LET'S PLAY CHECKERS

ABOUT US

MARTIN PALANJYAN, ARMAN KHACHATRYAN, GOR HOVAKIMYAN

CHECKERS

Checkers, also called draughts, board game, one of the world's oldest games. Checkers is played by two persons who oppose each other across a board of 64 light and dark squares, the same as a chessboard.

FMJD

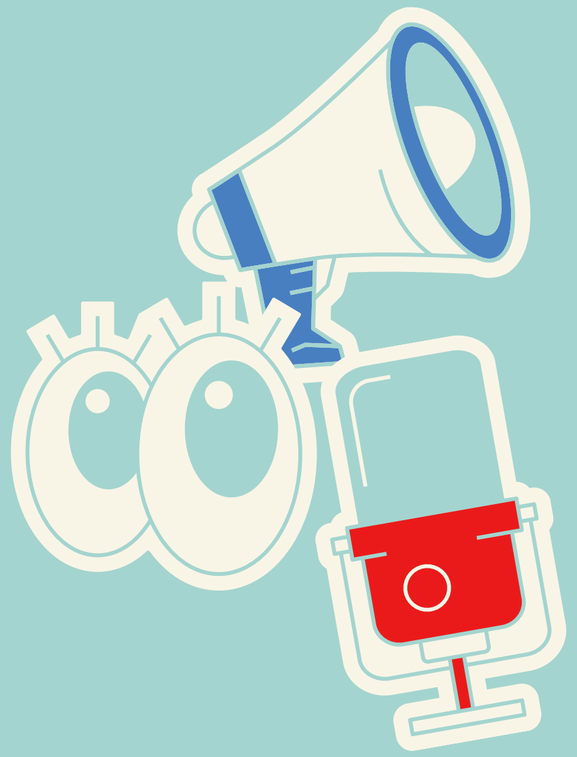
Rules of the game:

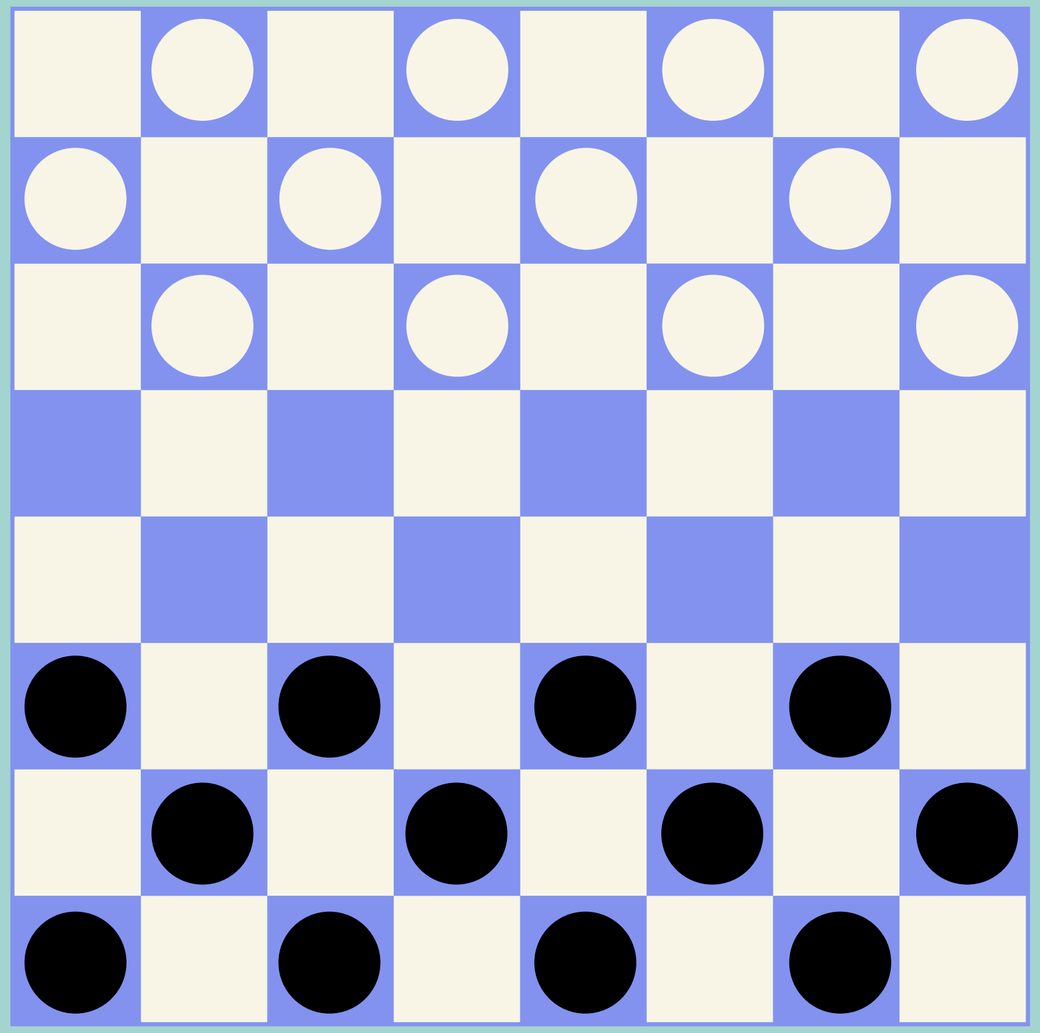
The first move of the turn should be a forward move unless the piece is a king.

Multiple captures are allowed.

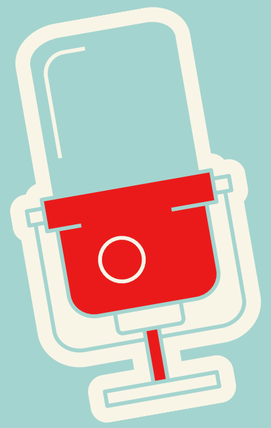
Backwords capture is allowed only from the second capture.

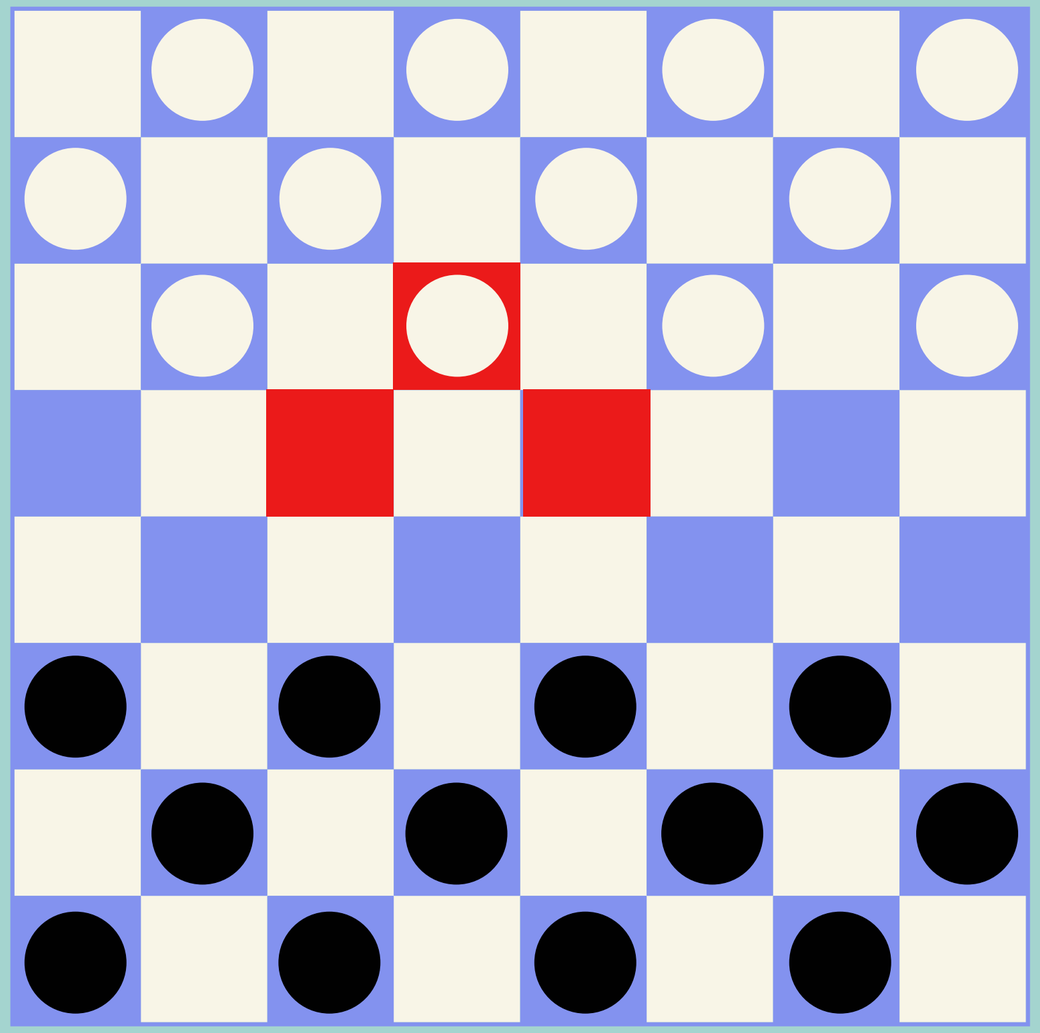
Not capturing a piece is allowed.



