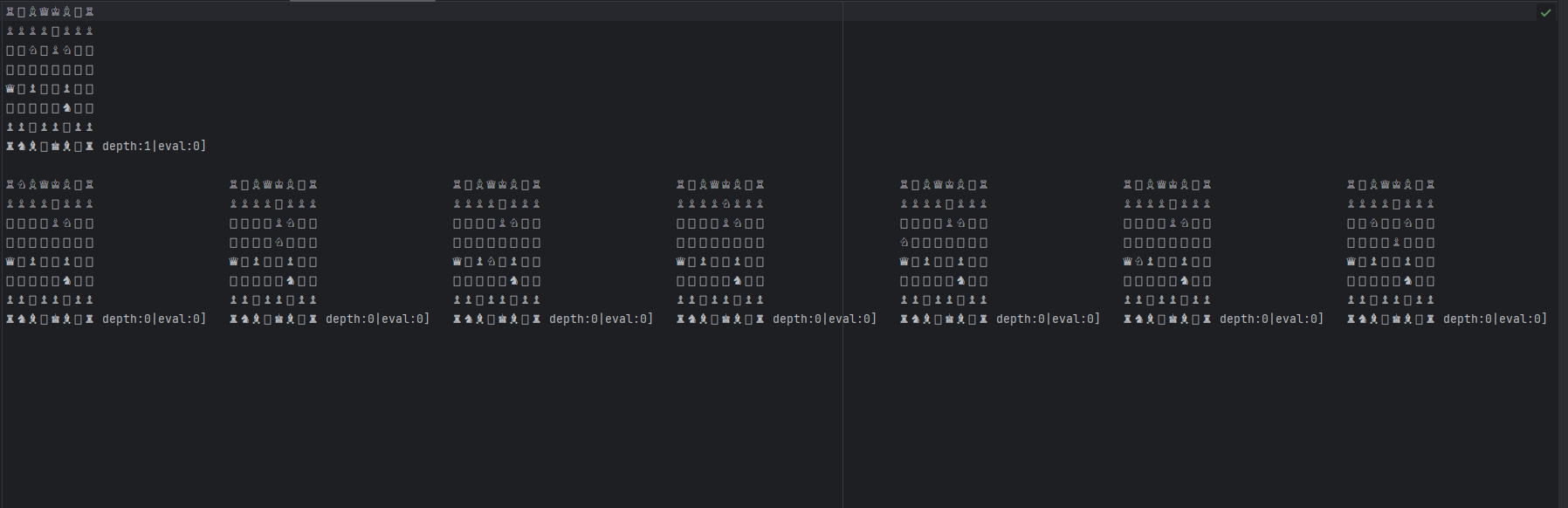
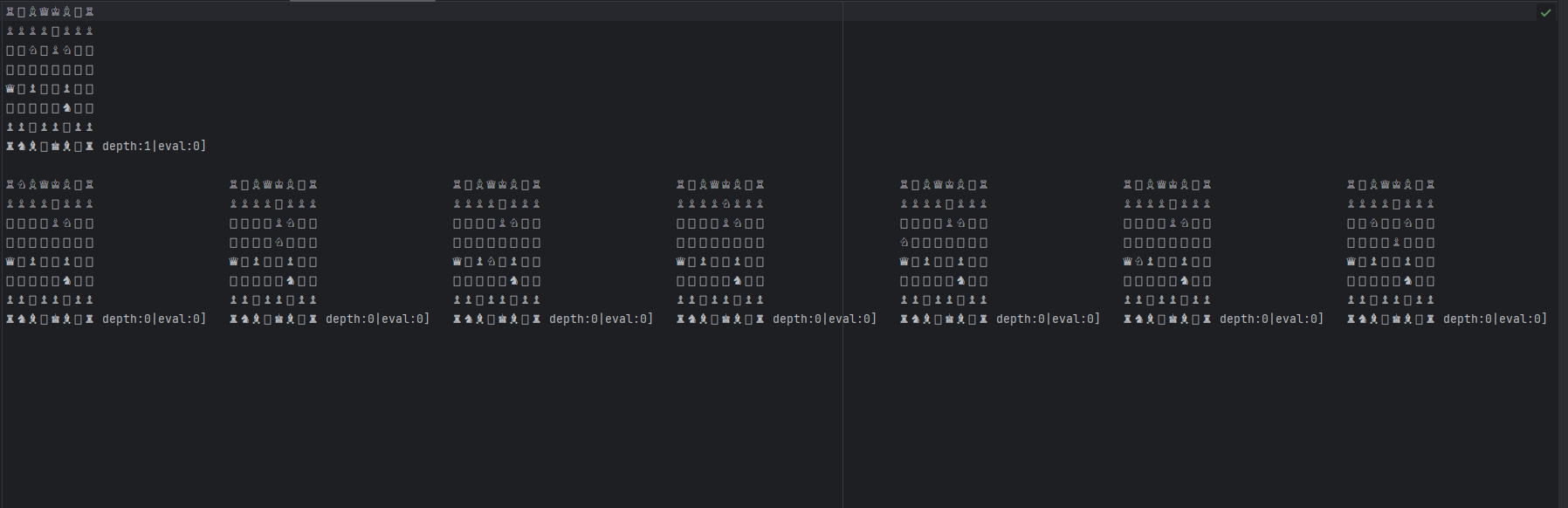
1. Play chess by running “main.py” from “chess-ai-master” folder.
2. Copy the screenshot of your turn in one round



1. Copy the screenshot of “minimax\_tree.txt”



1. Write your analysis

1. Overall: Major chess techniques such as castling and promotion have not been implemented, and the status of the check or checkmate is not informed. The playing level of AI also shows the ability of beginners who are familiar with chess rules.

2. algorithm: A minimax algorithm was used. The minimax function is a function that determines the optimal motion in a given board state. It recursively evaluates the state of the board, searching until it reaches its maximum depth or is\_terminal() state. It alternates between maximization and minimization according to the max\_player parameter.

3. Performance: Minimax algorithms allow for relatively in-depth exploration. The logging functionality allows you to record board status and evaluation values at each stage. The current code does not use Alpha-Beta Pruning, which can lead to low exploration efficiency, and performance can vary significantly depending on the implementation of the evaluation function. In addition, when selecting an optimal solution, it may not guarantee consistent performance, as it makes a random selection among several optimal solutions.

4. Improvements: Implementation of Alpha-Beta Pruning can improve the speed of exploration. More sophisticated evaluation functions can be developed to improve AI's play level. Certain chess strategies can be applied to filter or prioritize movements. This code is sufficient to implement basic chess AI, but it will require improvement of algorithms and advancement of strategies to achieve a higher level of competitiveness in practice.