ANALYSIS REPORT

MOULILA YOUNES, NASSER ABATTOUY

GROUP D221

SCHOOL YEAR : 2021 – 2022

PIECE :

We have chosen to create a class named Piece in order to represent several kinds of piece of the game. In order to do that, we assume that a piece has:

Attributes

-**TYPE :** is an enumeration set because there are different kind of pieces and each one are ranked according to the rules. The type is defined by an enumeration « TypePiece ».

-**COLOR** : represent the color of the piece which is also the color of the player ,so that if one of them want to move a piece, he/she needs to check if the color is equal to their own color. The color is define by an enumeration « Color ».

-**VISIBLE** : this is an attribute set in order to know if a Piece is visible or not while playing.

-**HISTORICAL** : is an attribute written in order to get 5 movements of the piece so that we are able to make the « 3 same round trip » forbidden.

Methods

-**THE GETTERS AND SETTERS**.

-**ISHIGHERVALUE** : This is a method used to attack, it allow the user to know which piece is the stronger one according to the value.

-**ISMYCOLOR** : This method allow the user to know if the piece has the same color as the player in order to know what are his right : moving, attacking and so one...

-**ADDMOVE**: addMove gives the right to add a movement in the background of the moves of a piece.

HistoricalMove

We have chosen to make this class in order to save the fives lasts movement of a piece with the method addmove. And to prevent from making more than 2 round trips.

Thus, an historical move is made with moves objects that his also made of two Positions objects.

Attributes

-HISTORICALMOVE[5] : Is an array of 5 movement because we don't need more to check how many round trip we do.

-**LENGHT :** This attribute is set as a boundary indicator of the array

Methods

- **THE GETTERS AND SETTERS.**

-**ADDMOVE**: add the movement that the function received in parameter in the array.

SQUARE

This class is useful in order to know how to define a case of our boardgame. There are two kinds: water or field. Because we are not allowed to walk through the water.

Attributes

-**PIECE**:

is an optional type « optional<Piece> » because a square could be empty or could contains a piece.

-**TYPE**:

Type shows which kind of terrain we got field or water so that the player knows if he/she is allowed to make a specific movement.

Methods

- **THE GETTERS AND SETTERS.**

-**ISHIGHERVALUE**: This method calls the other one from the class piece who has the same name.

ISPIECECOLOR: This one call the method isMyColor from the Piece class.

HASSAMERANK: This function calls the method from the Piece class which has the same name.

BOARD

We have chosen to implement this class because it was necessary. Because we need an array of 10X10 which contains ours squares. The behaviors from the functions of this class modify the 10x10 array.

Attributes

-**BOARD [10][10]** :

This attribute is in summary the boardgame. In this object we are going to instantiate the squares objects which have also the pieces (or not)

Methods

-**MOVE**: is called when we tend to move a piece from a position to another one so that we can swap the complete squares objects.

-**CHECKPOSITIONISINBOARD**: this method checks if we've given a valid position.

-**ISQUAREEMPTY:** Checks if the square is empty. This is going to be used before a movement.

-**ISSAMECOLOR**: this method checks if the piece has the same color as the current player in order to know if he/she has the right to move it or not.

-**CHECKWINER**: this method checks if a player won and return his color.

-**ATTACK**: this method is called while moving when it appear that two opposite pieces are meeting in the same case of the array. The method is going to see which piece has the best rank one in order to define which piece is the winner of the « fight ».

STRATEGO

This is the facade class for the model.

Attributes

-PLAYER 1&2: Instantiate two players

-CURRENTPLAYER: TIP the current player

-**BOARD:** Instantiate a board in order to start the game.

METHODS

-**GETMOVEPOSSIBLE**: This method gives all the possible moves for a player.

- GETCURRENTPLAYER: Return the reference of the current player**.**

-CURRENTPLAYERHASMOVE: This method checks if a player still has unparalyzed pieces.

-GETWINNER: This method returns if there is a winner, the player who won**.**

-SWAPCURRENTPLAYER: Change the current Player**.**

-GETMOVEPOSSIBLEFORPIECE: This method gives return a vector which contains all the possible movement for a player**.**

-MOVE: move a piece from a source to destination**.**