import java.util.Scanner;

import static java.lang.System.\*;

public class Multiplayer

{

public static String player1 = "";

public static String player2 = "";

private static boolean playerTurn = true;

private static String name1 = "";

private static String name2 = "";

public static String player1Color = "";

public static String player2Color = "";

private static boolean win = false;

private static Scanner keyboard = new Scanner(System.in);

private static int colm1,rowm1;

public static int wKingc = 4;

public static int bKingc = 4;

public static int wKingr = 7;

public static int bKingr = 0;

public static String color = "";

public static boolean stalemate = false;

public static boolean checkmate = false;

public static void start()

{

int j = 0;

boolean playerOneWin = true;

boolean playerTwoWin = true;

out.print('\u000c');

out.print("Enter the name of player 1 --> ");

name1 = keyboard.nextLine();

out.print("Enter the name of player 2 --> ");

name2 = keyboard.nextLine();

while(j != 1) // Start of code by Ayaan Nazir

{

out.print("\n" + name1 + ": Black? (b) or White? (w) : ");

String color = keyboard.nextLine();

setColor(color); //Jon and Sherwin did setColor

out.print('\u000c');

switch(color)

{

case "w" : out.println("Great! You start first\n\n"); player1 = name1; player2 = name2; j++; break;

case "W" : out.println("Great! You start first\n\n"); player1 = name1; player2 = name2; j++; break;

case "b" : out.println("Great! You start second\n\n");player2 = name1; player1 = name2; j++; break;

case "B" : out.println("Great! You start second\n\n");player2 = name1; player1 = name2; j++; break;

default : out.println("Invalid color. Please enter either black (b) or white (w)\n"); break;

}

}

Board.create(player1, player2); // End of code by Ayaan Nazir

// playerOneWin && playerTwoWin

int cur = 0;

int l = 0;

while(!(stalemate || checkmate))

{

callMove();

Board.update(player1, player2, playerTurn, colm1, rowm1);

if(Board.board1[0][0].checkForStalemate(color))

stalemate = true;

else if(Board.board1[0][0].checkmate(color) && Board.board1[0][0].inCheck(color))

checkmate = true;

out.println("\n");

out.println(player1 + "'s score = " + Board.board1[rowm1][colm1].getPlayerOneScores() + " " + player2 + "'s score = " +Board.board1[rowm1][colm1].getPlayerTwoScores() +"\n\n");

//Board.board1[rowm1][colm1].printCaptured();

}

if(checkmate)

{

out.println("###########\n#Checkmate#\n###########");

if(color.equals("w"))

out.println("Congratulations " + player1 + "! You have successfully checkmated " + player2 + "!");

else

out.println("Congratulations " + player2 + "! You have successfully checkmated " + player1 + "!");

}

else

{

out.println("###########\n#Stalemate#\n###########");

if(color.equals("w"))

out.println("Congratulations " + player1 + "! You have successfully outsmarted " + player2 + " and forced a stalemate!");

else

out.println("Congratulations " + player2 + "! You have successfully outsmarted " + player1 + " and forced a stalemate!");

}

}

public static void setColor(String c)//Jon and Sherwin

{

if(c.equals("w") || c.equals("W"))

{

player1Color = "white";

player2Color = "black";

}

else if(c.equals("b") || c.equals("B"))

{

player1Color = "black";

player2Color = "white";

}

}

public String getPlayer1Color()

{

return player1Color;

}

public String getPlayer2Color()

{

return player2Color;

}

public static void callMove()//Start of Evan

{

String piece = "";

String col = "";

String row = "";

int colm = 0;

int rowm = 0;

colm1 = 0;

rowm1 = 0;

if(playerTurn)//Checks to see if it is player one’s turn

{

playerTurn = false;//switches to player 2

color = "w";

int f = 0;

while(f!=1)

{

out.print("\n\n" + player1 + " select your piece → "); //Start of Ayaan

String input = "";

input = keyboard.nextLine();

if(checkValidity(input,2) && checkInput(input,2))

{

colm = (interpret(input,2)[0]);//equal to 2

rowm = (interpret(input,2)[1]);//equal to 6

if(!(Board.board1[rowm][colm].getPieceType().equals("#") || Board.board1[rowm][colm].getPieceType().equals("/")))

{

out.print("\n\n" + player1 + " select where you would like the piece to go → "); //Start of Ayaan

int l = 0;

while(l!=1)

{

input = keyboard.nextLine();

if(checkValidity(input,2) && checkInput(input,2))

{

colm1 = (interpret(input,2)[0]);//equal to 2

rowm1 = (interpret(input,2)[1]);//equal to 4

l++;

}

else

out.println("Invalid syntax. Please input the new location as follows: \"letter\"\"number\"");

}

if(!Board.board1[rowm][colm].checkForCheck(rowm1, colm1, "w"))//checks if the user's input will put them in check

{

if(Board.board1[rowm][colm].movePiece(rowm1,colm1,"w")){f++;}else{out.println("Invalid move. Please retry or reference the rules using ");}

}

else

out.println("If you made this move, you'd be in check!");

int v = 0;

while(v!=1)

