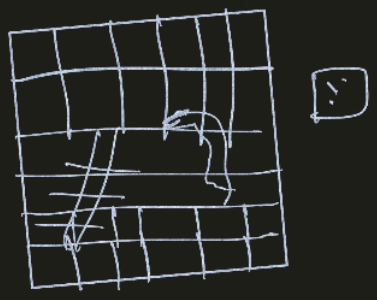
12 Snake and ladder

Rough flow



Requirement clarification

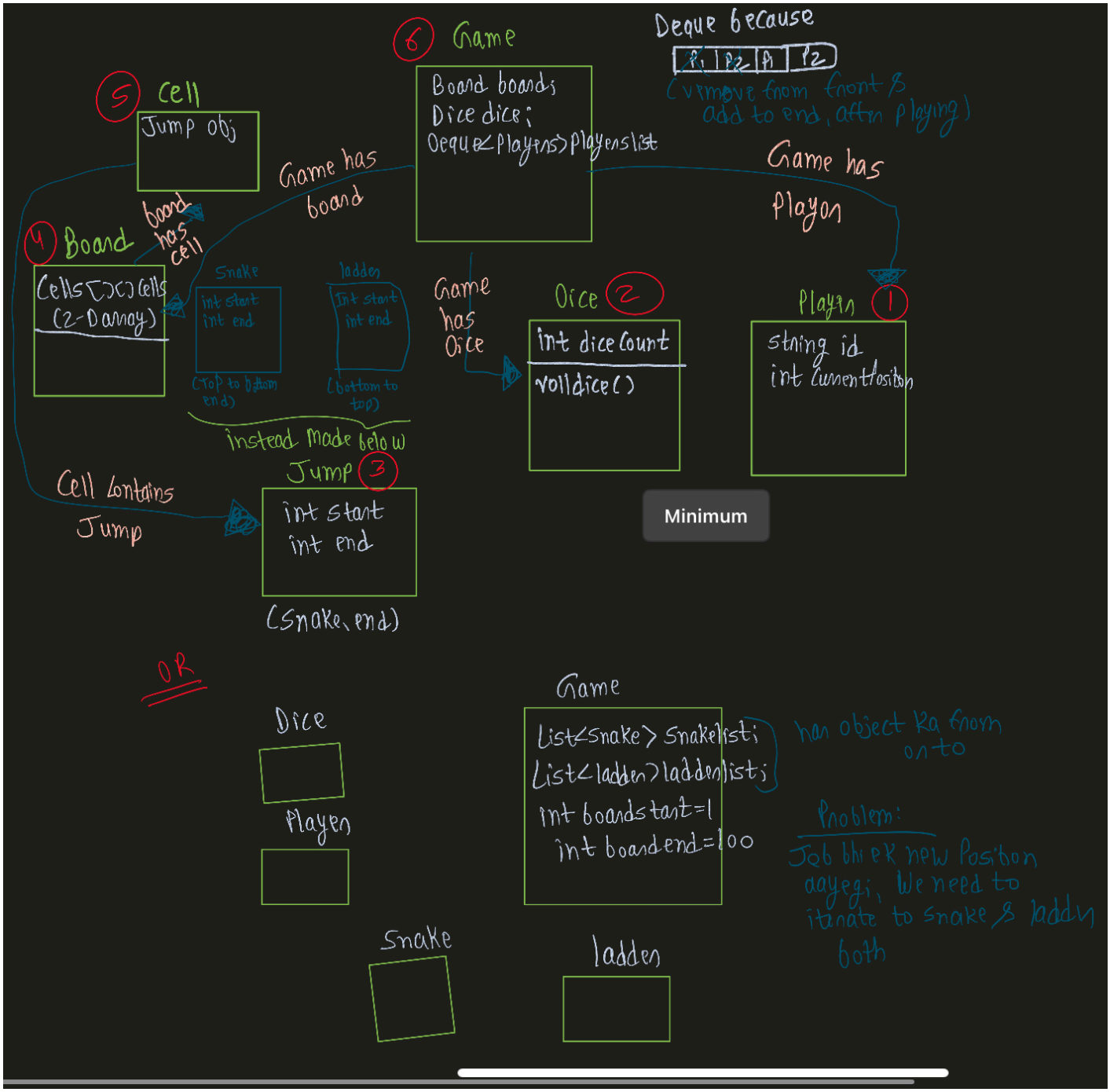
* How many dice?
  1, but should be Scalable
* How many Snakes & ladden?
  Setup time →
  We should be able to dynamically define
* What should be Winning Condition?
  (1 Player Win, game stops?
  on 1-by-1 Wins)

