




## INSTRUCTIONS

Complete the project detailed in this document using valid, executable Java code. Submit your work (including only the `.java` source code files indicated with  in the specification below) to your lab instructor via eCampus no later than the deadline on the course calendar. At the top of every file you submit, include a comment with your full name and the tracking code `1908-110-3` for instructor use. Your work will be compared to the work of your peers and online sources to detect plagiarism.

## LEARNING GOALS

The goal of this project is to reinforce key concepts of object oriented programming, data structures, and algorithms in a compelling scenario. It is also intended to reinforce the ability to work with an existing code base.

## SPECIFICATION

The game known in this project as Connect Three is a variant of the commercial game [Connect Four](#) in which a player wins by matching 3 of their pieces in a line (rather than 4) on board that is 5 rows by 4 columns (rather than 6 rows by 7 columns). The players are X and O, and X always goes first, resulting in first player advantage for X due to the peculiarities of this variant of the game.

You are given two files that form a partially working implementation of the Connect Three game using the JavaFX library.

- > The abstract class `ConnectThree` provides the GUI (graphical user interface), most of the logic except for win detection, and several artificial intelligences (AIs) for playing the game. The abstract class can't be executed, and its code may not be changed.
- > The class `Sample` includes inefficient and incomplete win detection, a mediocre AI, and the `main` method to launch the game.

Rename `Sample` to `LastName` (use your own last name) and modify the source code as follows.

- > Change the title of the window from *Connect Three* to *Connect Three by Your Name*, where you fill in your own name.
- > Implement **correct and efficient win detection** including all possible vertical, horizontal, and diagonal wins of 3 pieces. The sample win detection only detects vertical wins and does so inefficiently. Without implementing this first, it's difficult to create an AI.
- > Implement **at least one original AI to play the game** based on recommendations from the instructors.
  - > There are three built-in AIs (*Human*, *Random*, and *Naive*) included in the abstract class and one (*Lefty*) included in the sample class. These AIs and any custom AIs you implement in your own class can be selected independently for X and O using the drop-down menus at the bottom of the GUI. An AI plays automatically without delay if it decides a column to play in for each of its turns.
  - > Due to first player advantage, an AI generally wins more often as X as O (for example, *Naive* as X wins roughly 60% of games against *Naive* as O). The GUI includes testing features to automate many games and measure the win rates of both players.

Support for JavaFX depends on the JDK your project uses. In JDK 8 to 10, the JavaFX library is already included. In JDK 11 and newer, the JavaFX library is excluded and must be installed manually from the [JavaFX SDK](#) release and configured for use in your project.

## BONUS OPPORTUNITY

For **25 bonus points**, generalize your win detection and AI to work for Connect Three, Connect Four, and all other Connect N games, where N is the number of pieces to match in a line and is at least 3. In other words, do not hardcode your algorithms to work only for Connect Three, but instead generalize them to use the match count constant. Add the phrase *with Bonus* at the end of the window title.

## GRADING RUBRIC

Your grade will be based on your win detection and AI according to the following table.

<u>Your win detection accuracy</u>	<u>Your AI's approximate win rate as Player O against Naive as Player X</u>		
	At least 75%	At least 55% but less than 75%	Less than 55%
<b>Always correct</b>	100 points (A)	85 points (B)	75 points (C)
<b>Almost always correct with rare exceptions</b>	85 points (B)	75 points (C)	60 points (D)
<b>Rarely or never correct</b>	75 points (C)	60 points (D)	0 points (F)