Connect Four Comprehensive Exercise Project

Om Pandey, Anthony Du, Bowen Deng, Shuhao Liu

Requirements/Analysis

- List of Requirements:
  - Detect horizontal, vertical, and diagonal connections.
  - 2 different chips for 2 different players.
  - Chips only drop down to the lowest empty spot in the column.
  - Detect whether a player has won (horizontally, vertically, and diagonally).
  - Get user input from the player to see which column they want to drop.
  - Keep track of how many chips either player has dropped.

Design Options

- Design Option 1:
  - Create 3 separate classes to each handle their own functionalities.
  - The user interface, connect four, and player classes have their own fields and methods.
- Design Option 2:
  - Create just 1 connect four class that will handle all functions of the entire program.
  - This includes doing the software inputting and outputting, as well as controlling the rules of the players and the game.

Pros and Cons of each Design Option

Design Option 1:

-Pros:

- The program is easier to keep track of and understand.
- Avoids some repetition.

-Cons:

- Makes the program fragmented and harder to read with all of the class calls if done incorrectly.

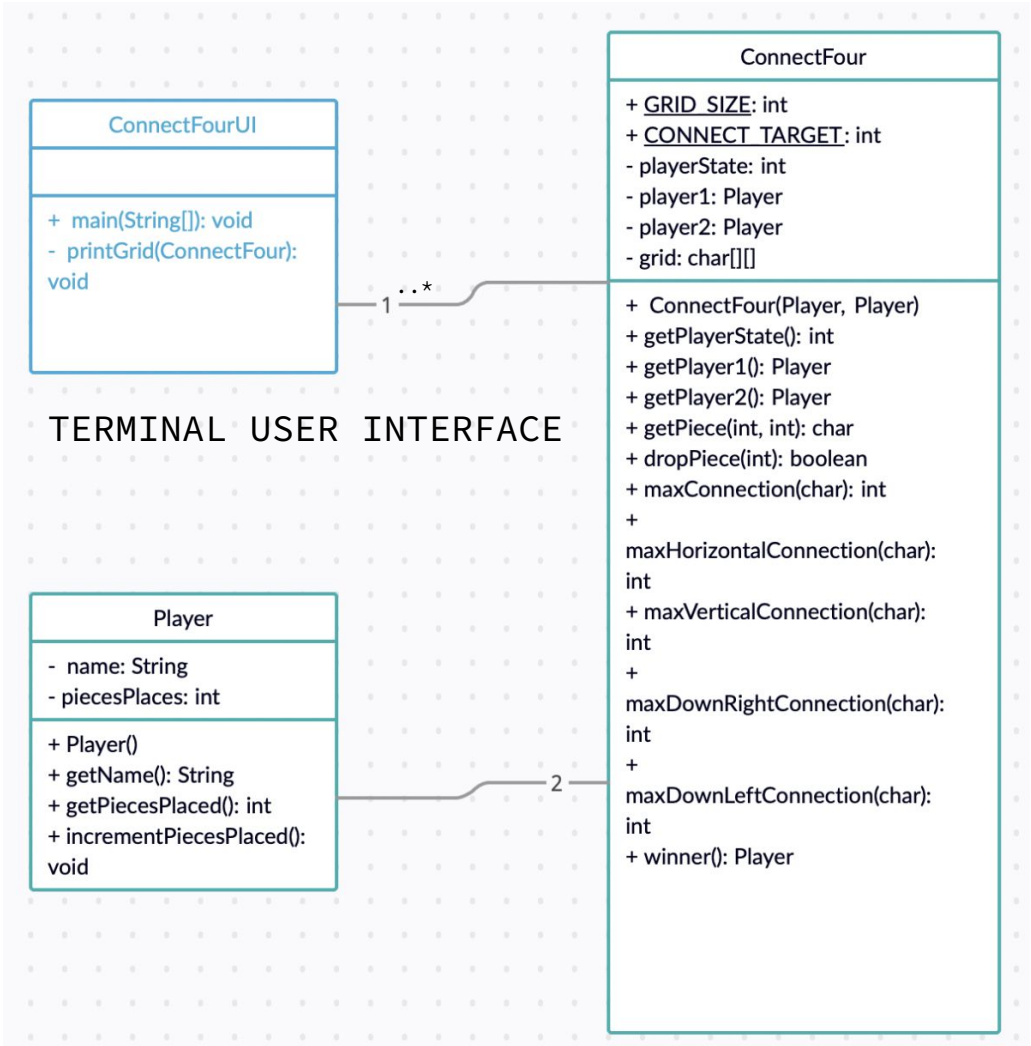
Design Option 2:

-Pros:

- Makes the program more cohesive since all functionalities are in one class.

-Cons:

- The program is harder to read and comprehend.



TERMINAL USER INTERFACE

We Chose Design Option 1

Demo of Implementation

- This is a new implementation updated with the updated client.
- It’s design would be marginally different from our initial design with added private attributes and methods in the Player and ConnectFour classes including a gamesWon variable in Player and a non-final gridSize and connectTarget in ConnectFour and their respective getters and a reset method in ConnectFour for looping the game.

- javac -d bin -cp bin src/*
- java -cp bin ConnectFourUI

Unit and Integration Testing

- ConnectFourTest.java
  - testMaxConnection
  - testDropPiece
  - testGetPiece
  - testGetPlayerState
  - testGetPlayer1
  - testGetPlayer2
- PlayerTest.java
  - testGetName
  - testGetPiecesPlaced
  - testIncrementPiecesPlaced
- System Test on ConnectFourUI
  - testUserInterface
  - testDropColumn1
  - testDropColumn8
  - testInvalidInputNonInteger
  - testInvalidInputOutOfRange
  - testHorizontalWin
  - testVerticalWin
  - testDiagonalWin

Lessons Learned

- It is a good idea to have multiple design options available so there is a chance you have come up with an easier way of designing and implementing a program.

- It is a good idea to form a overall structure in your head about how to write the code before writing it.

- When dealing with 2D Arrays, visualization helps.