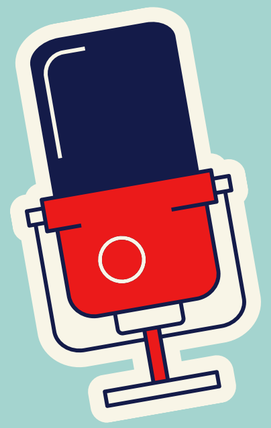


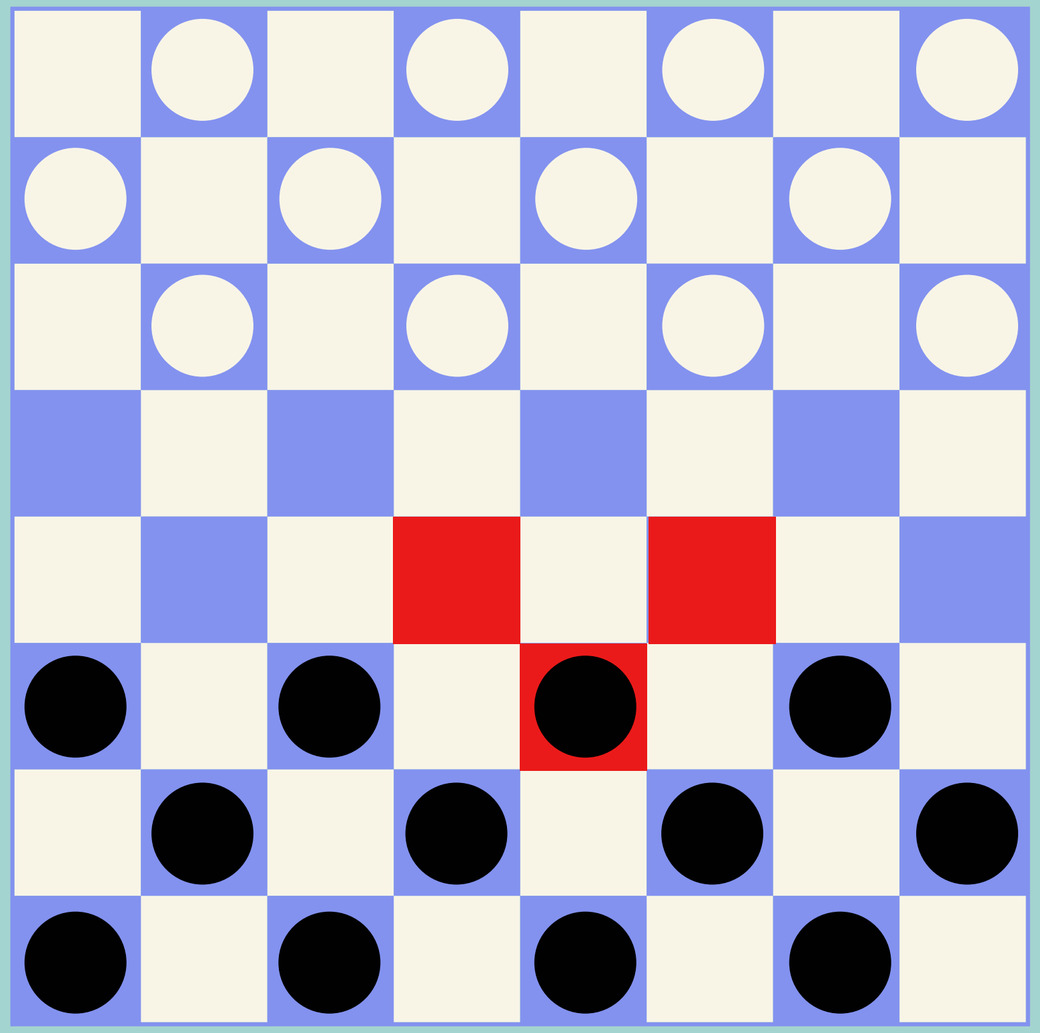
Default arrangement of the board.



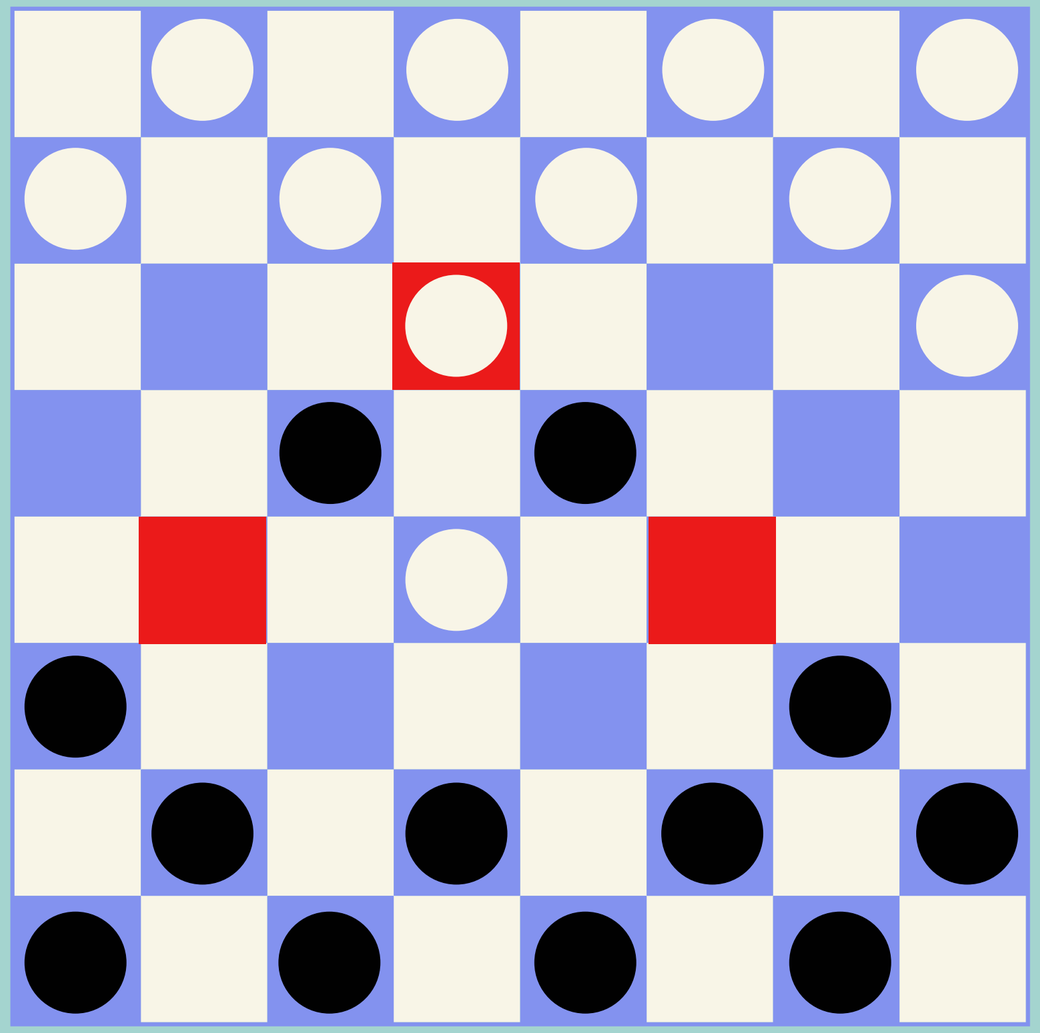


Moves of a white "Man".

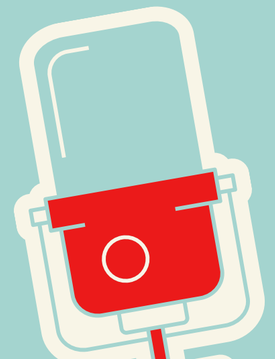


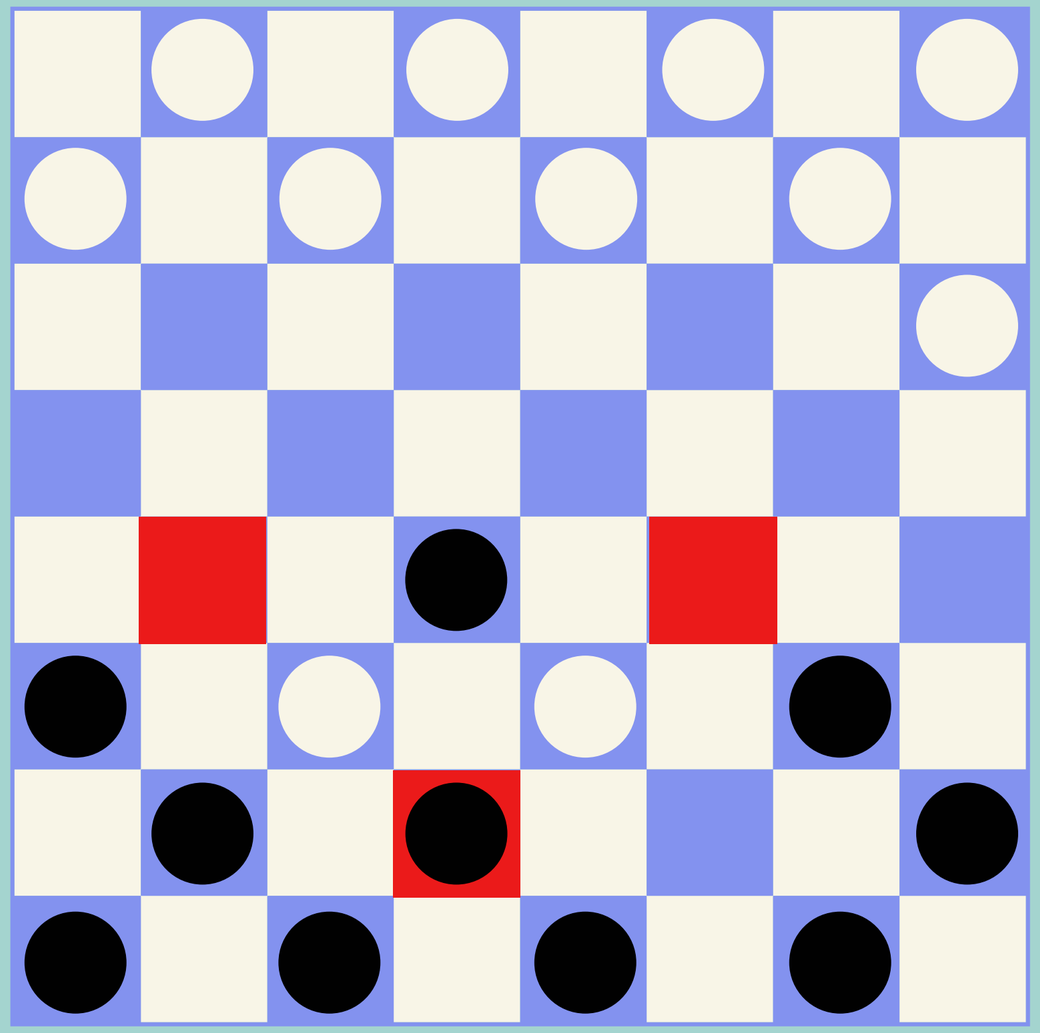


Moves of a black "Man".

