AGENDA

1.> DESIGN SPLITWISE

"Design splitwise"

A B C → lunch

✓

1200/-

↓

400 PP

A → 1200

A → expense "lunch" → ✓

(A, B, C) → 1200

≠ GROUPS:

"GOA" 5 days.

d_1 → lunch
dinner
transport

d_2 → —
—
←

[A, B, C, D, E]

↓

3 DRINKS
2 SFT DRINKS

20,000 500

E_1 → DRINKS (A, B, C) → 20,000

$E_2 \rightarrow$ SD (D-E) $\rightarrow$ 500

[A, B, C, D, E]

21,000

(A, B, C)

A Paid $\rightarrow$ 21,000

7000

Who Paid what
A Paid 21,000

Who owes what
B owes 7000
↓
(had to pay)

Who Paid what v/s who owes what

500
GoA (x)
U
Flat (y) 200
= (x+y)

Mohsin owes 700/-

Me Alex

-5000 +5000

Settle up balance (0)

(i) A* → B 1000 ✓

(ii) B → C 500 xx

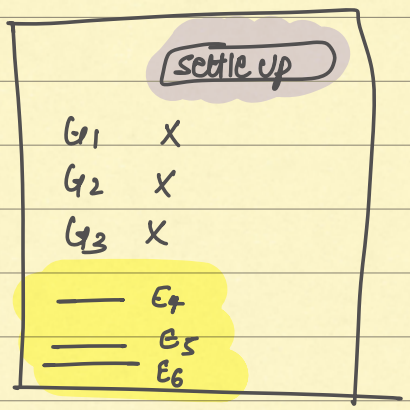
(iii) C → A* 250 ✓

= 0 XXX

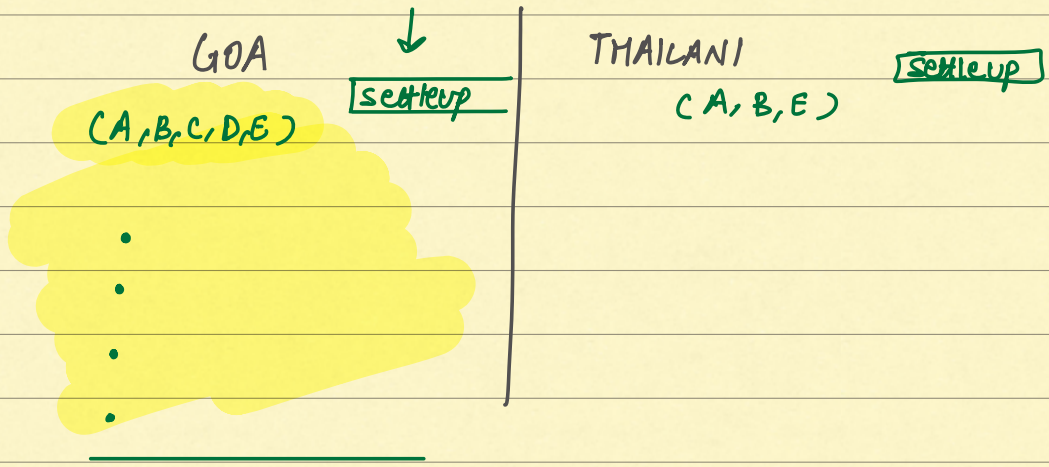
A settle up

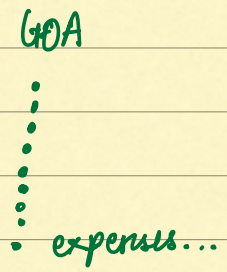
GLOBAL →

GROUPS



X





settleup

A: 2000
B: 500
C: 10,000
D: 0
E: 4000

who owes how much to whome

0

Net balance = 0

A → 5000
B 10000

2000
2000

C 000 2000

3000 /-

C → -2000
A → +30000
B → -1000

= 0

MINIMISE THE TRANSACTIONS' :

[A, B, C, D]

A: Get 100/-
B: Pay 200/-
C: Get 200/-
D: Pay 100/-

settleup ✓

1.) manually → 3 transactions
2.) direct → 2 transactions

↓

discuss

*) HOW SETTLE UP IN SPLITWISE:

Group

'm' expenses added ...

↓

who should Pay whome, how much ??