{

if(Board.board1[rowm1][colm1].getPieceType().equals("P") && rowm1 == 0)

{

out.print("Enter (in lowercase) the first letter of the piece you would like to promote your piece to (n for knight)-->");

String promotion = keyboard.nextLine();

if(promote(rowm1,colm1,promotion, "w"))

{

out.println("Your pawn has been successfully promoted!");

v++;

}

else

out.println("Incorrect promotion.");

}

else

v++;

}

}

else

out.println("Invalid location. Please input a location that contains a piece");

}

else

out.println("Invalid syntax. Please input the location: \"letter\"\"number\" without spaces or quotes");

}

}

else

{

playerTurn = true;//switches to player 1

color = "b";

int f = 0;

while(f!=1)

{

out.print("\n\n" + player2 + " select your piece → "); //Start of Ayaan

String input = "";

input = keyboard.nextLine();

if(checkValidity(input,2) && checkInput(input,2))

{

colm = (interpret(input,2)[0]);//equal to 2

rowm = (interpret(input,2)[1]);//equal to 6

if(!(Board.board1[rowm][colm].getPieceType().equals("#") || Board.board1[rowm][colm].getPieceType().equals("/")))

{

out.print("\n\n" + player2 + " select where you would like the piece to go → "); //Start of Ayaan

int l = 0;

while(l!=1)

{

input = keyboard.nextLine();

if(checkValidity(input,2) && checkInput(input,2))

{

colm1 = (interpret(input,2)[0]);//equal to 2

rowm1 = (interpret(input,2)[1]);//equal to 4

l++;

}

else

out.println("Invalid syntax. Please input the new location as follows: \"letter\"\"number\"");

}

if(!Board.board1[rowm][colm].checkForCheck(rowm1, colm1, "b"))//checks if the user's input will put them in check

{

if(Board.board1[rowm][colm].movePiece(rowm1,colm1,"b")){f++;}else{out.println("Invalid move. Please retry or reference the rules using ");}

}

else

out.println("If you made this move, you'd be in check!");

int v = 0;

while(v!=1)

{

if(Board.board1[rowm1][colm1].getPieceType().equals("p") && rowm1 == 7)

{

out.print("Enter (in lowercase) the first letter of the piece you would like to promote your piece to (n for knight)-->");

String promotion = keyboard.nextLine();

if(promote(rowm1,colm1,promotion, "b"))

{

out.println("Your pawn has been successfully promoted!");

v++;

}

else

out.println("Incorrect promotion.");

}

else

v++;

}

}

else

out.println("Invalid location. Please input a location that contains a piece");

}

else

out.println("Invalid syntax. Please input the location: \"letter\"\"number\" without spaces or quotes");

}

}

}

public static int[] interpret(String move, int number)

{

String letCord = move.substring(0,1);

String numCord = move.substring(1);

int[] translated = new int[number];

switch(letCord)

{

case "a": translated[0] = 0;break;

case "b": translated[0] = 1;break;

case "c": translated[0] = 2;break;

case "d": translated[0] = 3;break;

case "e": translated[0] = 4;break;

case "f": translated[0] = 5;break;

case "g": translated[0] = 6;break;

case "h": translated[0] = 7;break;

}

switch(numCord)

{

case "1": translated[1] = 7;break;

case "2": translated[1] = 6;break;

case "3": translated[1] = 5;break;

case "4": translated[1] = 4;break;

case "5": translated[1] = 3;break;

case "6": translated[1] = 2;break;

case "7": translated[1] = 1;break;

case "8": translated[1] = 0;break;

}

return translated;

}

public static boolean checkValidity(String check, int length)

{

if(check.length() == length)

return true;

else

return false;

}

public static boolean checkInput(String check, int length)

{

switch(check.charAt(length-2))

{

case 'a':break;

case 'b':break;

case 'c':break;

case 'd':break;

case 'e':break;

case 'f':break;

case 'g':break;

case 'h':break;

default: return false;

}

switch(check.charAt(length-1))

{

case '1':break;

case '2':break;

case '3':break;

case '4':break;

case '5':break;

case '6':break;

case '7':break;

case '8':break;

default: return false;

}

return true;

}

public static boolean promote(int r, int c, String n, String co)

{

if(co.equals("w"))

switch(n)

{

case "b": Board.board1[r][c] = new Bishop(r,c,"w","B");return true;

case "n": Board.board1[r][c] = new Knight(r,c,"w","N");return true;

case "q": Board.board1[r][c] = new Queen(r,c,"w","Q");return true;

case "r": Board.board1[r][c] = new Rook(r,c,"w","R");return true;

default: return false;

}

else

switch(n)

{

case "b": Board.board1[r][c] = new Bishop(r,c,"b","b");return true;

case "n": Board.board1[r][c] = new Knight(r,c,"b","n");return true;

case "q": Board.board1[r][c] = new Queen(r,c,"w","Q");return true;

case "r": Board.board1[r][c] = new Rook(r,c,"b","r");return true;

default: return false;

}

//Board.board1[r][c] (i,j,"b","r")

}

}