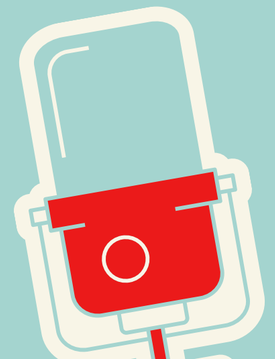


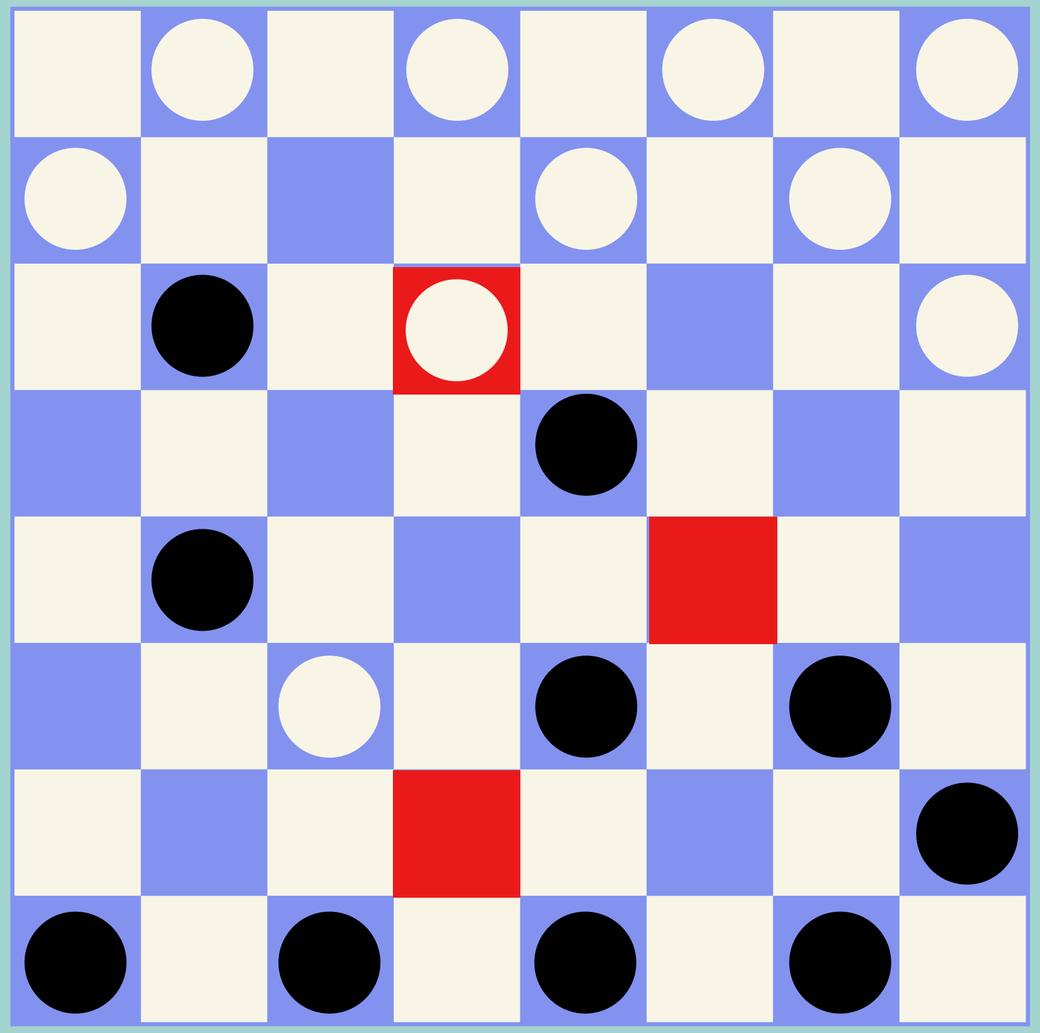
Capture of a white "man".



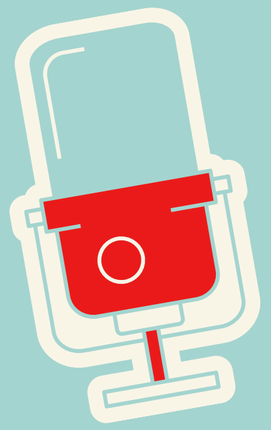


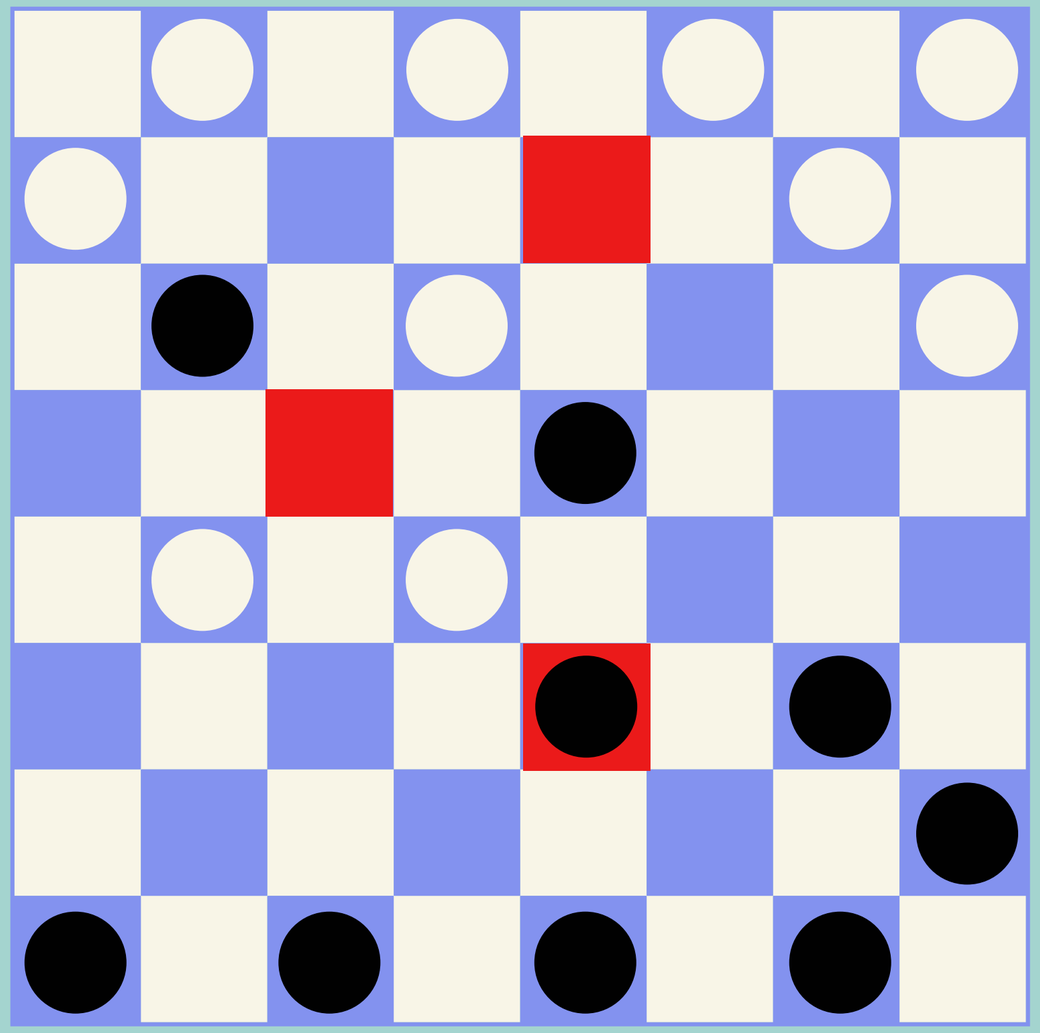
Capture of a black "man".



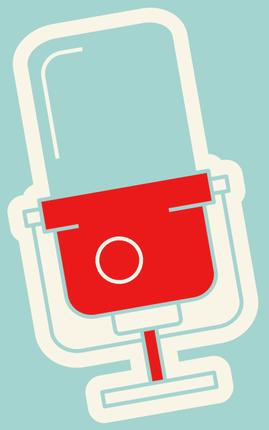


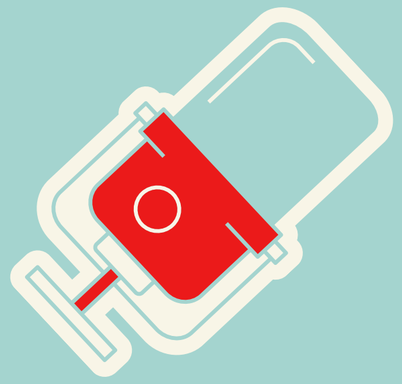
Multiple captures of white piece.

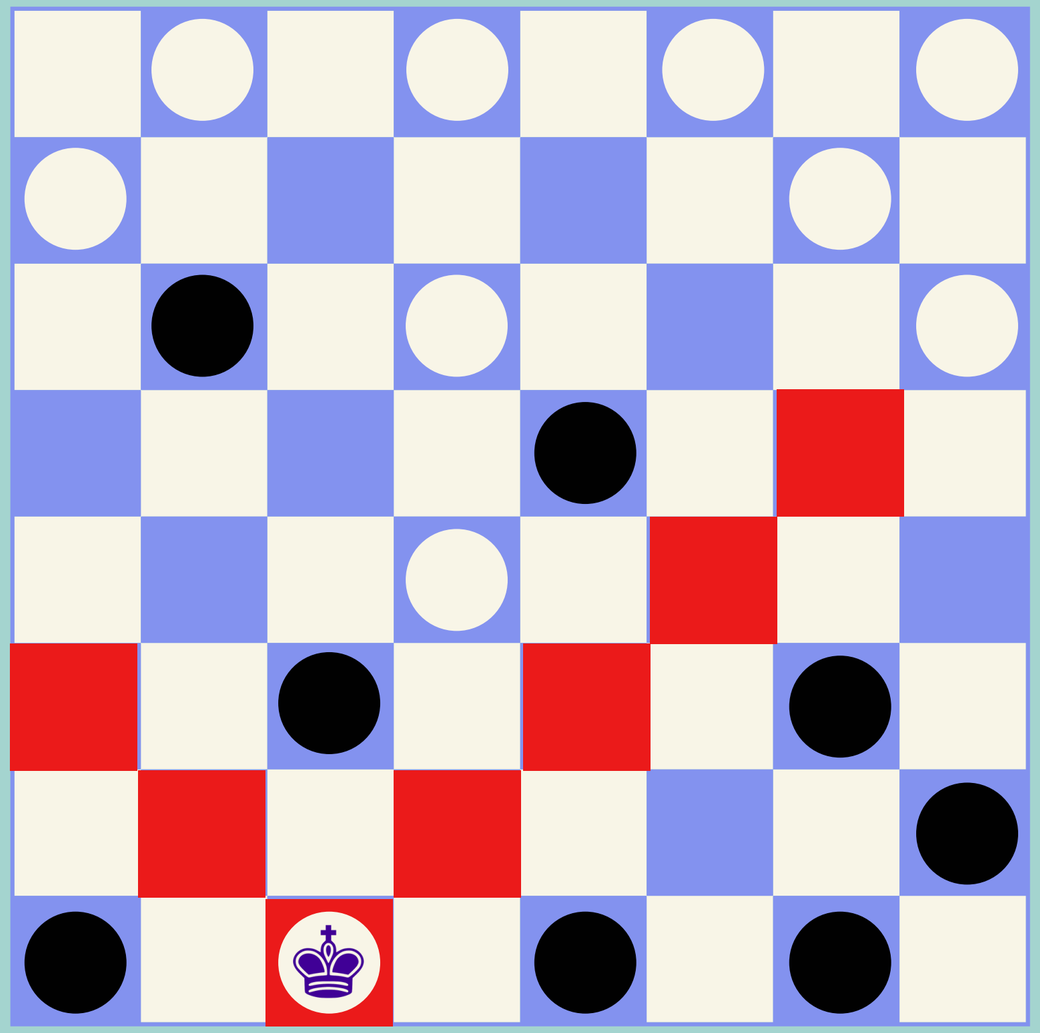




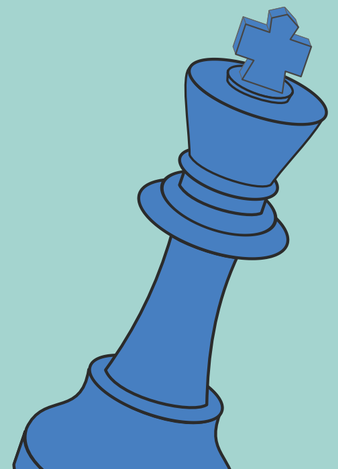
Multiple capture of black piece.

