Team Solid FOURtress: Adeebur Rahman, Shivasuryan Vummidi, Edwin Mok
APCS1 pd 3
Project UML Diagram Final Phase
2017-01-21

Interface ConnectFour

Methods:

void create_board(int rows, int columns)
/ Creates a 2D array with the number of rows and columns inputted */*

void check_winner (Player p1, Player p2)
*/*Checks which player has won the game*/*

Class Woo

*/*DRIVER CLASS*/*

Methods:

void create_board(int rows, int columns)
void check_winner (Player p1, Player p2)
void is_column_full(int column)
*/*Returns true if a column is not full; returns false otherwise*/*
void set(int row, int column, char x)
*/*Gets a specific row and column position on the board and sets that space to the player's character token.*/*
void drop(int column, Player p)
*/*Figures out which is the lowest row in the column specified and then calls set() on the last available empty space*/*
*/*Updates _lastRow and _lastColumn for the player p*/*
String printBoard()
*/*Prints the board as a string*/*
void newGame()
*/*Begins the session of the game.*/*
*/*It first asks whether player 1 or player 2 are humans, or computers with varying difficulty*/*
*/*Assigns a new instance of a computer or a User to either player 1 or player 2*/*
*/*Player options:*

1. *A real person (Class User)*
2. *An EasyComputer (random AI)*
3. *A MediumComputer (vertical and horizontal offense; vertical defense)*
4. *A HardComputer (vertical and horizontal defense and offense*
5. *TBM (vertical, horizontal, and diagonal defense and offense*/*

Abstract superclass Player

Instance Variables:

protected int tokens; *//number of tokens each Player has*
protected String name; *//name of the Player*
protected char token_name; *//the token the Player uses*
protected int _lastRow;
*/*the last row the Player dropped the token in*/*
protected int _lastColumn;
*/*the last column the Player dropped the token in*/*
String win; /*Contains 4 tokens next to each other*/

Methods:

public void drop_token (int column, Woo w)
*/*Drops a token in the specified column*/*
public hasSubString (String str, String lookingFor)
*/*returns true if lookingFor is inside str*/*
public String getLastRowString (Woo w) and public String getRowString (Woo w, int row)
*/*Returns whatever was inside the last row the Player dropped the token in.*/*
public String getLastColumnString (Woo w) and public String getColumnString (Woo w, int column)
*/*Returns whatever was inside the last column the Player dropped the token in.*/*
public String getLastLeftDiagonalString (Woo w) and public String getLeftDiagonalString (Woo w, int row, int column)
*/*Returns whatever was inside the last left diagonal the Player dropped the token in.*/*
public String getLastRightDiagonalString (Woo w) and public String getRightDiagonalString (Woo w, int row, int column)
*/*Returns whatever was inside the last right diagonal the Player dropped the token in.*/*
public boolean is_win (Woo w)
*/*Checks whether there is 4 in a row for horizontal, vertical, or diagonal columns by calling hasSubstring on getLastRowString, getLastColumnString, getLastLeftDiagonalString, getLastRightDiagonalString and testing to see if either strings have win inside them. It returns true if at least one string tested has win inside it.*/*
public String toString() *//returns the name of the Player*
abstract int pick_column() ***// to be implemented in subclasses***

Subclass User extends Player

public User(): /*Name = 'User'; token_name = 'U'*/
public User(String userName, char token_Name)
/*name = userName; token_name = token_Name*/

Methods:

public int pick_column(Woo w, Player p)
*/*Asks the Player p to pick a column number. It then returns an integer for drop() in Woo to drop the token in */*

Subclass EasyComputer extends Player

public EasyComputer(): /*Name = 'EasyComputer'; token_name = 'E'*/
public EasyComputer(String userName, char token_Name)
/*name = userName; token_name = token_Name*/

Methods:

public int pick_column(Woo w, Player p)
*/*Asks the Player p to pick a column number. It then returns an integer for drop() in Woo to drop the token in */*
*/*In EasyComputer, the AI randomly selects a column to drop a token in.*/*

Subclass MediumComputer extends EasyComputer

public MediumComputer(): /*Name = 'MediumComputer'; token_name = 'M'*/
public MediumComputer(String userName, char token_Name)
/*name = userName; token_name = token_Name*/

Methods:

public int[] randomizeArray(int[] array) *// Randomly returns an array and randomizes its element positon*
public int defense (Woo w, Player p) *//Decides which column to defensively place a token. Vertical defense only.*
public int offense (Woo w, Player p)
//Decides which column to offensively place a token. First vertical offense, then horizontal.
public int pick_column(Woo w, Player p)
*/*Asks the Player p to pick a column number. It then returns an integer for drop() in Woo to drop the token in */*
*/*In MediumComputer, the AI calls on defense (Woo w, Player p) and offense (Woo w, Player p).*/*
//Defense first, then offense. If it has no other option, then act randomly

Subclass HardComputer extends MediumComputer

public HardComputer(): /*Name = 'HardComputer'; token_name = 'H'*/
public HardComputer(String userName, char token_Name)
/*name = userName; token_name = token_Name*/

Methods:

public int defense (Woo w, Player p) *//Decides which column to defensively place a token horizontally and/or to win.*
public int secondaryDefense(Woo w, Player p) *//Inherits the vertical Defense from MediumComputer*
public int pick_column(Woo w, Player p)
*/*Asks the Player p to pick a column number. It then returns an integer for drop() in Woo to drop the token in */*
*/*In HardComputer, the AI calls on defense (Woo w, Player p), secondaryDefense (Woo w, , Player p) and offense (Woo w, Player p).*/*
//Horizontal defense first, then vertical defense, then offense. If it has no other option, then act randomly

Subclass TBM extends HardComputer

public TBM(): /*Name = 'TBM'; token_name = 'T'*/
public TBM(String userName, char token_Name)
/*name = userName; token_name = token_Name*/

Methods:

public int defense (Woo w, Player p) *//Decides which column to defensively place a token diagonally.*
public int offense (Woo w, Player p) *//Decides which column to offensively place a token diagonally.*
public int secondaryDefense(Woo w, Player p) *//Inherits the defense from MediumComputer and uses that*
public int pick_column(Woo w, Player p)
*/*Asks the Player p to pick a column number. It then returns an integer for drop() in Woo to drop the token in */*
*/*In TBM, the AI calls on defense (Woo w, Player p), secondaryDefense (Woo w, , Player p), super.secondaryDefense(Woo w, Player p) and offense (Woo w, Player p).*/*
//Diagonal defense, then horizontal defense first, then vertical defense, then diagonal offense, horizontal offense, and then vertical offense.. If it has no other option, then act randomly