Bottom to Top

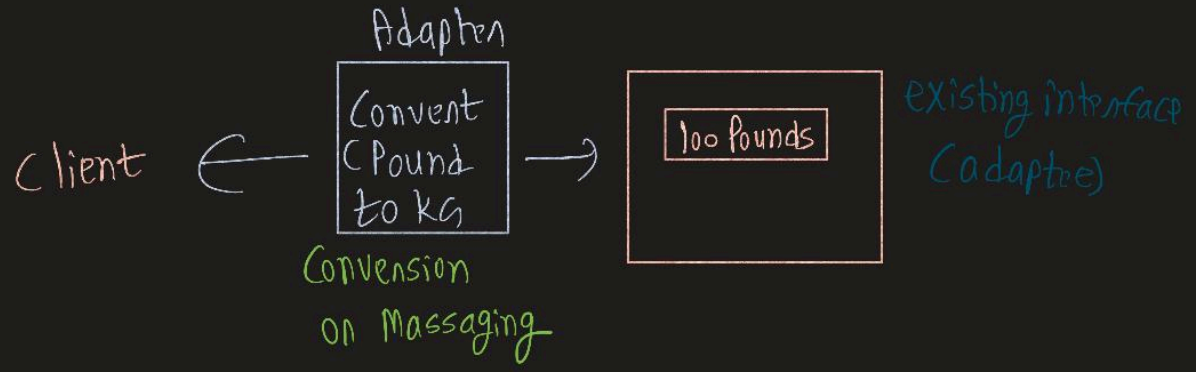
Object identification

* dice
* Snake, ladden
* board
* Players
* Cells

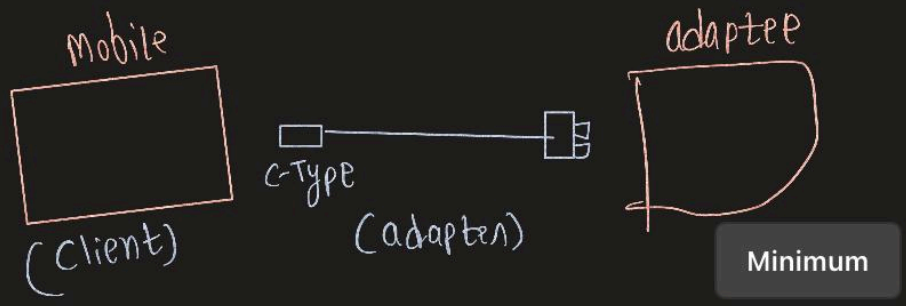
Minimum


6 Game
Board boards;
Dice dice;
Deque<Playrns> playerslist
Deque because
P1 P2 P1 P2
(remove from front & add to end, after playing)
5 Cell
Jump obj
Game has board
Game has playon
board has cell
4 Board
Cells[][] cells (2-D array)
Snake
int start
int end
(Top to bottom end)
ladder
int start
int end
(bottom to top)
instead made below
Game has Dice
Dice 2
int diceCount
rollDice()
Playrn 1
string id
int currentPosition
Minimum
Cell contains Jump
Jump 3
int start
int end
(Snake, end)
OR
Dice
Playrn
Snake
ladder
Game
List<Snake> SnakeList;
List<ladder> ladderList;
int boardstart=1
int boardend=100
has object ka from on to
Problem:
Job bhi ek new Position aayegi; We need to iterate to snake & laddr both

3 Wright Machine



Example :



3 - Type of Design Pattern

Clubbing together
of 2 or more objects
to create a biggin
Structure (to solve bigger Problem)