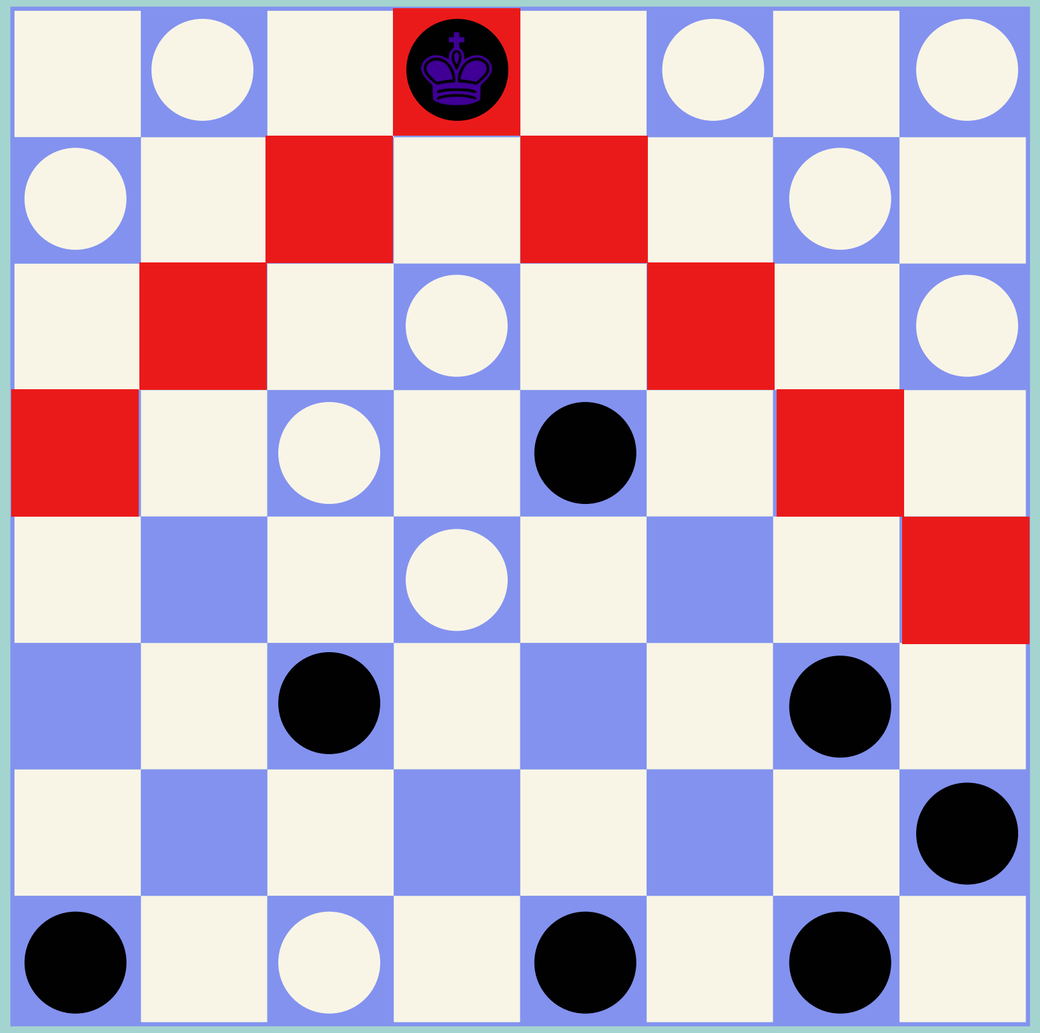




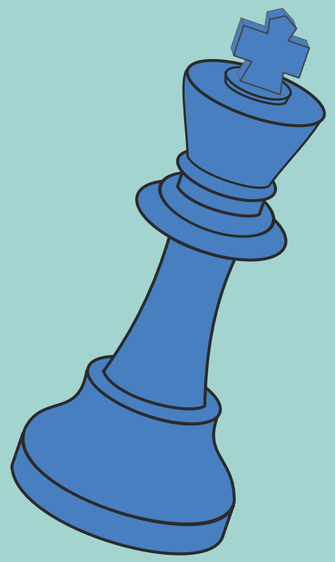


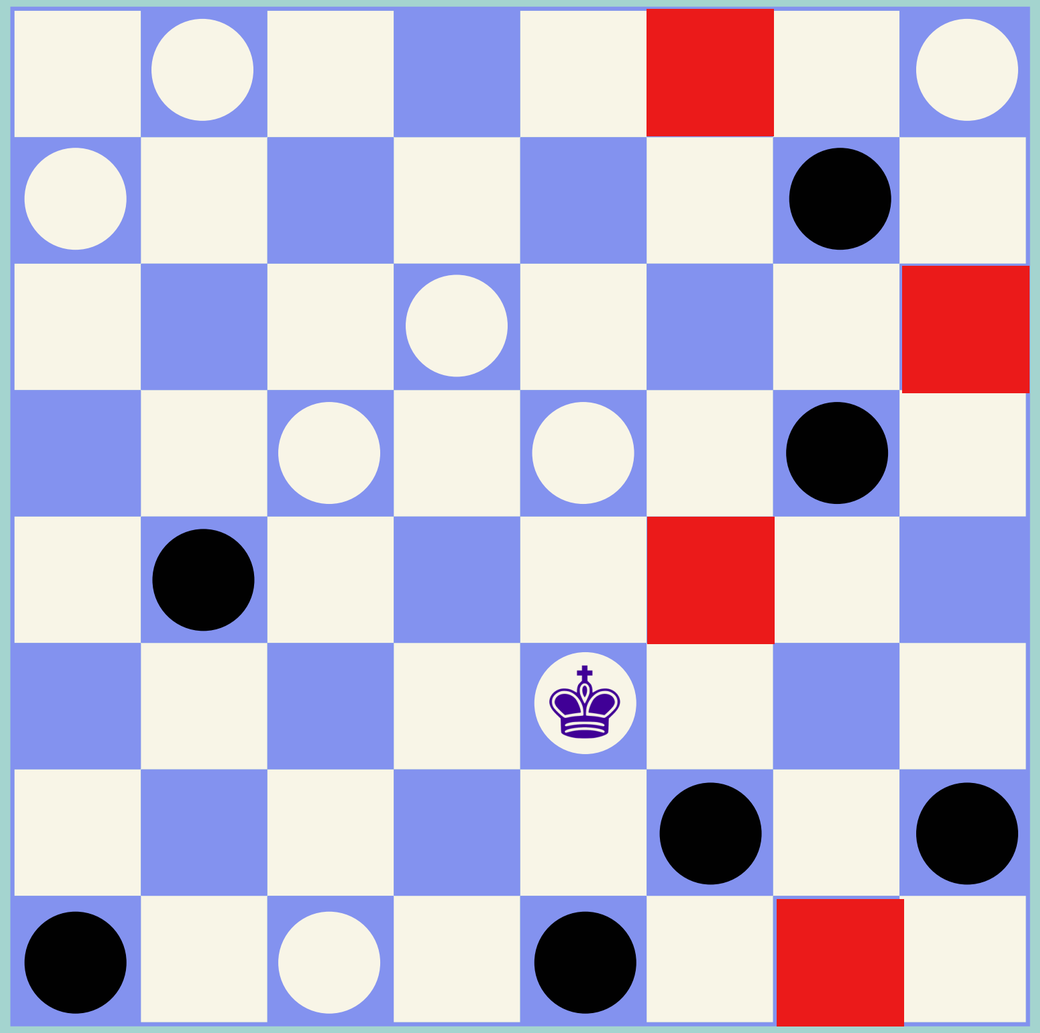
Moves of white "King".



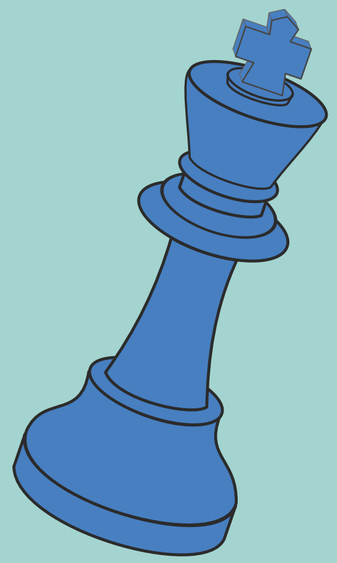


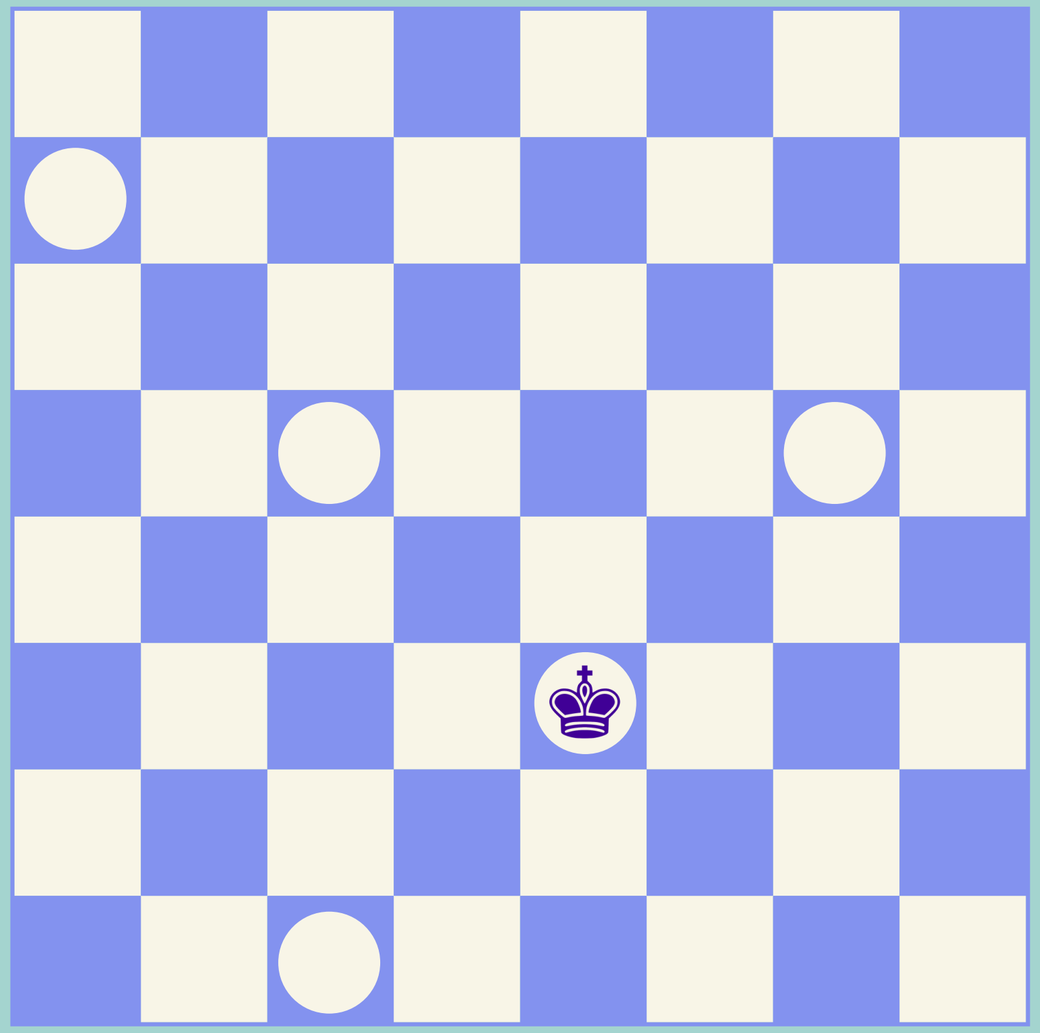
Moves of black "King".



