Programming Project - Assignment 1
Intelligent Systems Techniques (MSc) 2018/19

[50% of total module marks]

Overview

This is an individual assignment worth 50% of your Intelligent Systems Techniques module marks. The task is to construct and document a checkers game (draughts) in an object-oriented language (preferably: Java). The program should present an interactive draughts board that allows pieces to be moved around by the human player and the AI as per the rules of the game, which can be found here: http://www.indepthinfo.com/checkers/play.shtml
Your program should produce legal gameplay at different levels of difficulty, and employ an AI that is based on your implementation of Minimax and Alpha-Beta pruning.

You may start work on this assignment as soon as you like but bear in mind that the lectures on Game-playing, Minimax, and Alpha-Beta pruning will be given around the middle of term. Normally, it will make sense to commence work after the lecture which covers Minimax implementation in Java. This will still leave you plenty of time to complete the assignment.

Programming

This assignment involves programming so if you have difficulties in this area, it will be a good idea to take action earlier rather than later. Remember that I am more than happy to provide you access to additional Java material and exercises that will help you brush up on the basics quickly. All you need to do is write me an e-mail to sign up and then work hard to cover the relevant sections as outlined on Study Direct. I don't expect you to deliver perfect code but you should aim to follow the basic principles of good programming. These include:

Modularity: modular programs are made up from small, independent components (typically methods) that can be tested and debugged independently from the rest of the program. Such components get their data via formal input parameters. They only affect the rest of the program via the formal outputs (results) they return. They are like "black boxes" which, once validated, can be ignored. Modular programs make little or no use of class variables for communication information from one part of the program to another. Writing your program in a modular way is a divide-and-conquer strategy.

Simplicity: Do not introduce complications unless there is a real need. Do not start experimenting with advanced features of Java (e.g. final variables and inner classes) without good reason. If you find you are writing the same (or almost the same) code more than once, look for a way to get rid of the redundant parts. You'll probably need to create a new method or class which can be used in different situations.

Transparency: Choose the names for variables and methods so as to make the code self-explanatory as far as possible, e.g.: the name of a variable or method needs to describe its role or behavior. Use in-line comments in the code to document individual statements (if they do something important, or complex) but keep them short, and format them so as to minimise disruption to the visual structure of the code. Don't let line-wrapping obscure indentation.

Presentation: The program should be presented neatly and clearly, using consistent conventions for use of blank-lines etc. Indentation should be consistently used to show the logical structure of the program, particularly the presence of constructs like `if' statements and loops. This shouldn't be difficult as most programming environments take care of indentation almost automatically. You are not primarily marked on the style or quality of your code but should still strive to adhere to these common practices to avoid too many unnecessary deductions.

Marking Criteria

There are two groups of assessment criteria, Game Internals and Human-Computer-Interface (e.g., a GUI). Each of the criteria in these groups is worth an equal amount of marks (Total: 100).
When developing your program, you should try to satisfy as many of these as you can.
Satisfying a criterion in full will give you up to 4 marks. There may be deductions of varying degree for wrong or sub-standard solutions, any mistakes made, or when insufficient explanation was provided in the report.

Game Internals (AI Core, max. 4 marks each)

1. Interactive checkers gameplay (user v. AI) of some sort
2. Valid state representation
3. Successor function of some sort
4. Valid successor function
5. Use of successor function for validating user moves
6. Use of successor function for generating AI moves
7. Rejection of invalid user moves, with explanation given
8. Elements of Minimax evaluation present
9. Valid Minimax evaluation
10. Minimax evaluation uses pruning of some sort
11. Minimax evaluation uses valid Alpha-Beta pruning
12. Variable level of verifiable AI cleverness, adjustable by the user (e.g. a difficulty slider)
13. Valid AI moves
14. Multi-step user moves
15. Multi-step AI moves
16. Forcing of takes
17. Automatic king conversion

Human-Computer-Interface (HCI, max. 4 marks each)

1. Some kind of board representation displayed on screen
2. GUI updates the display properly after any completed moves (User and AI moves)
3. Full graphical board display
4. Interactive GUI of some sort
5. Mouse interaction focus, e.g., click to select & click to place, or drag & drop (better)
6. GUI pauses to show intermediate states in multi-step moves
7. Display of the basic game rules (e.g., a corresponding button opening a help window)
8. A help facility that provides hints about available moves, given the current game state. For instance, when enabled, any squares containing movable pieces might get a different colour.

Programming Project - Assignment 1
Intelligent Systems Techniques (MSc) 2018/19

[50% of total module marks]

Report

Your report is a very important part of the assessment and required to confirm that you have written the program by yourself and understood the AI functionality implemented in it. You must therefore clearly point out relevant marking criteria that you have addressed, and provide detailed descriptions of how you implemented each part, with references to your source code given. If you document your code extensively during development, then this will be a lot easier to do and you need less explanation in the actual report text. For this reason, there is no minimum word count given, but as a guideline, try not to exceed 2000-2500 words in total (excluding Appendix).
Your report should have a title page with your candidate number and contain the following sections:

1. **Introduction** – No more than a page (include a representative screen shot of your program here), introducing the task and your general approach to it. For instance, you could write here a few sentences on why you designed your main classes and methods the way they turned out, and very briefly, list the identity/functionality covered by each.
2. **Description of Program Functionality** – Here, you need to explain how your implementation addresses the marking criteria. You do not need to describe anything that is obvious and can normally ignore all GUI criteria as these tend to be self-evident. Also note that some criteria are qualifications (e.g., improvements) of other criteria where descriptions can be grouped. As a guideline, and depending on how much you've achieved, you should focus your description on the main criteria related to the Game Internals. For everything else, you need to make an intelligent choice as to what may need explaining.
3. **Appendix** (Source code as text, best with line numbers for easier reference from section 2).

Submission Requirements

Your submission must contain the following parts:

- **Your program files (source code, jar, class files)** – must be zipped up into a single ZIP file!
- **Your report as PDF** (included in the ZIP or separately)

Keep in mind that this is an individual assessment - the related University rules on plagiarism and collusion apply. While it is OK to inspect existing programs for learning purposes, it is not acceptable to reuse program code that was developed by your fellow students or copy code from the Web, even if you reference and amend it before submission. Should you use any graphics assets that are not your own work, please reference their source in the report.

Submission Deadline

The assignment is due on Thursday, 3rd May 2018, 4pm.

Good luck!