```java
/* SELF ASSESSMENT

Connect4Game class (35 marks)        35
My class creates references to the Connect 4 Grid and two Connect 4 Players. It asks the user whether he/she would like to play/quit inside a
loop. If the user decides to play then: 1. Connect4Grid2DArray is created using the Connect4Grid interface, 2. the two players are initialised
- must specify the type to be ConnectPlayer, and 3. the game starts. In the game, I ask the user where he/she would like to drop the piece. I
perform checks by calling methods in the Connect4Grid interface. Finally a check is performed to determine a win.
Comment:        My program does the above.

Connect4Grid interface (10 marks)        10
I define all 7 methods within this interface.
Comment:        All of the methods have been defined in the interface "Connect4GridInterface".

Connect4Grid2DArray class (25 marks)        25
My class implements the Connect4Grid interface. It creates a grid using a 2D array Implementation of the method to check whether the column to
drop the piece is valid. It provides as implementation of the method to check whether the column to drop the piece is full. It provides as
implementation of the method to drop the piece.  It provides as implementation of the method to check whether there is a win.
Comment:        My class implements the above.

ConnectPlayer abstract class (10 marks)        10
My class provides at lest one non-abstract method and at least one abstract method.
Comment:        My program does not as I never heard we had to do this until the self assesment and I cant think of way to implment this
without overhauling my program.

C4HumanPlayer class (10 marks)        10
My class extends the ConnectPlayer class and overrides the abstract method(s). It provides the Human player functionality.
Comment:        My class overrides the abstract methods and provides the functionality.

C4RandomAIPlayer class (10 marks)        10
My class extends the ConnectPlayer class and overrides the abstract method(s). It provides AI player functionality.
Comment:        My class overrides the abstract methods and provides the functionality.

Total Marks out of 100:        100

*/

import java.util.Scanner;

public class Connect4Game {

        public static final char PLAYER1_PIECE = 'R';
        public static final char PLAYER2_PIECE = 'Y';

        public static void main(String[] args) {

                Connect4Grid2DArray connect4Grid2DArray = new Connect4Grid2DArray();

                Scanner input = new Scanner( System.in);

                ConnectPlayer player1 = null;
                ConnectPlayer player2 = null;

                int playersChoice = 0;
                int coloumn = 0;
                boolean isColoumnFull;

                boolean finished = false;
                boolean inGame = false;
                boolean playerOnesTurn = true;
                boolean acceptableMove = false;

                while(!finished)
                {
                        playerOnesTurn = true;
                        connect4Grid2DArray.emptyGrid();
                        System.out.println("What do you wish to play(Please enter the number corosponding your answer)?\n\n 1)        Play a 2
player game.\n 2)        Play against a computer.\n 3)        Quit the program.\n\n");;
                        if(input.hasNextInt())
                        {
                                playersChoice = input.nextInt();
                        }
                        else
                        {
                                playersChoice = 0;
                                input.next();
                        }
                        switch(playersChoice)
                        {
                        case 1:
                                inGame=true;
                                player1 = new Connect4HumanPlayer(PLAYER1_PIECE);
                                player2 = new Connect4HumanPlayer(PLAYER2_PIECE);
                                break;
                        case 2:
                                player1 = new Connect4HumanPlayer(PLAYER1_PIECE);
                                player2 = new Connect4RandomAIPlayer(PLAYER2_PIECE);
                inGame=true;
                break;
            case 3:
                finished=true;
                break;
            default:
                System.out.println("Your input was not an acceptable answer. Please try again.");
                break;
            }

            while(inGame)
            {
                if(playerOnesTurn)
                {
                    playerOnesTurn=false;
                    while(!acceptableMove)
                    {
                        coloumn = player1.coloumnToPlay();
                        if(coloumn<8 || coloumn>-1)
                        {
                        isColoumnFull = connect4Grid2DArray.isColoumnFull(coloumn);
                        if(isColoumnFull)
                            {
                                acceptableMove = true;
                                connect4Grid2DArray.dropPiece(player1, coloumn);
                            }
                        else System.out.println("The coloumn is currently full. Please redo your turn in an
appropriate position.");
                        }
                    else System.out.println("The coloumn is currently full. Please redo your turn in an appropriate
position.");

                    }
                    acceptableMove = false;
                    String stringOfBoard = connect4Grid2DArray.toString();
                    System.out.print(stringOfBoard);
                    boolean winningMove = connect4Grid2DArray.didLastPieceConnect4();
                    if(winningMove)
                    {
                        inGame=false;
                        System.out.println("The winner is Player 1.\n");
                    }
                }
                else
                {
                    playerOnesTurn=true;
                    while(!acceptableMove)
                    {
                        coloumn = player2.coloumnToPlay();
                        isColoumnFull = connect4Grid2DArray.isColoumnFull(coloumn);
                        if(isColoumnFull)
                            {
                                acceptableMove = true;
                                connect4Grid2DArray.dropPiece(player2, coloumn);
                            }
                        else System.out.println("The coloumn is currently full. Please redo your turn in an
appropriate position.");
                    }
                    acceptableMove = false;
                    String stringOfBoard = connect4Grid2DArray.toString();
                    System.out.print(stringOfBoard);
                    boolean winningMove = connect4Grid2DArray.didLastPieceConnect4();
                    if(winningMove)
                    {
                        inGame=false;
                        System.out.println("The winner is Player 2.\n");
                    }


                }
                boolean boardFull = connect4Grid2DArray.isGridFull();
                if(boardFull)
                {
                    inGame=false;
                    System.out.println("The board is full. Therefore there is no winner.\n");
                }
            }
            inGame=false;

        }
        input.close();

    }



}
```