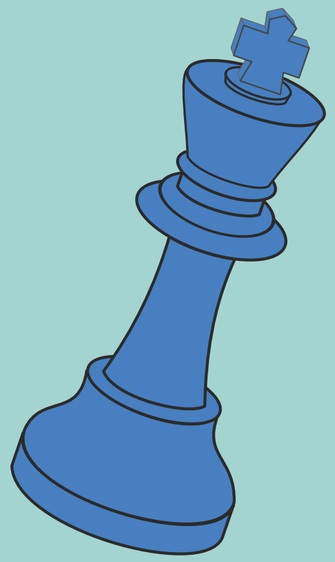


Captures of Kings.





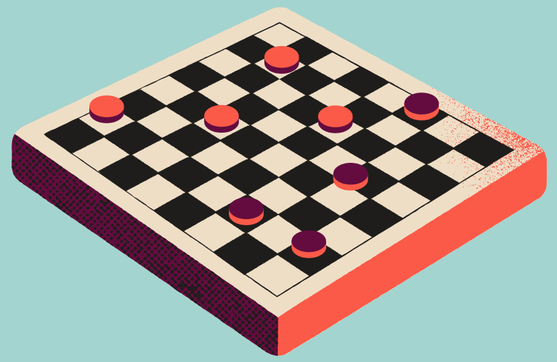
End of the game.
(white victory)



