25S\_CST8116\_472 Intro to Computer Programming

Project #3 – Putting it all together.

Student Name: Eduardo Henrique Pinheiro Silva

Student ID: 041217885

Instructor: Istiaque Shahriar

Date: September 28, 2025

# Abstract

This report presents the development and implementation of a console-based Connect Four game written in Java. The project integrates multiple programming concepts such as control structures, arrays, methods, and user input validation. The following sections summarize the design, pseudocode, source code, execution outputs, testing plan, and conclusions.

# 1. Introduction

The Connect Four project is designed to consolidate programming skills acquired during the course. The game allows two players, represented by 'R' (Red) and 'Y' (Yellow), to alternate turns dropping discs into a 6x7 grid. The objective is to align four identical symbols horizontally, vertically, or diagonally before the opponent does.

# 2. Program Design and Pseudocode

The program is organized into three Java classes: Board.java, ConnectFourGame.java, and ConnectFour.java. Each class has a distinct responsibility: Board handles grid operations and win logic, ConnectFourGame manages the game loop and input validation, and ConnectFour contains the main method. The pseudocode below provides an overview of the core logic.

Pseudocode — Connect Four

SET ROWS = 6, COLS = 7

CREATE grid[ROWS][COLS] filled with ' '

FUNCTION dropDisc(col, disc):

IF col out of range OR top cell not empty: RETURN -1

FOR r from ROWS-1 down to 0:

IF grid[r][col] == ' ':

grid[r][col] = disc

RETURN r

RETURN -1

FUNCTION hasConnectFourFrom(row, col, disc):

FOR each direction in [(0,1),(1,0),(1,1),(1,-1)]:

count = 1

count += countDirection(row, col, +dr, +dc, disc)

count += countDirection(row, col, -dr, -dc, disc)

IF count >= 4: RETURN TRUE

RETURN FALSE

MAIN LOOP:

current = 'R'

WHILE TRUE:

printBoard()

col = promptUser(1..7)

row = dropDisc(col-1, current)

IF row == -1: print "invalid"; CONTINUE

IF hasConnectFourFrom(row, col-1, current): printBoard(); print current + " wins"; BREAK

IF board.isFull(): printBoard(); print "Tie"; BREAK

current = (current == 'R') ? 'Y' : 'R'

# 3. Source Code Screenshots

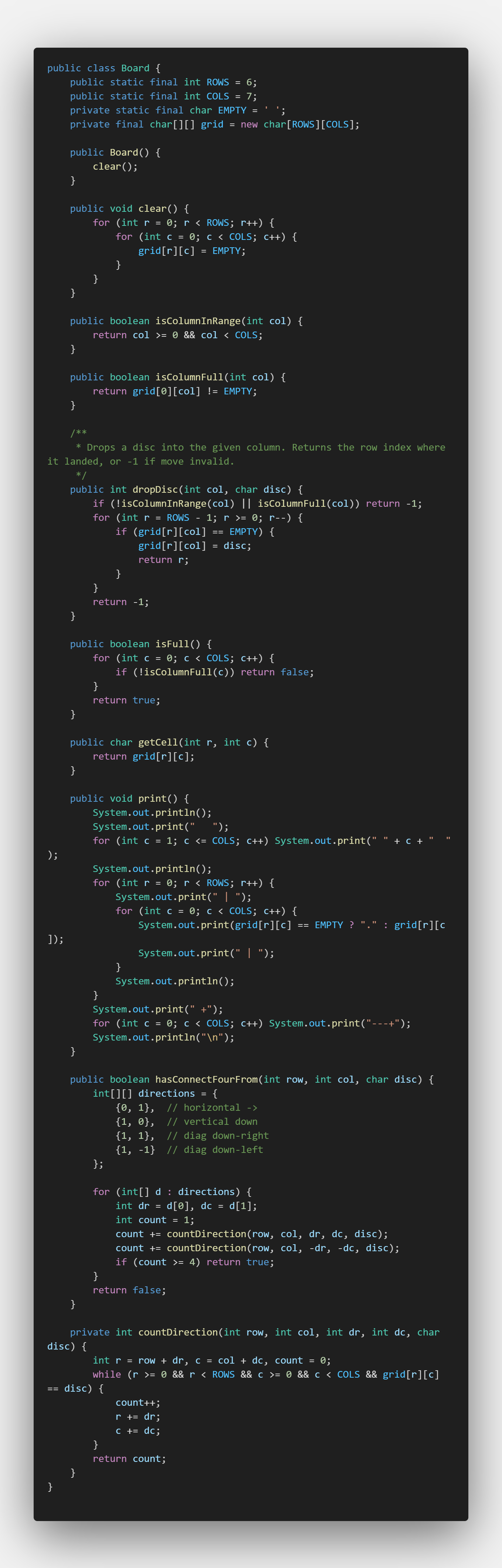
Insert screenshots of your modified source code files below:

• Board.java – include methods dropDisc() and hasConnectFourFrom()

• ConnectFourGame.java – include the start() loop and input validation logic

• ConnectFour.java – show the main() method

# 4.



# Program Execution (Console Output)

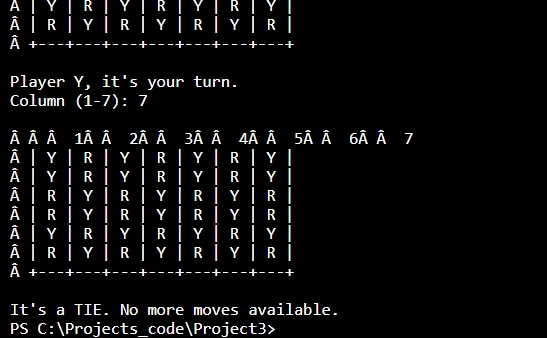
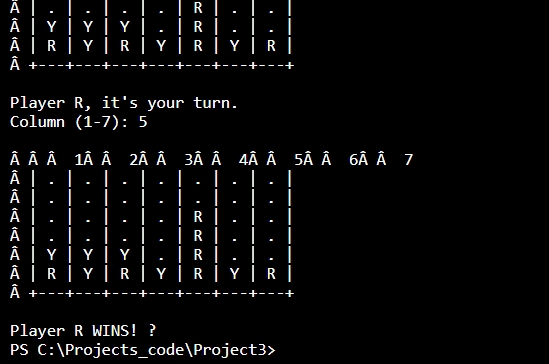
Insert screenshots showing the program running in the console:

• Player R wins (horizontal example)

• Player Y wins (diagonal example)

• Tie (board full)

• Input validation (invalid column or full column)



# 5. Test Plan

The following table summarizes the main test scenarios used to validate the Connect Four program. Each case verifies a distinct rule or behavior such as input validation, win detection, and tie conditions.

# 6. Conclusion

This project successfully demonstrates the application of programming concepts learned throughout the course. It integrates conditionals, loops, arrays, and method abstraction in a structured and readable program. The process reinforced problem-solving, debugging, and version control practices using Git and GitHub.

# 7. GitHub Repository Link

Repository URL: https://github.com/edhdevps/connect4\_project