? ⇒ No.

Sol 2 Divide the users into 2 buckets

DSA



Group 2 users should pay to Group 1 users

↳ Give you as a Small HW

Hints

- Min of bucket
- Max of bucket
- Heap DS

Continue right from here...

Thank You