Group of Expenses
two states:

1) either someone will PAY
2) ——————→ GET

MIN TRANSACTION TO SETTLE UP

P1: How to calculate who will PAY / GET amount.
P2: FROM WHOME TO GET / who should Pay amount
P3: find MIN # transactions to settle up.

A B C

A = 500 +
B = -200
C = -300

PROB 1:

How to calculate who will PAY / GET amount.

- ✓ whoPaid: <Person, Amount>
- ✓ whoHadToPay: <Person, Amount>

A, B, C, D

$-500 - 800 + 500 - 200 - 250$

(1) Expense 1

whoPaid A: 1000 B: 1000

whoHadToPay A, B, C, D → 500

(2) Expense 2

whoPaid A: 3000

whoHadToPay A: 1000, B: 200, C: 800, D: 1000

(3) Expense 3

whoPaid C: 500 D: 800

whoHadToPay A: 500 B: 100 C: 200 D: 500

(4) Expense 4

whoPaid D: 1000

whoHadToPay A, B, C, D: 250

HOW TO FIND FINAL BALANCE:

```
for every person:
    extraamount=0
    for every expense:
        extraamount += whoPaid[person]
        extraamount -= hadToPay[person]
```

(A):

~~ea+1000~~ ea=0

(0+1000) = 1000

1000 - 500 = ea

~~500~~ + [3000 - 1000] + [~~-500~~] + [-250]

= 2000 - 250 = +1750

A: +1750

B -50

C -1250

D -450

0

MINIMIZE NO. OF TRANSACTIONS → NP-H PROBLEM.

GREEDY

SOLUTIONS: (P2)

1.> SOLUTION-1

[A, B, C, D]

A: +1750 ✓

B: -50 (1750) ✓

C: -1250 (+450)

D: -450

↑ =0

(A) ✓
(B) ✓
(C) ✓
(D) ✓

-> Go from A → D

-> settle everyone Individually:

1.> Get

2.> Pay

Get: Take exact MONEY from next person.

A: +1750
B → 1750 ↑

•> B pays 1750 to A
A: 0
B: 1750 − 50 = +1700

•> B = ~~+1700~~ 0
C = 1700 − 1250 = +450

•> C = +450 ↑
D = −450)

•) (N-1)

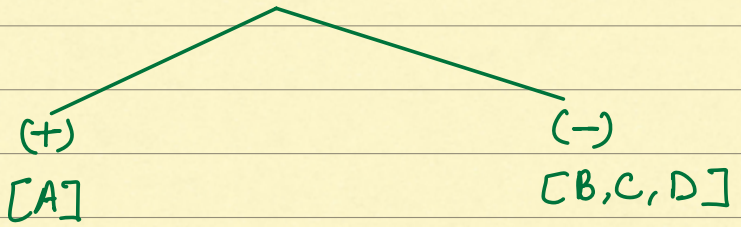
[A, B, C, D]

A: +1750 ✓
B: −50 (1750) ✓

↘ ↓
C: −1250 (+450)
D: −450
↑ = 0

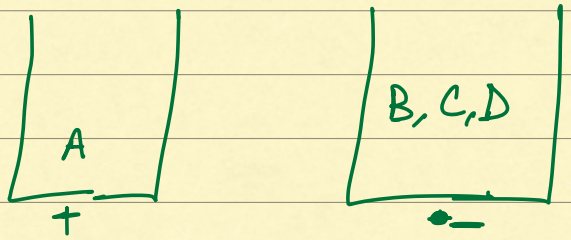
PRACTICAL SOLUTION:

(GREEDY)



1.) divide People into (2) Groups
+/-

2.) while both sides are NOT EMPTY:



·) Get Person with MOST amount (+)
·) Get Person whose has to pay MOST amount (-)

→ X (+) Get 500 ✓
Y (-) Give 600 — (i)
y = -100

X Get 600 x = +100
Y Give 500 ✓ — (ii)

X get 500 ✓
Y give 500 ✓ (-iii)

•) any of them settled up,
Remove them from Queue

•) put them back else on
Queue

DRY RUN:

(+) A:
(-) B:

1st

+ 1750
- 1250

Min (1750, 1250)

C ✓

2nd

+ 500(A)
- -450(D)

MIN (500, 450)

D ✓

3rd

+ A: 50
- B: -50

0 =

MaxHeap

HIGHLIGHTS:

✓ :)

(N-1)

② people will get settled in last transaction

TC: (N-1) * logN → N.logN

logN)

HW

1.) define classes / attributes
2.) define schema diagram
3.) code models