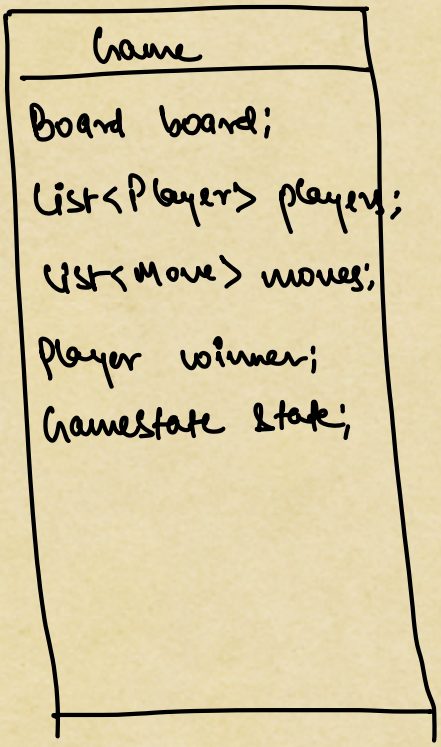
1) Class diagram
2) Design Pattern
3) Winning strategy O(1)
4) Undo Scenario.

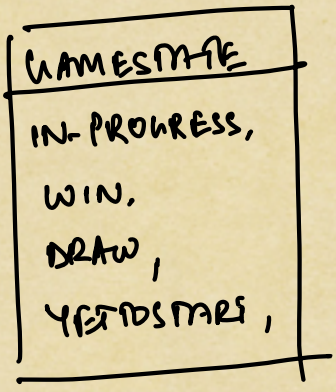
⇒ Class diagram

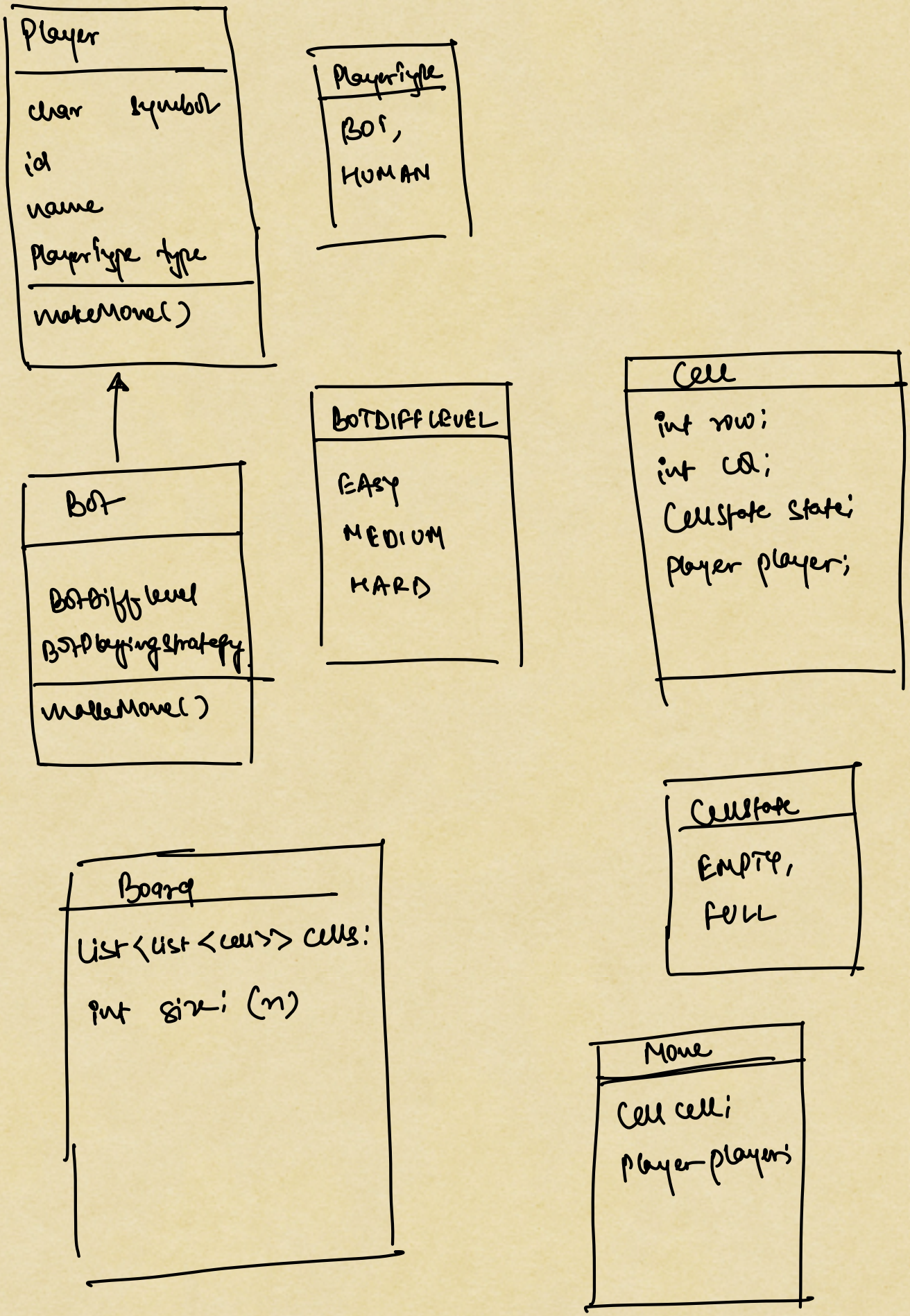
- Nouns
- Visualization of user journey

Classes

- Game
- Player
- Bot
- Board
- Cell
- Move
- Symbol

