INF1B Assignment 1

s2085530

Jack MCPAKE

2021

# Overview

My connect four program works by modelling a connect 4 board, manipulating the board through a controller class, and displaying it via a view class.

The users will play against each other by placing pieces on a board. The game will end when one of the players concedes or if the board is full.

I will be using the object-oriented language java to implement my solution.

# Basic Features

## Representing the Board

I will be modelling the state of the game through a board, which I have chosen to be as a 2d array of characters. With ‘a’ representing the character of one player, and ‘b’ representing the character of another player.

## Playing Pieces

To play a piece, the player must choose a column to ‘drop’ their piece into. If they chose a column with free space in it. The player’s piece will be dropped into the column one slot above the last played piece in that column. If the column is empty, the piece is dropped into the bottom of that column (or the highest index of that column). If the column is full, the player will be asked to put their piece into a different column.

To check if the player can play a piece, I created a module gameTermConditions() that would return one of two characters. ‘g’ for ‘go ahead’, meaning that the game should go ahead as normal, and ‘f’ for ‘full’ for when the board is full. I chose char for this so that I could display different messages based on the state of the board once the game is over.

## Game Loop

The game will continually ask the player to insert a piece, if they conceded, check if that piece is valid, and then play that piece until the board is full, or the player concedes. I chose a while loop to implement this, because I would have to loop the instructions based on if gameTermConditions() would return ‘g’.

## Game Over

Once the game is over (i.e. gameTermConditions() returns ‘f’, or move is put as 0 which means concede), the user will be prompted that either the current player has conceded or that the board is full. Based on which one of these are true, the message will be different.

## Testing

I tested this part of the program by playing some games against myself. Early on I came across and error where the program would crash if the user put a value that wasn’t one of the columns from the game board, I fixed this by adding a check when a user plays a move called isMoveValid() which checks if the move is within the permissible range of column values.

# Intermediate Features

## Start New Game

## Variable Game Settings

## Enhanced Input Validation

## Automatic Win Detection

## Changes to Basic Features

# Advanced Features

## Save State

# Evaluation