// ###########################################################

// # init\_board - This function initializes the board to default values #

//

// # Arguments:

// # board - character, 2-D array

//

// # Return value: # None

// ###########################################################

void init\_board**(**char board**[**8**][**8**]);**

This function initializes the board with default values as all the board should be empty accept the middle of the board will have 4 coins, 2 black and 2 white coins which is initial state of the board.

// ###########################################################

// # print\_board - This function print the elements of board #

//

// # Arguments:

// # board - character, 2-D array

//

// # Return value: # None

// ###########################################################

void print\_board**(**char board**[**8**][**8**]);**

This function print the board in such a manner that it looks like a matrix and just as it is played. It just loops through the board array and prints elements one by one.

// ###########################################################

// # checkFlip- This function checks if flip of a color is possible #

//

// # Arguments:

// # board - character, 2-D array

// # color - character, player

// # line - integer, row

// # col - integer, column

// # deltaline - integer, row in specific direction

// # deltacol - integer, colum in specific direction

// # Return value: true or false

// ###########################################################

bool checkFlip**(**char board**[**8**][**8**],**char color**,** int line**,** int col**,** int deltaline**,** int deltaCol**);**

this is a Boolean function and returns true or false. It basically checks whether the next coin can be flipped or not when a move is performed. The deltaline and deltaCol is the row and column where to check whether the coin can be flipped or not.

// ###########################################################

// # changeCoins - This function changes the coin in a specific row and column #

//

// # Arguments:

// # board - character, 2-D array

// # color - character, player

// # line - integer, row

// # col - integer, column

// # deltaline - integer, row in specific direction

// # deltacol - integer, colum in specific direction

// # Return value: none

// ###########################################################

void changeCoins**(**char board**[**8**][**8**],**char color**,** int line**,** int col**,** int deltaline**,** int deltaCol**);**

This is a non return type function and this takes arguments as listed above. This changes the coins in such a manner that if there is a black coin in deltaline and deltacol, then it will change it to white.

// ###########################################################

// # isValidMove - This function checks if the move performed is valid or not #

//

// # Arguments:

// # board - character, 2-D array

// # color - character, player

// # line - integer, row

// # col - integer, column

// # Return value: true or false

// ###########################################################

bool isValidMove**(**char board**[**8**][**8**],**int line**,** int col**,**char color**);**

This function by name is made to check if a move performed is valid or not. Because there can be chance that user can input a row and column where already a coin exists.

// ###########################################################

// # getColumn - This function convert char column to column number #

//

// # Arguments:

// # col - integer, column

//

// # Return value: integer, index

// ###########################################################

int getColumn**(**char col**);**

This is a basic function that gives the integer value for the passed character of column. As character cannot be made to access an index so it is converted to integer.

// ###########################################################

// # play - This function make move of the player #

//

// # Arguments:

// # board - character, 2-D array

// # color - character, player

// # line - integer, row

// # col - integer, column

// # Return value: none

// ###########################################################

void play**(**char board**[**8**][**8**],** int line**,** char colm**,** char color**);**

This is the important function of the code. It is play function which takes in the moves of the player and then later the moves are checked, and coins are flipped accordingly.

// ###########################################################

// # count\_flips\_dir - This function counts the no of flips in a direction #

//

// # Arguments:

// # board - character, 2-D array

// # color - character, player

// # line - integer, row

// # col - integer, column

// # deltaline - integer, row in specific direction

// # deltacol - integer, colum in specific direction

// # Return value: interger, counter

// ###########################################################

int count\_flips\_dir**(**char board**[**8**][**8**],**char color**,** int line**,** int col**,** int deltaline**,** int deltaCol**);**

This function gives the amount of coins that can be flipped for a certain move and for a certain deltaline and deltacol.

// ################################################################

// # flanked - This function gives the total coins fliped by move #

//

// # Arguments:

// # board - character, 2-D array

// # color - character, player

// # line - integer, row

// # col - integer, column

// # Return value: integer, counter

// ###########################################################

int flanked**(**char board**[**8**][**8**],** int line**,** int colm**,**char color**);**

This function gives the total amount of coins that can be flipped for every deltaline and deltacol for a certain move.

// ###########################################################

// # isGameOver - This function checks if game is over or not #

//

// # Arguments:

// # board - character, 2-D array

// # Return value: true or false

// ###########################################################

bool isGameOver**(**char board**[**8**][**8**]);**

This is a Boolean function and returns true if the board is all filles with coins and return false if even 1 of the index is not filled.

// ###########################################################

// # getWinner - This function gives the winner of game #

//

// # Arguments:

// # board - character, 2-D array

// # Return value: 'o' or 'x'

// ###########################################################

char getWinner**(**char board**[**8**][**8**]);**

This function return character of coin which has the highest count in the board. Basically this takes in the board and counts the o and x. if o has greater count then it returns o.