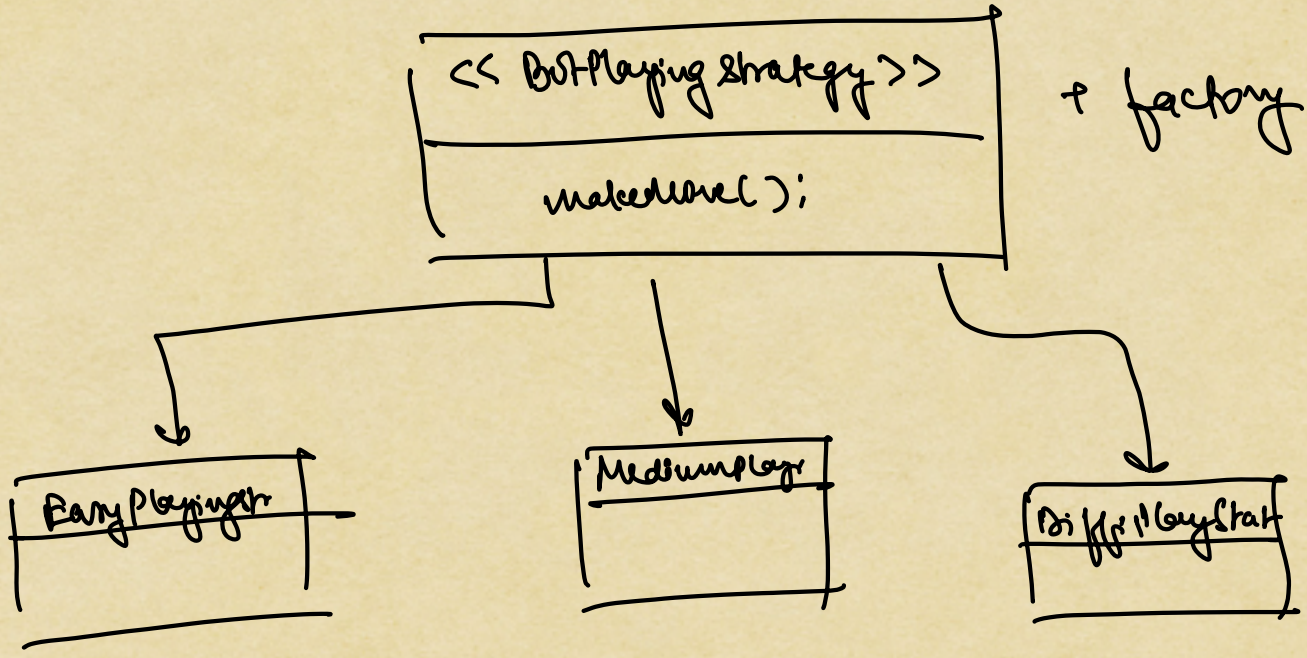





Player
char symbol
id
name
PlayerType type
makeMove()
PlayerType
BOT,
HUMAN
BOT
BOTdiffLevel
BOTplayingStrategy
makeMove()
BOTDIFFLEVEL
EASY
MEDIUM
HARD
Cell
int row;
int col;
CellState state;
Player player;
CellState
EMPTY,
FULL
Board
List<List<Cell>> cells;
int size; (n)
Move
