Agenda :-

1) Complete Tic Tac Toe

2) Design of parking lot

⇒ Tic Tac Toe :-

app 1 ⇒ after every move, check if any player is a winner

loop (N-1) players :-

iterate through entire board to check if any player is a winner ⇒ $O(N^2)$

TC ⇒ $O(N^3)$

app 2 ⇒ check for the player, if he/she is a winner after their last move :-

⇒ iterate through entire board to check if the player is a winner ⇒ $O(N^2)$

TC ⇒ $O(N^2)$

opt ⇒ we need to check only the row, column and diagonal involving the last move cell, for last player

⇒ iterate through row, column and diagonal involving the last move cell.

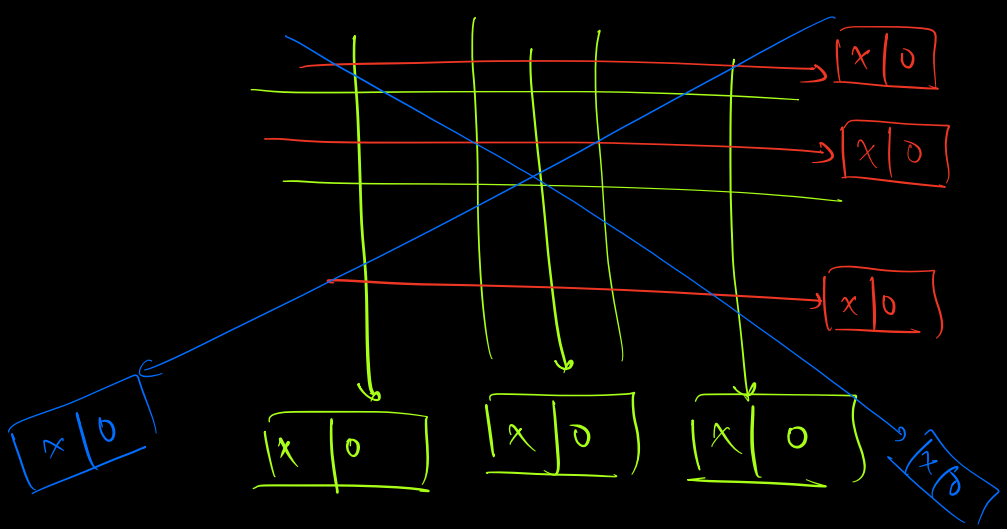
→ row ⇒ O(N)

→ col ⇒ O(N)

→ diag ⇒ O(N)

⇒ TC ⇒ O(3N) ~ O(N)

: $O(N^3) \longrightarrow O(N^2) \longrightarrow O(N) \longrightarrow O(1)$??



⇒ maintain N hashmaps for N rows

N hashmaps for N cols

2 hashmaps for 2 diagonals.

If at any hashmap for a row / colm / diag, we get 'N' no. of same symbol, we have a winner

O(1) → update hashmap after every move

O(1) → check row / colm / diag. if we have N symbols

↓

TC = O(1)

How

⇒ Undo code

⇒ Fix issue

⇒ Draw Logic

⇒ DESIGN PARKING LOT :-

⇒ Align yourself and get some overview

⇒ Requirements :-

i) Multiple floors, each floor has multiple slots

ii) Multiple entry points

iii) Multiple exit points

iv) Multiple types of vehicle

v) At entry a ticket is generated

vi) Assign a spot at entry

vii) Payment has to be done at exit

viii) Diff. types of vehicles will have different types of parking slots. A vehicle will have a single slot/spot.

ix) Diff. types of payment methods. (Cash/online)

x) Ticket will contain operator details

xi) Ticket should contain entry time & vehicle details.

⇒ Qn regarding Implementation details :-

i) Support multiple ways of assigning a spot.

ii) How is the bill going to be calculated :-

→ duration of parking

→ type of vehicle

iii) Multiple strategies of calculating fare.

iv) Payment algo :-

⇒ for every type of vehicle, diff base price / hour

⇒ hourly multiplier [can be different for diff vehicles]

0 - 2 ⇒ 1x

2 - 6 ⇒ 2x

6 - 10 ⇒ 3x

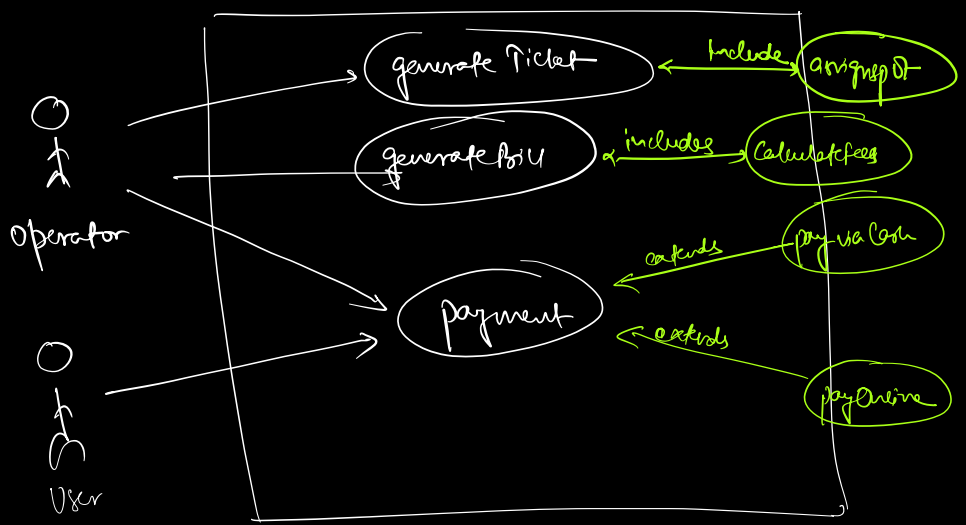
[20
40
60
80
...]

v) Support electric vehicles

→ spot should have charger

→ fare calculation → base parking fare + electricity used

=> Use Case Diagram



=> Class Diagram

↓

Code