

Introduction to Artificial Intelligence

Summer 2023

EXERCISE 2: Two people deterministic games

Teacher: Tomasz Lehmann, tomasz.lehmann.dokt@pw.edu.pl

This exercise is done in pairs.

1. Exercise details

Variant 2: “Draughts”

Write a program that plays draughts (checkers) with the user on a 8×8 checkerboard. The game continues until one of the players wins or it is a draw.

Algorithm: min-max with alpha-beta pruning.

The game may be played on the console and the interface may be as simple as possible (but of course it does not have to).

Rules of the game:

- 1) Each player begins the game with 12 colored discs (one set of pieces is black and the other white).
- 2) The board consists of 64 squares, alternating between 32 dark and 32 light squares. Each player has a light square on the right side corner closest to him or her.
- 3) Each player places his or her pieces on the 12 dark squares closest to him or her.
- 4) White moves first.
- 5) Moves are allowed only on the dark squares, so pieces always move diagonally.
- 6) Single pieces are always limited to forward moves.
- 7) A piece making a non-capturing move (not involving a jump) may move only one square.
- 8) A piece making a capturing move (a jump) leaps over one of the opponent's pieces, landing in a straight diagonal line on the other side. Only one piece may be captured in a single jump; however, multiple jumps are allowed on a single turn.
- 9) When a piece is captured, it is removed from the board.
- 10) If a player is able to make a capture - the jump must be made. If more than one capture is available, the player is free to choose whichever he or she prefers.
- 11) When a piece reaches the furthest row from the player who controls that piece, it is crowned and becomes a *king*.
- 12) *Kings* are limited to moving diagonally, but may move both forward and backward.
- 13) A player wins the game when the opponent cannot make a move because:
 - a. all of the opponent's pieces have been captured
 - b. all of the opponent's pieces are blocked in

2. Technical details

- a. The solution must be implemented in Python
- b. Please ensure that your code adheres to basic standards of lean coding in accordance to PEP 8. Additionally, it should contain comments in the crucial parts to help with readability and understanding
- c. The clear instruction how to run and test the code should be included
- d. The submission of the final report is mandatory, and the task will not be accepted without it

3. Handing-in guidelines

- a. You should submit the source code of your solution and final report to tomasz.lehmann.dokt@pw.edu.pl not later than:
 - i. **2023.04.03 9:59:59 (Monday) GMT+1 for the Monday group**
 - ii. **2023.04.04 23:59:59 (Tuesday) GMT+1 for the Wednesday group**
 - iii. **2023.04.06 23:59:59 (Thursday) GMT+1 for the Friday group**

Programs delivered after the deadline will not be assessed.

- b. Please include "[EARIN] Exercise 2: Name Surname 1, Name Surname 2" in the title, and do not forget about adding emails of both team members in the email content.
- c. You may get 0-5 pts for this assignment.
- d. In case of questions, please contact me only via MS Teams (on the laboratory channel).
- e. Online assessments are scheduled for the Monday, Wednesday, and Friday groups on **03.04.23**, **05.04.23**, and **14.04.23**, respectively. Further information regarding the details will be provided at a later time.

4. Assessment Criteria

The following criteria will be used to evaluate your work:

- a. Proper implementation of the min-max algorithm: **1.5 points**
- b. Proper implementation of the alpha-beta pruning algorithm: **1 point**
- c. Clean and well-documented code, with clear explanations and well-implemented logic: **1 point**
- d. Final report including results, clear explanation of the programmed solutions and reflections on what was done well and what could be improved: **1.5 points**