Cell cell;
Player player;

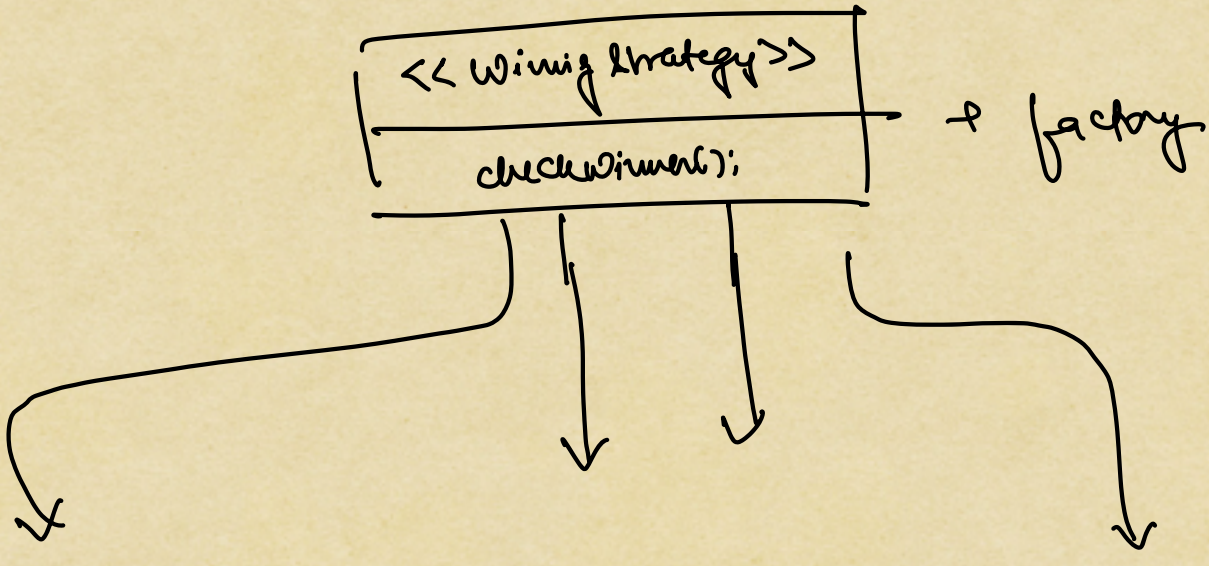
⇒ Design Pattern:

⇒ multiple ways to make move for BOT



⇒ Builder DP for game

⇒ we will ways to find the winner



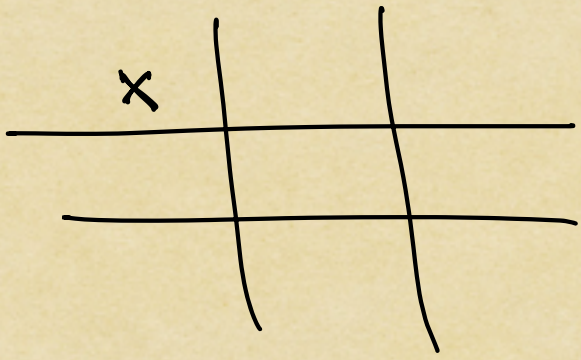
⇒ Builder

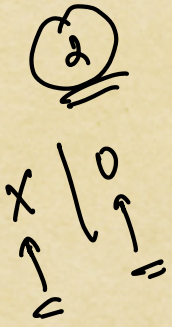
⇒ Strategy + factory

⇒ Check winner in O(1)

