# Introduction

This Coursework purpose is to implement a Checkers game using appropriate algorithms and data structures. To implement this game we will be using Java language with an integrated Java UI called Java Swing. As there are different versions of checkers game from a country to another, I chose to implement the international Draughts rules. To find official rules, we refer to the official world draught Federation (Fédération Mondiale du Jeu de Dames, http://fmjd.org ). This website does not provide rules but it lists all national draughts federations. None regions of the UK have a website, so I found out that French draught federation has one with well explained rules with schemes etc. You can find it out at : [www.ffjd.fr/Web/index.php?page=reglesdujeu](http://www.ffjd.fr/Web/index.php?page=reglesdujeu). You will find in the appendices a recap of the rules traduce in English language. This project contains following features : a gameboard representation, player representations, an artificial intelligence which is able to play, the game logic and rules, a graphical or textual interface, a game history with back/redo functionality, a game save option. This report will explain you the choices made to implement these functionalities, the possible enhancement that could have been done if more time for this coursework was available, a critical evaluation and finally a personal evaluation.

# Design

## Class Diagram representation

First, we made an object model representation of the game (see … in the appendices). <<Explain why I design this feature. >>

Then we had to implement each class with appropriate data strucures. We will explain for each element how it is implemented in this programm.

## Main used data structures

### Gameboard class :

The first functionality to implement was the gameboard. With international draught rules, The gameboard contains 10 lines and 10 columns (a total of 100 Checks). The most efficient and practical way to do this is to represent it with a double integer array of size 10: gameboard = new int[10][10]. With this structure, we can access a Check from its line number for the first array and column number for the second one (eg : Check line5Column7 = gameboard[5][7]. This will also be useful to move piece because we will just need to take Piece’s position and add or substract 1 from its current line and column to move it to its new position.

### Checks class

The gameboard is composed of 100 Checks. A Check is an object which has attributes such as a line number, a column number, a reference to a piece object or null if there is nothing on it. Check is a composition association from the gameboard, so it has a reference to it.

### Pieces abstract class (and Men/Kings classes implementation)

A piece is an abstract class. It can whether be a Man or a has a position (it is a reference to a Check or null if the piece has been taken by the adversary). It also have a color (black/white). For men, they have a destination which is an integer (0 for black pieces : the piece must go from gameboard’s top to bottom, 1 for white pieces : they must go from gameboard’s bottom to top). This value is unnecessary for King because they can move in the direction they want. Kings and men have also specific capture and movement methods because these two entities do not respond to same rules. ( Further explanation will be provided in <<ENTER SECTION NAME>> section .

### Player abstract class (and Humans/AI classes implementation)

Player is an abstract class because it can whether be a human or an AI. Each of them have an ArrayList linking to the pieces they own (it is the easiest way to stock and get a specific piece)+ they have a name.

### Game class

The game class is the main class which regroup every necessary objects for a to play a game. It has 2 players, a gameboard and two LinkedList. The first LinkedList is to keep copies of the gameboard (for undo a move or more). The second one is to stock redo moves.

### Graphical User Interface classes

The package UI contains every classes required for the Graphical User Interface. We will not explain in details the content of these classes as this functionality was not mandatory.

### Other classes

The code also contains other classes such as Tree.java and DeepCopy.java

The Tree class is a classical Node structure used to resprend n-ary trees. I needed it to calculate the longest capture possible for a given piece. The code is issued from (<<LINK>>)

The DeepCopy class is used to copy element and all the referenced elements of class as new object. It was also useful to calculate this longest capture to avoid infinite looping on same capture and display every possibility for every move.

## Main used algorithms