**Testing, Validation and Evaluation**

To ensure that our RL Agent follows rule and works as intended, we tested and evaluated our Agent using both standard Python testing frameworks and custom scripts.

**1. Unit Testing**

We used pytest to test different components of the game, for example one of the tests included testing the board set up. It verified the board dimensions being 10\*8 grid and checked if the pieces were placed at their correct starting positions. We performed 10 different tests to test different aspects of the game and 2 failed.

**2. Validation Testing**

We also performed validation testing, one of the ways we performed it, included creation of a **TestBoardInitialization** class to validate larger components.

* Consistency across tests by resetting the environment before each validation.
* Checking the consistency to check that the pieces were placed at the correct position.
* Structural correctness using assertions across different modules.

**3. Custom Evaluation Technique.**

Due to our game not being finished, the standard win/loss/draw conditions could not be evaluated. So, we created custom evaluation techniques to evaluate our model. One of the scripts used for evaluation was board\_evaluation.py.

* Board shape and dimension correctness.
* Placement and logic of corner and special pieces.
* Validity of all game pieces across the board.
* Proper initialization of the environment and agent state spaces.

A full report and a summary of pass/failed was printed afterwards.