⇒ cont'd ⇒

Same symbol in a row
" " " a col
" " " " diagonal
Same symbol in all 4 corners





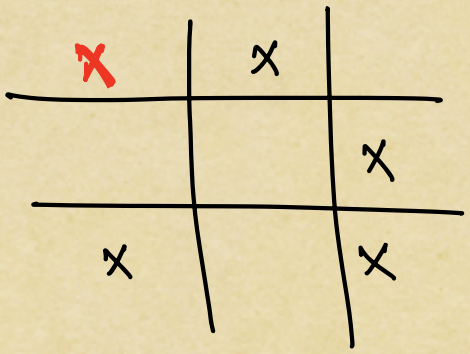
* for all symbols, check if anyone is the winner by iterating the entire board!

$(N-1) \times (N \times N) \Rightarrow O(N^3)$

↓

we dont need to check for all symbols, just check for most recently played symbol

$O(N^2)$



⇒ we only need to check for the row, col, or diag. corresponding to the last played cell

↓

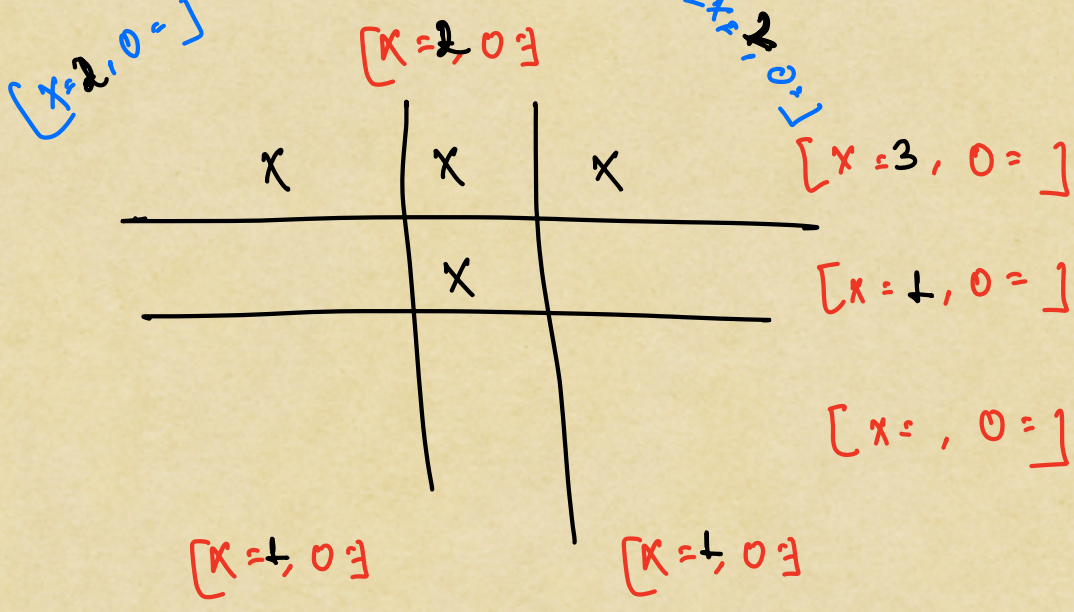
worst case ⇒ middle cell

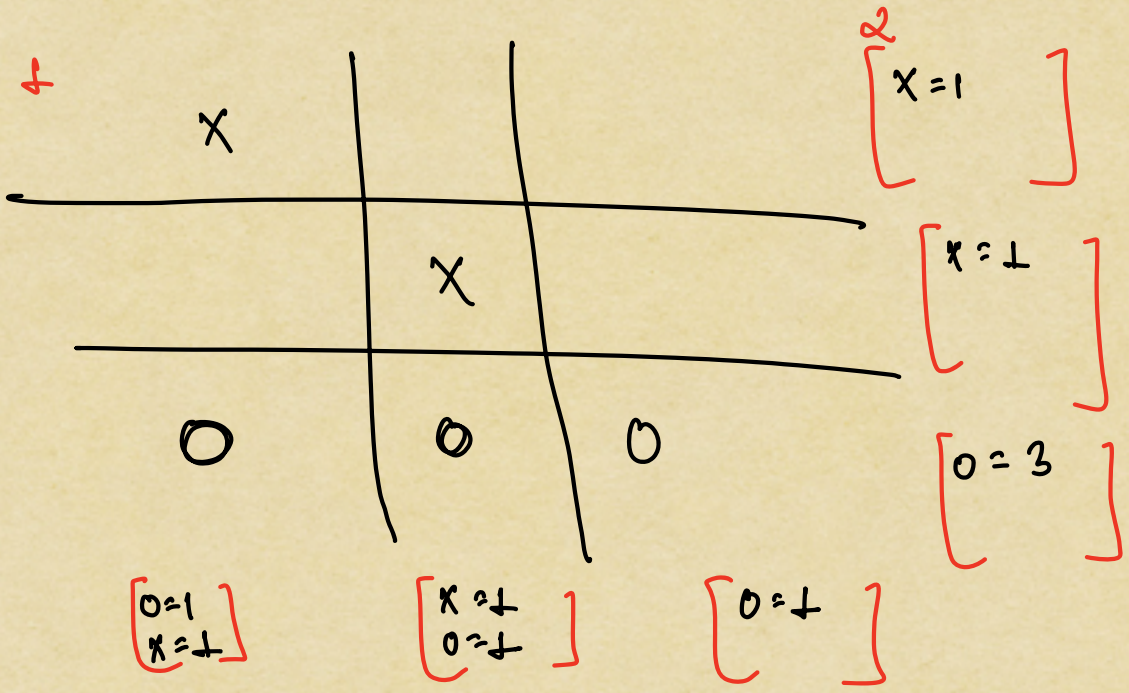
+ row → N
+ col → N
+ diag → N
+ diag → N

4N

TC ⇒ O(4N) ≤ O(N)

x,0





diag1 [X=2, O=1]

diag2 [X=1, O=1]

⇒ maintain hashmaps

⇒ all rows

⇒ all colums

⇒ 2 diagonals

3x3 ⇒ matrix

⇒ 3 rows → 3 hashmaps

3 col ⇒ 3 "

2 diag ⇒ 2 "

⇒ player plays a move:

(4)
- 1 ⇐ i) update & check row hashmap count
- 1 ⇐ ii) " col " "
- 2 ⇐ iii) " diag " "

10×10

check if at any place, count for that symbol == N

↓

TC ⇒ O(1)

SC ⇒ O(2N+2) ⇒ O(N) + (N-1)

⇒ O(N2)

⇒ How to implement UNDO:

3 ways
i) Kuch Kuch Hota Hai
ii) Om Shanti Om
iii) Doraemon

i) Kuch Kuch Hota Hai

LastMove ⇒ [[0,0 X] [0,1 O] [0,2 X] ...]

Undo

i) delete the move the list

ii) remove the last change from the board.

ii) Our thanks Our

List<Moves> = [[0,0 X] [0,1 O] [0,2 X] [1,0 X] [1,1 O]]

i) everytime we do undo,
we clean the entire board

ii) replay all the remaining moves

if only redone + Step ⇒ high π

" " multiple steps ⇒ ✗

iii) Doraemon approach

time machine (anywhere gate

List<Move> ⇒ [[0,0 X], [0,1 O], [0,2 X], [1,1 O] ..]

List<Board> ⇒ [X | | , X | O | , X | O | X , X | O | X / O | , ...]


Current
Current