DEVELOPMENT PDF OF 2048

1) Initial Values (First Version of Player)

- **Algorithm**: Simple Minimax search without pruning, with a trivial heuristic of state.getScore().
- **Search**:
    - Alternated maxPlayer and minPlayer to handle player moves and tile placement.
    - No alpha-beta pruning.
- **Heuristic**: Just used the game score directly (return state.getScore()).
- **Scores**:

  At the very start my score was like some 3000 like that and after running some games it got increased to 6000 but probably not the best.
- **Issues**:
    - Slow due to lack of pruning or expectimax.
    - Weak evaluation: score alone does not represent board quality.
    - Frequent crashes due to bestMove being undefined when time runs out.
    - Incompatible constructor with learning_rate.

2) Next Values (Alpha-Beta Version)

- **Algorithm**: Upgraded Minimax with alpha-beta pruning.
- **Changes**:
    - Added alpha and beta parameters to maxPlayer and minPlayer.
    - Added cutoffs when alpha >= beta for faster pruning.
- **Benefits**:
    - Reduced number of nodes explored.
    - Faster and deeper searches within the same time limit.
- **Scores:**

After adding alpha beta and training the game I got my results better but still not the highest like some 10000 to 15000 probably

- **Remaining Issues**:
    - Still only uses Minimax; doesn't model random tile placements (i.e., not expectimax).
    - Heuristic still weak.
    - Crashes possible if time runs out before a valid bestMove is set.

3) Next Values (Expectimax with Strong Heuristic)

- **Algorithm**: Switched from Minimax to Expectimax, modeling randomness of new tiles.
- **Changes**:
    - Replaced minPlayer with expectiPlayer to average outcomes of random tiles.
    - Integrated a **strong heuristic**:
        - **Corner score**: prefers big tiles in corners.
        - **Empty tiles**: more empty spaces better.
        - **Monotonicity**: encourages smooth increase/decrease along rows/columns.
        - **Smoothness**: penalizes big jumps between neighboring tiles.
        - **Merge opportunities**: rewards immediate merges.
- **Scores:**

  At last my scores were good like they are high and if I keep on playing the game for number of times even my score went high also but sometimes even got some errors and crashing also
- **Move Order**: Added preference for moves ['U','L','R','D'] but filters legal moves.
- **Stats**: Added average depth and branching factor calculations for analysis.

4) Final Values (Cleaned & Compatible Version)

- **Code cleanup**:
    - Consistent naming (best_value, current_best_move etc.).
    - Compact, readable formatting with clear comments.
    - Replaced direct BasePlayer.__init__ with super().__init__ for clarity.

- **Constructor compatibility**:
    - Added learning_rate argument so your agent works with Play.py.
- **Edge cases**:
    - Checks if no actions are available.
    - Ensured fallback bestMove is always valid before calling self.setMove.
- **Performance**:
    - Achieves deeper search with expectimax.
    - Heuristic now aligns with strategies used by top 2048 bots.
- **Compatibility**:
    - Runs without crashing under the given Play.py.