UML

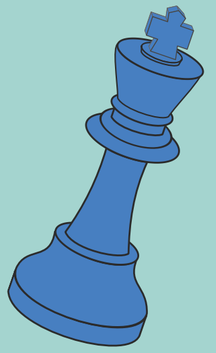
CORE

CLASSES



MOVE

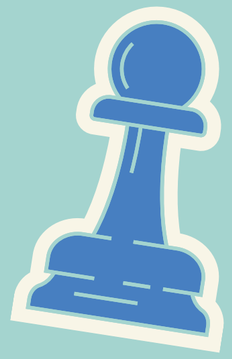
KING



POSITION

CHECKERS

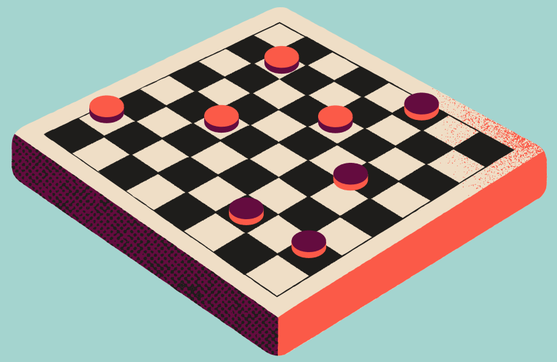
MAN



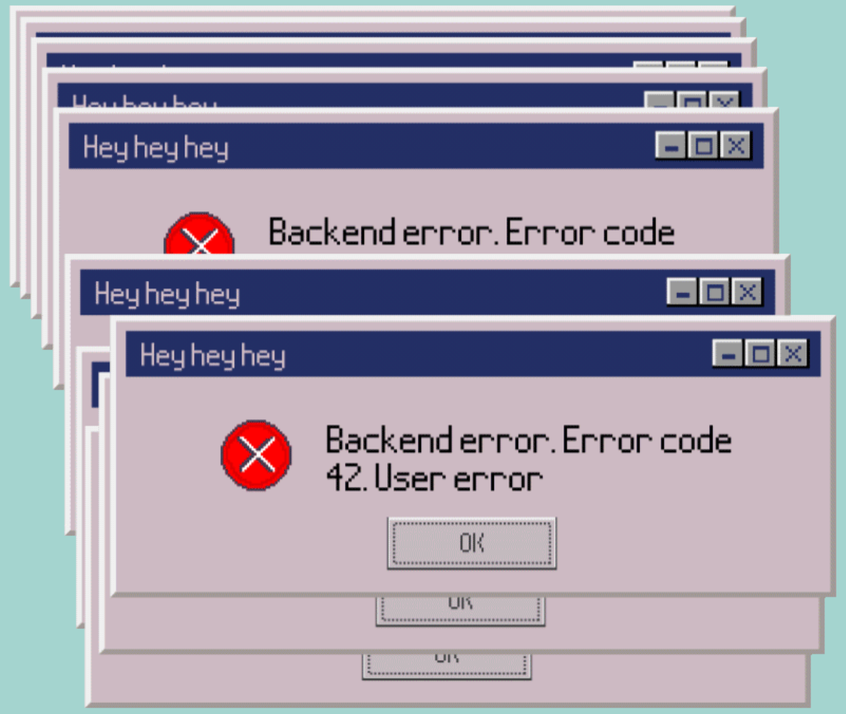
PIECE

CLI

CLASSES

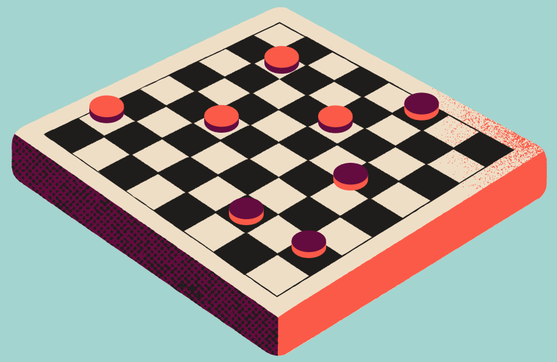


CHECKERSCONSOLE

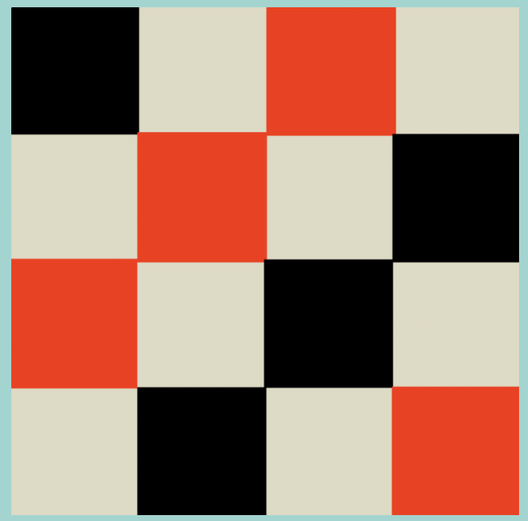


UI

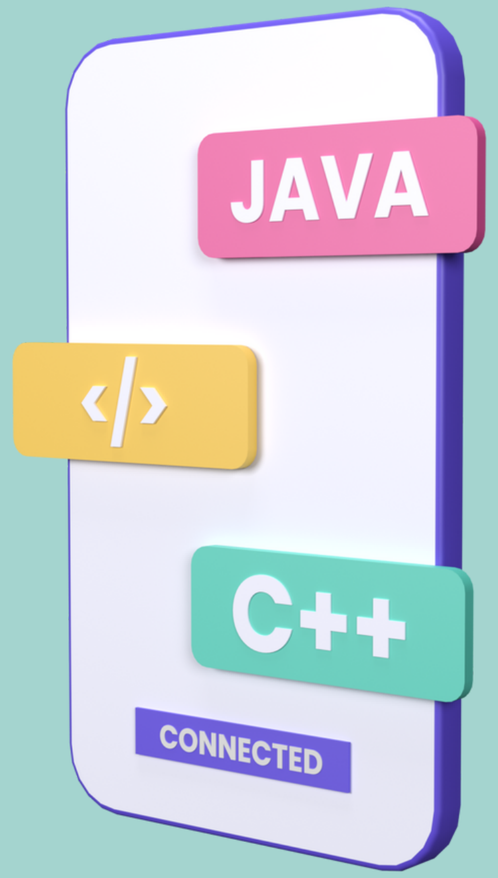
CLASSES

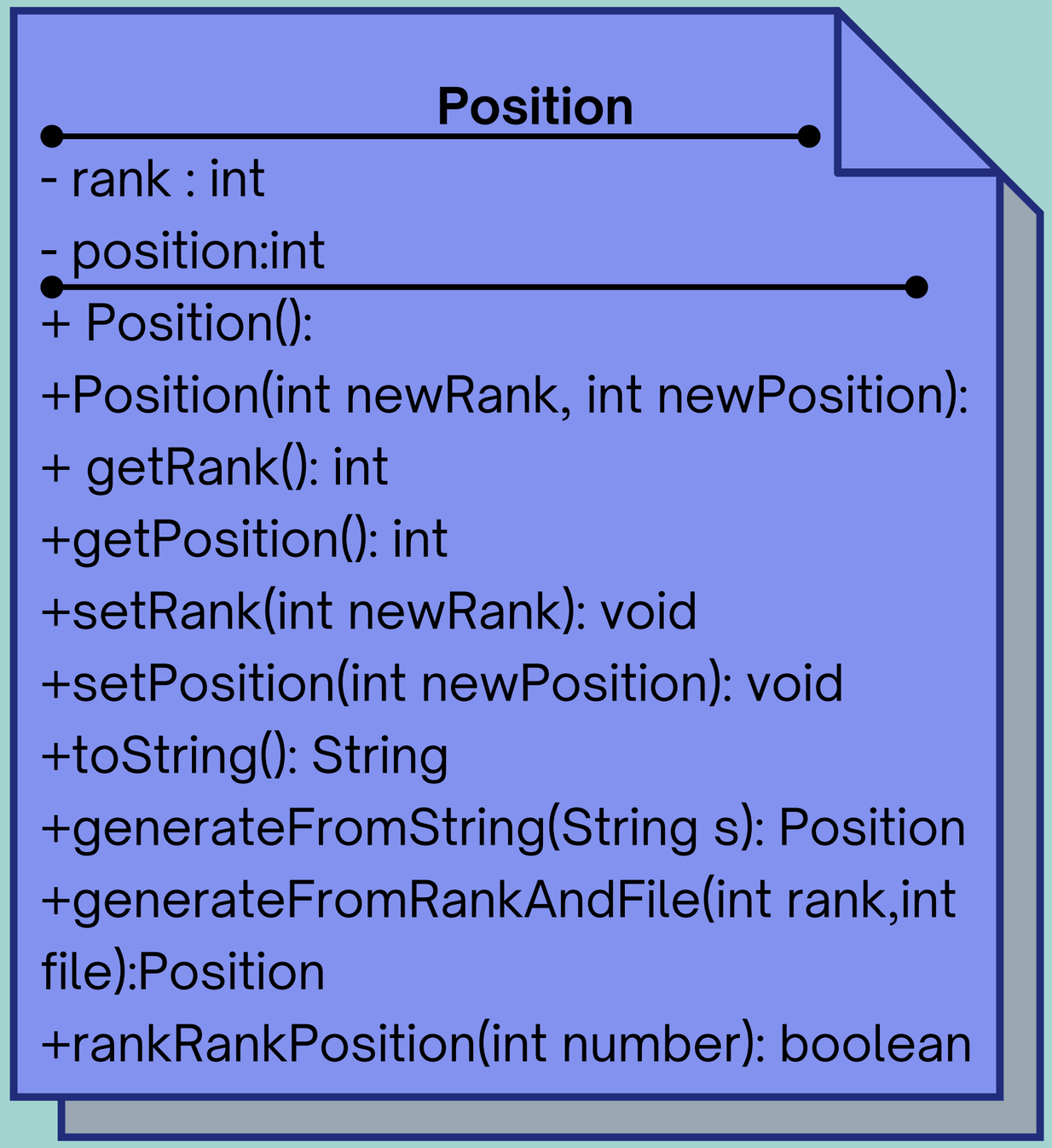


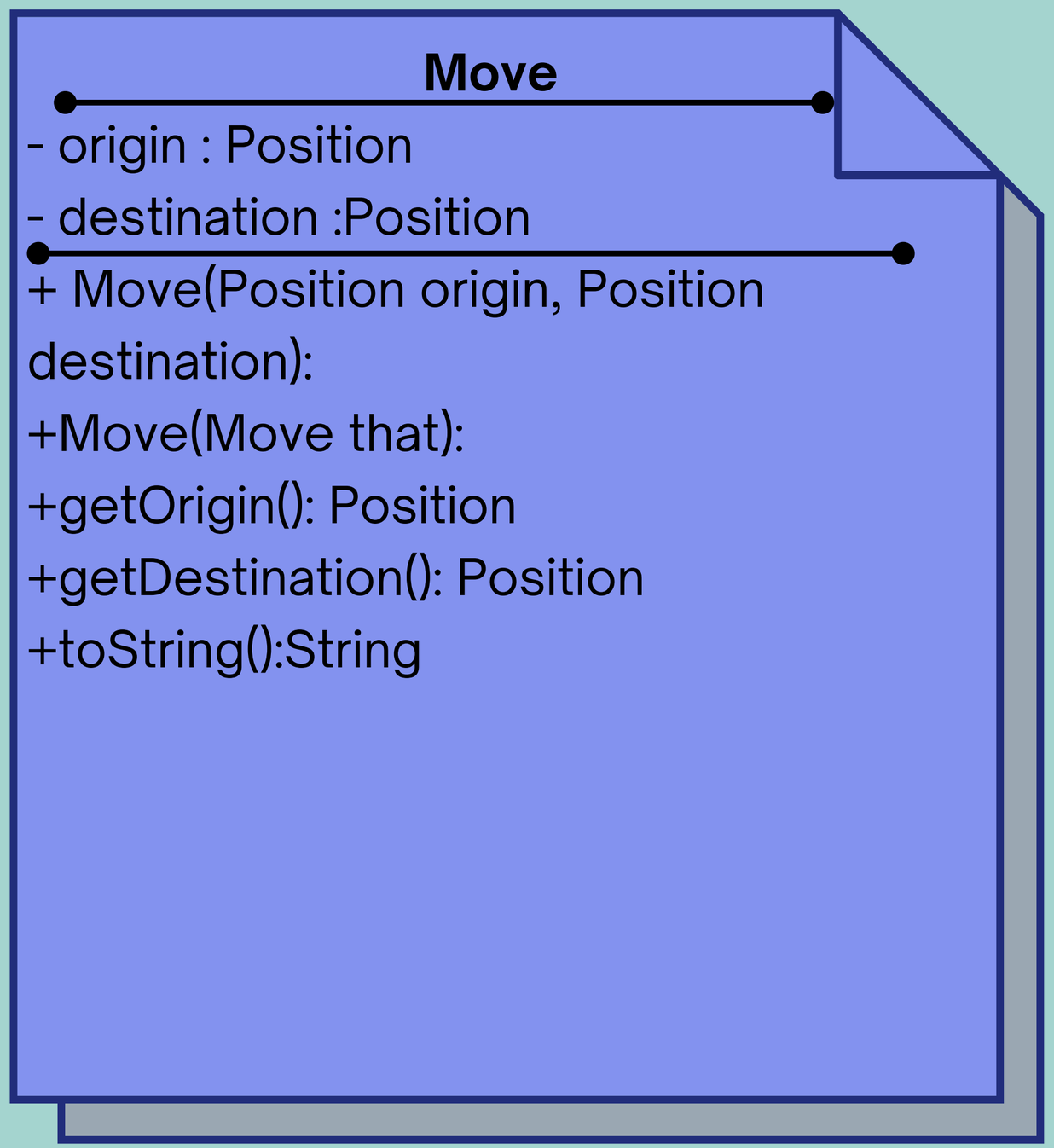
BOARDSQUARE

