Programming Project - Assignment 1
Knowledge & Reasoning 2020/21 UG 3rd Year

[50% of total module marks]

Overview

This is an individual assignment worth **50% of your total Knowledge & Reasoning module marks**. The task is to construct and document a **Checkers game** (draughts) in **Java or Python**, using the object-oriented programming paradigm. The program should present an interactive game board that allows pieces to be moved around by the human player and the AI as per the following basic rules of the game: http://www.indepthinfo.com/checkers/play.shtml
Any of the marking criteria below that are not explicitly specified in these rules can be implemented as you see fit. Your program should produce legal gameplay with reasonable performance, as well as different levels of discernible difficulty, and employ an AI core that is based on your implementation of **Minimax with Alpha-Beta pruning**.

You may think about and start working 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 the term. Until then, use your time to brush up on object-oriented programming generally and on how to build user interfaces specifically. Normally, it will make sense to start programming after the lectures which cover the implementation of Minimax and Alpha-Beta pruning in Java. This will leave you plenty of time to complete the assignment, which is due in week 10.

Marking Criteria

There are a total of 10 assessment areas, split into two groups: Game Internals (76% of total marks) and Human-Computer Interface (24% of total marks). The assessment areas contain further individual criteria for your guidance and are worth different amounts of marks (Total: 100).
When developing your program, you should try to satisfy as many of these as you can.

Note that individual marks will be qualified by your code and the contents of your report, and there may be deductions of varying degree for omissions, inefficiencies, insufficient documentation / explanation, or any mistakes made. You are not primarily marked on the style or quality of your code, but try to adhere to common programming practices to avoid deductions. This includes: Abstraction and modularisation of classes and methods, producing loosely coupled and highly cohesive code, avoiding code duplication, and providing a reasonable level of documentation by using comments.

Game Internals

1. Gameplay **(20 marks)**
    a. Interactive checkers gameplay (Human vs. Computer) of some sort
    b. Different levels of verifiably effective AI cleverness, adjustable by the user

2. Search algorithm **(20 marks)**
    a. Appropriate and efficient state representation

   b. Reasonable successor function to generate AI moves
   c. Minimax evaluation
   d. Alpha-Beta pruning
   e. Appropriate use of heuristics

3. Validation of moves **(16 marks)**
   a. No invalid moves carried out by the AI
   b. Automatic check for valid user moves
   c. Rejection of invalid user moves, with a specific explanation given
   d. Forced capture - an opportunity to capture an enemy piece <u>has to be taken</u>. If there is more than one capturing opportunity in the same turn, the user may choose which one to take.

4. Other features **(20 marks)**
   a. Multi-leg capturing moves for the user
   b. Multi-leg capturing moves for the AI
   c. King conversion at baseline (The king's row) as per the normal rules
   d. Regicide - if a normal piece manages to capture a king, it is instantly crowned king and then the current turn ends.
   e. Some kind of help feature that can be enabled at user's request to get hints about available moves, given the current game state. Sophisticated implementations of this feature may want to employ the AI to make suggestions on optimal moves.

Human-Computer Interface (up to 4 marks each)

5. Some kind of board representation displayed on screen
6. The interface properly updates the display after completed moves (User and AI moves)
7. Fully interactive GUI that uses graphics
   (You are permitted to use tools that make it easier to construct a GUI by generating the necessary code, e.g. layout managers provided with Java IDEs like Eclipse or NetBeans)
8. Mouse interaction focus, e.g., *click to select* & *click to place*, or *drag & drop* (better)
9. GUI pauses appropriately to show the intermediate steps of any multi-leg moves
10. Dedicated display of the rules (e.g., a corresponding button opening a pop-up window)

Report

Your report is a <u>very important</u> element of the assessment and required to confirm that you have written the program yourself and understand the AI functionality implemented in it. Your report should therefore refer to the relevant marking criteria that you have addressed, with detailed explanation given on how, and references to your source code. 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).

Programming Project - Assignment 1
Knowledge & Reasoning 2020/21 UG 3rd Year

[50% of total module marks]

Your report should have a title page with your candidate number on it 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 be come as a single ZIP file that contains the following three parts:

- **Your program files (source code, assets, classes if applicable)**
- **An executable file (e.g., JAR), or clear & specific instructions on how to run your program**
- **Your report as PDF**

You should submit your coursework online on Canvas.

Keep in mind that this is an individual assessment - the related University rules on plagiarism and collusion apply. For example, it is not acceptable to reuse any program code that was developed by your fellow students, or copy code from Web sources, even if you reference them and make amendments before submission.

Submission Deadline

Please refer to your own Assessment Deadlines & Exam Timetable in Sussex Direct for all submission details.

Good luck!