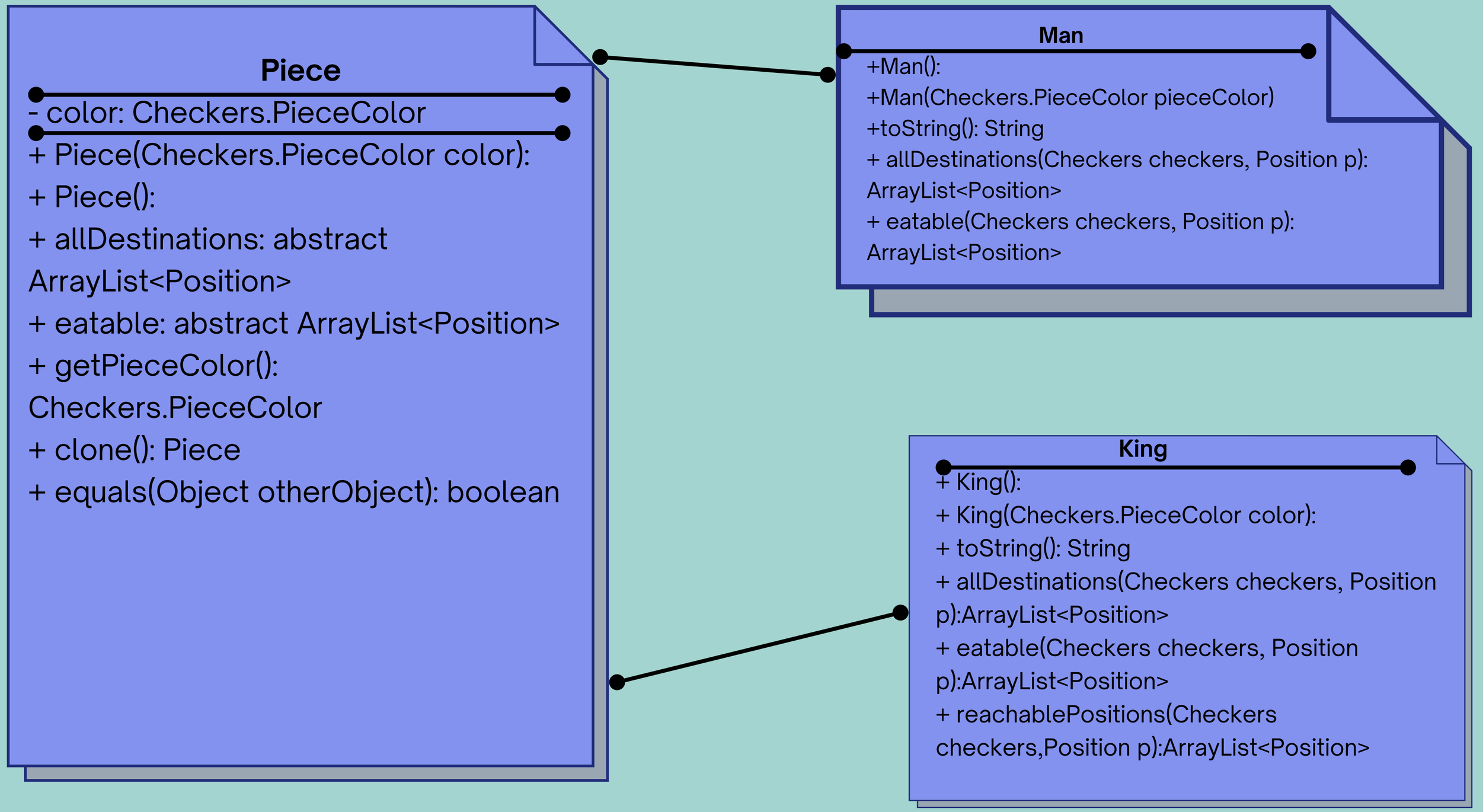


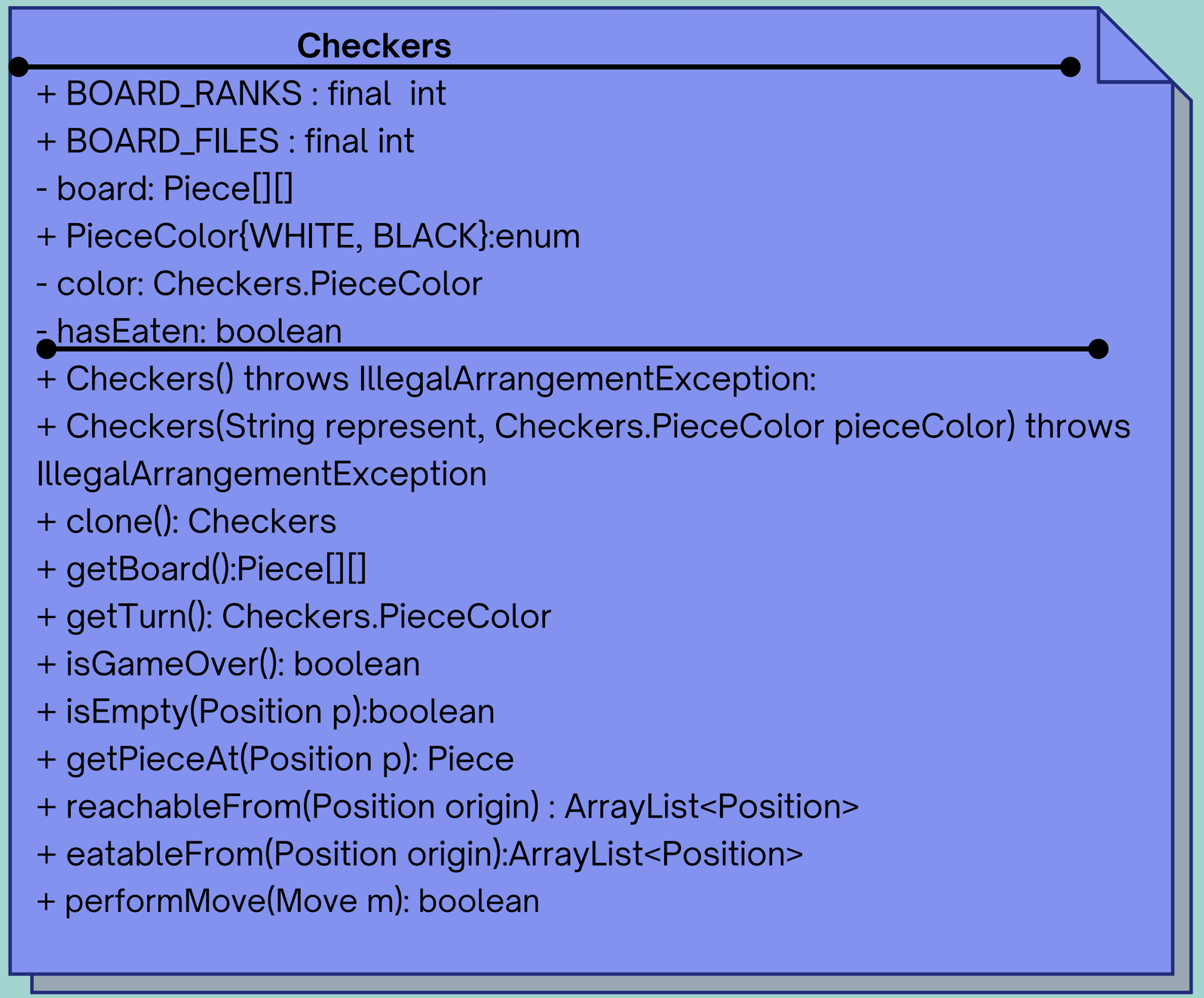
CHECKERSUI

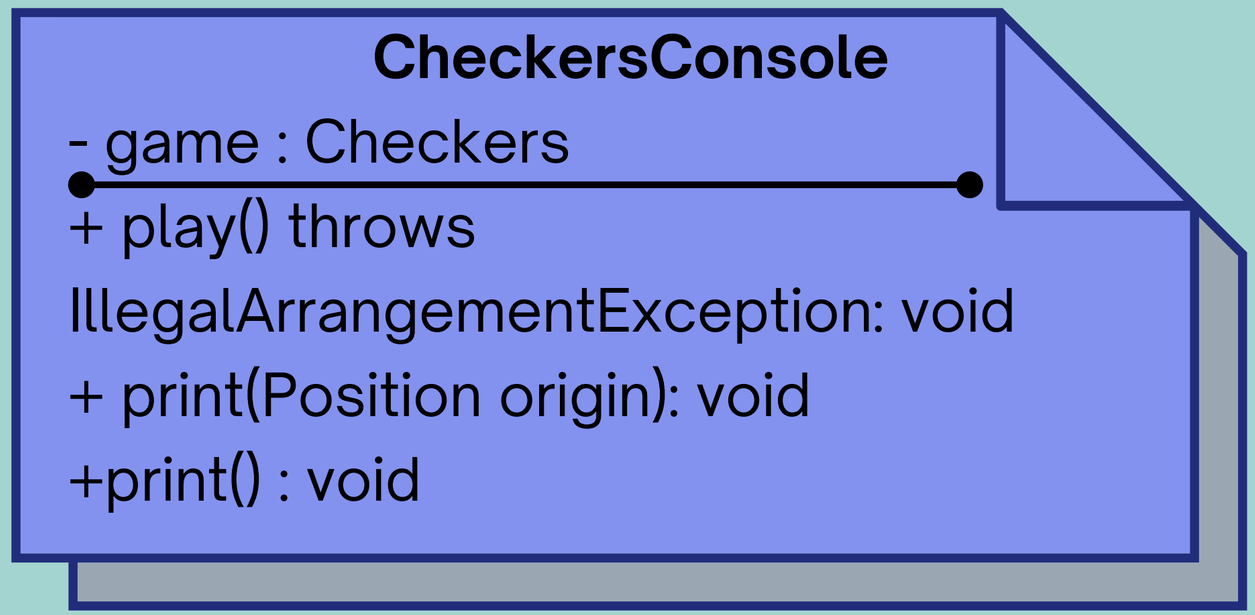


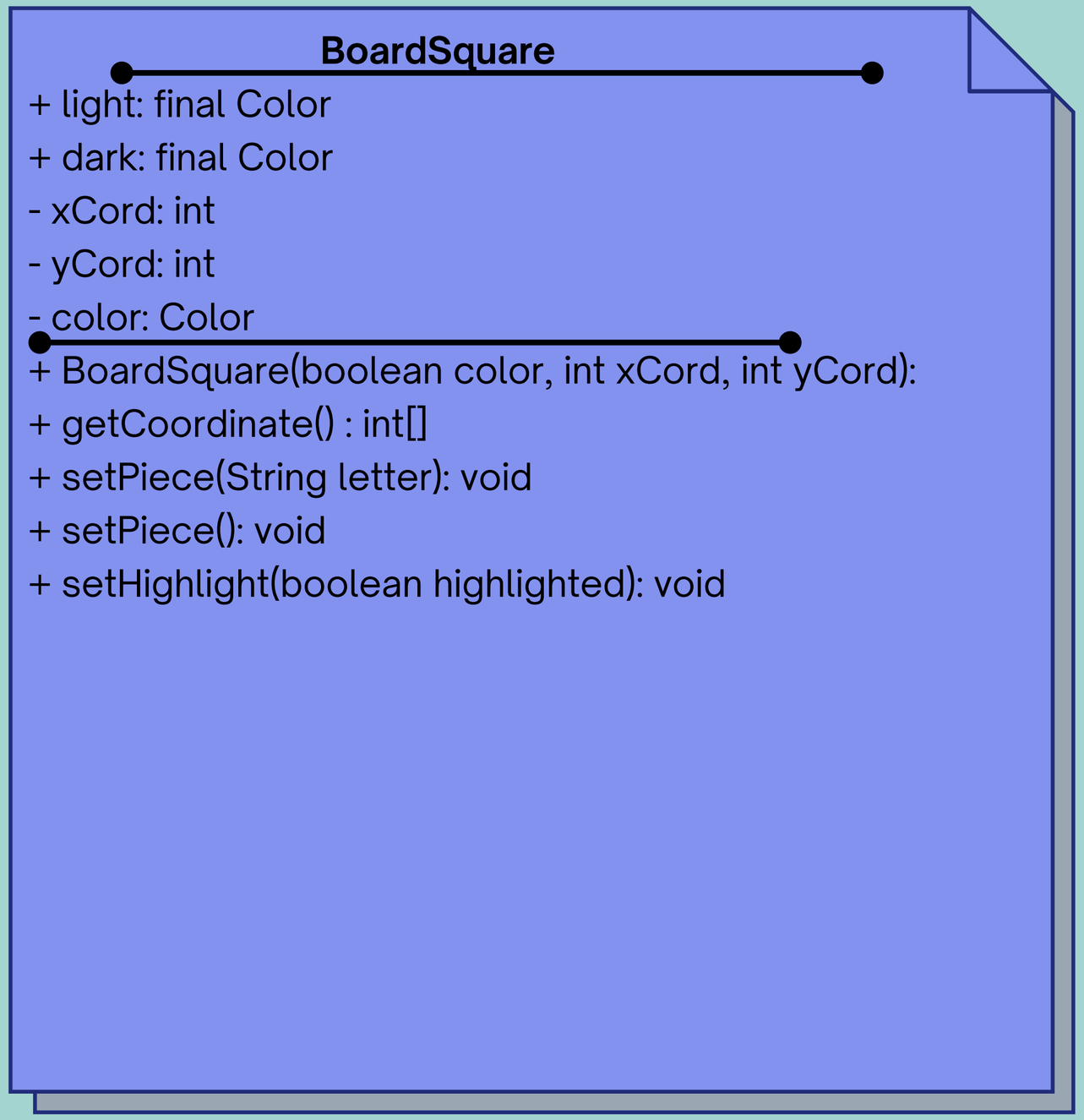

Position
- rank : int
- position:int
+ Position():
+Position(int newRank, int newPosition):
+ getRank(): int
+getPosition(): int
+setRank(int newRank): void
+setPosition(int newPosition): void
+toString(): String
+generateFromString(String s): Position
+generateFromRankAndFile(int rank,int file):Position
+rankRankPosition(int number): boolean

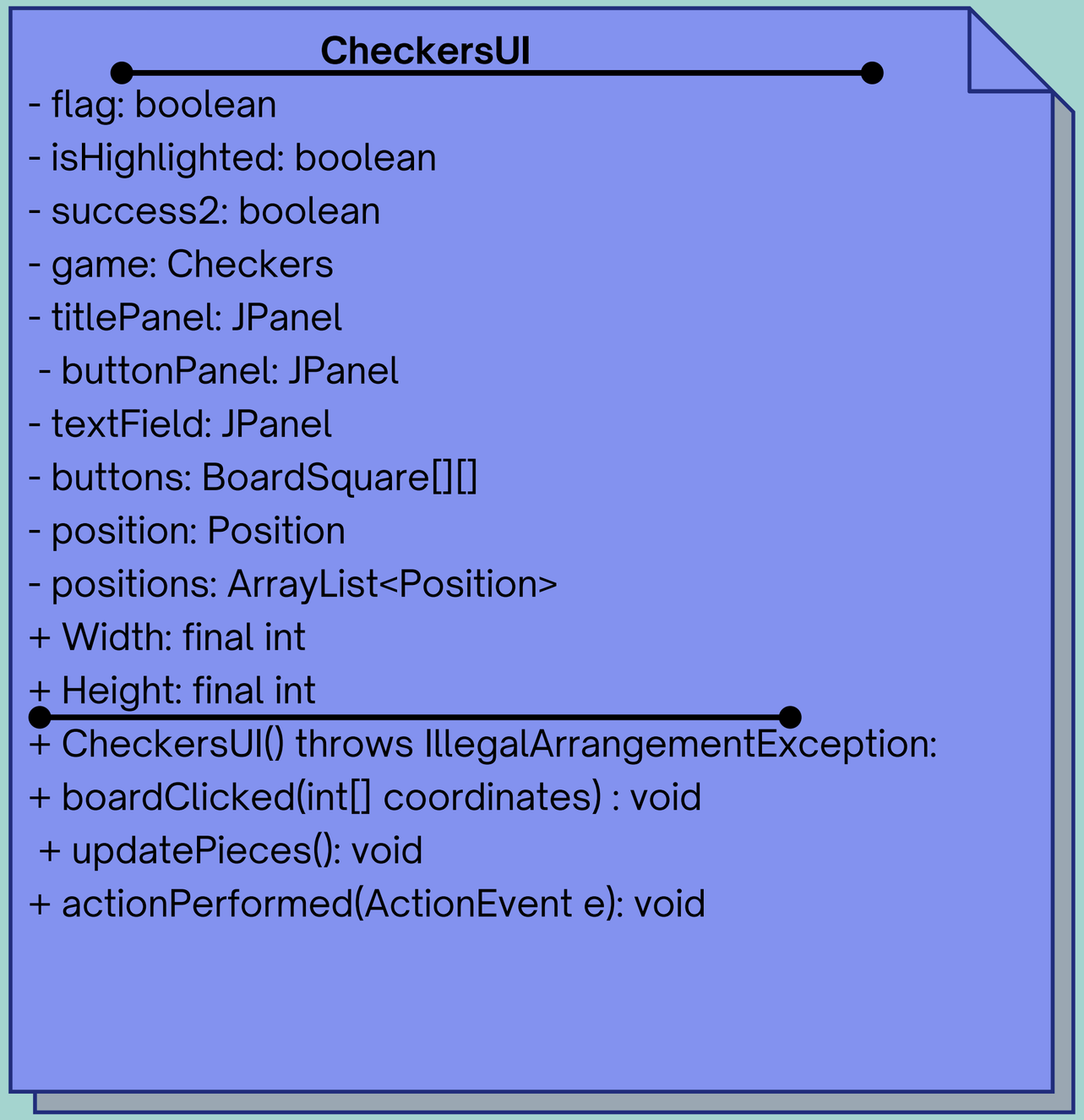

Move
- origin : Position
- destination :Position
+ Move(Position origin, Position destination):
+Move(Move that):
+getOrigin(): Position
+getDestination(): Position
+toString():String

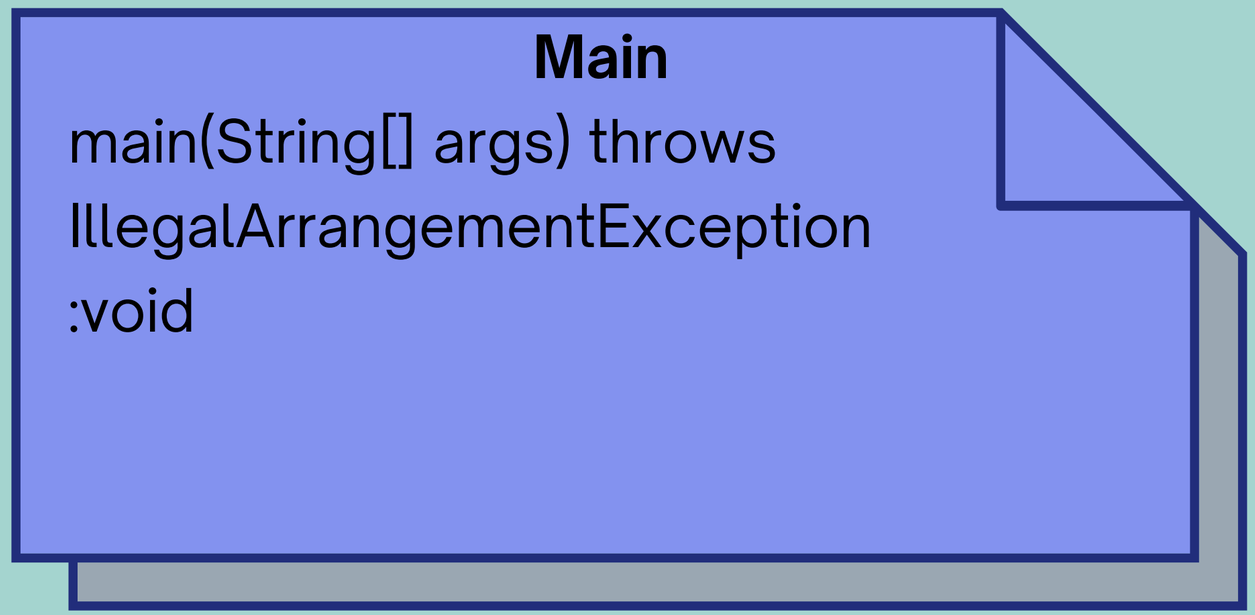

Piece
- color: Checkers.PieceColor
+ Piece(Checkers.PieceColor color):
+ Piece():
+ allDestinations: abstract ArrayList<Position>
+ eatable: abstract ArrayList<Position>
+ getPieceColor(): Checkers.PieceColor
+ clone(): Piece
+ equals(Object otherObject): boolean
Man
+Man():
+Man(Checkers.PieceColor pieceColor)
+toString(): String
+ allDestinations(Checkers checkers, Position p): ArrayList<Position>
+ eatable(Checkers checkers, Position p): ArrayList<Position>
King
+ King():
+ King(Checkers.PieceColor color):
+ toString(): String
+ allDestinations(Checkers checkers, Position p):ArrayList<Position>
+ eatable(Checkers checkers, Position p):ArrayList<Position>
+ reachablePositions(Checkers checkers,Position p):ArrayList<Position>


Checkers
+ BOARD_RANKS : final int
+ BOARD_FILES : final int
- board: Piece[][]
+ PieceColor{WHITE, BLACK}:enum
- color: Checkers.PieceColor
- hasEaten: boolean
+ Checkers() throws IllegalArrangementException:
+ Checkers(String represent, Checkers.PieceColor pieceColor) throws IllegalArrangementException
+ clone(): Checkers
+ getBoard():Piece[][]
+ getTurn(): Checkers.PieceColor
+ isGameOver(): boolean
+ isEmpty(Position p):boolean
+ getPieceAt(Position p): Piece
+ reachableFrom(Position origin) : ArrayList<Position>
+ eatableFrom(Position origin):ArrayList<Position>
+ performMove(Move m): boolean

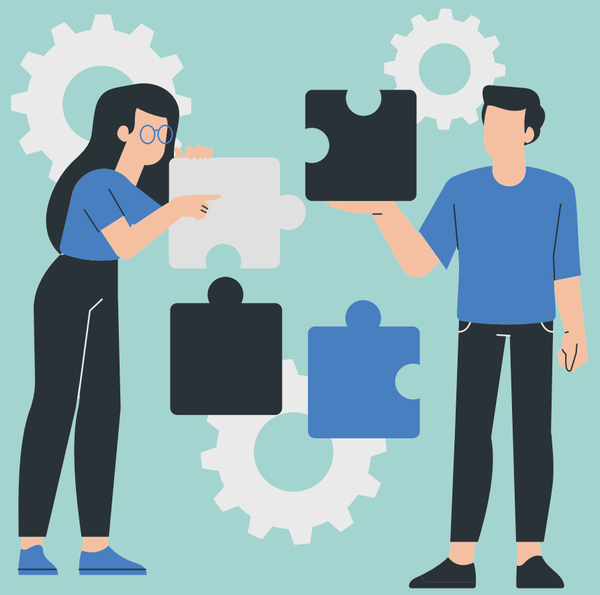

CheckersConsole
- game : Checkers
+ play() throws
IllegalArrangementException: void
+ print(Position origin): void
+print() : void


BoardSquare
+ light: final Color
+ dark: final Color
- xCord: int
- yCord: int
- color: Color
+ BoardSquare(boolean color, int xCord, int yCord):
+ getCoordinate() : int[]
+ setPiece(String letter): void
+ setPiece(): void
+ setHighlight(boolean highlighted): void


CheckersUI
- flag: boolean
- isHighlighted: boolean
- success2: boolean
- game: Checkers
- titlePanel: JPanel
- buttonPanel: JPanel
- textField: JPanel
- buttons: BoardSquare[][]
- position: Position
- positions: ArrayList<Position>
+ Width: final int
+ Height: final int
+ CheckersUI() throws IllegalArrangementException:
+ boardClicked(int[] coordinates) : void
+ updatePieces(): void
+ actionPerformed(ActionEvent e): void

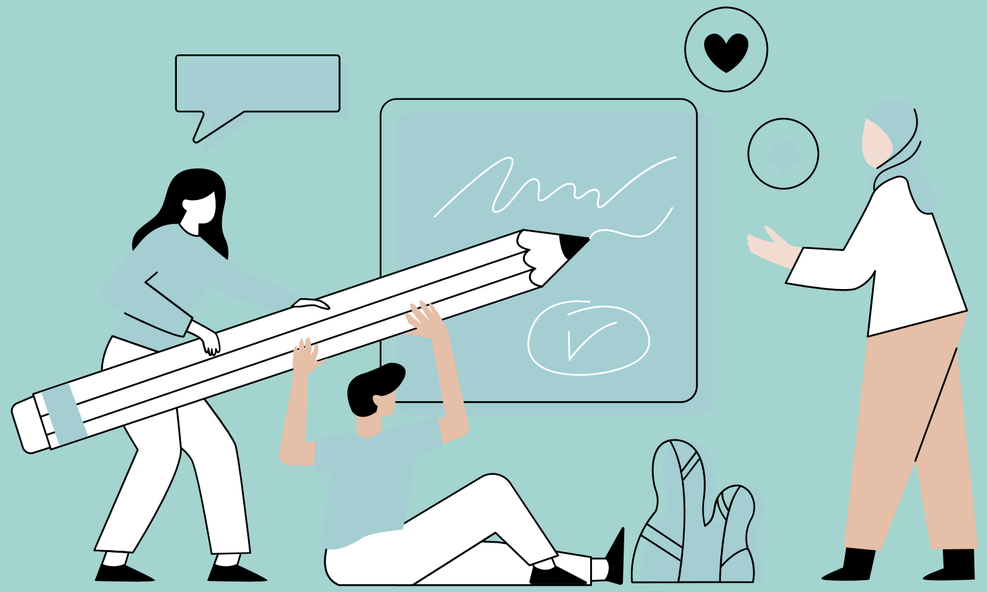

Main
main(String[] args) throws
IllegalArrangementException
:void

Problems we faced!



1. PERFORMMOVE IN CHECKERS
2. GUI

Our Future Plans



THANK YOU FOR PLAYING!