Behavior Contract for Minotaur Card Actions

The following behavior contract outlines the **pre-conditions** and **post-conditions** for actions performed by the active player under the assumption that they have the Minotaur card.

---

1. Move Action

Pre-Conditions

1. The target cell (x, y) must be adjacent to the current worker's position.
   - **Validation:** `Board.isAdjacent(currentX, currentY, targetX, targetY)`
2. The target cell must be occupied by an opponent's worker.
   - **Validation:** `Board.getWorkerAt(targetX, targetY) != null && worker.getOwner() != currentPlayer`
3. The cell behind the target cell (push location) must:
   - Be within the bounds of the board.
   - Be unoccupied.
   - **Validation:** `Board.isWithinBounds(pushX, pushY) && !Board.isOccupied(pushX, pushY)`

Post-Conditions

1. The active player's worker is moved to the target cell.
   - **Validation:** `Worker.getPosition() == targetCell`
2. The opponent's worker is pushed to the cell behind the target cell.
   - **Validation:** `opponentWorker.getPosition() == pushCell`
3. The state of the board is updated to reflect the new positions of both workers.
   - **Validation:** `Board.getWorkerAt(newX, newY) == worker`

---

2. Build Action

Pre-Conditions

1. The active player's worker must have successfully completed its move.
   - **Validation:** `Game.getCurrentPhase() == GamePhase.BUILD`
2. The target cell (x, y) for the build must:
   - Be adjacent to the worker’s current position.
   - Not be occupied by another worker or a dome.
   - **Validation:** `Board.isValidBuildCell(workerX, workerY, buildX, buildY)`

Post-Conditions

1. The height of the tower at the target cell is increased by one level or capped with a dome.
   - **Validation:** `Board.getTowerHeight(buildX, buildY) == previousHeight + 1` or `Board.getTowerHeight(buildX, buildY) == 4`

2. The game phase transitions to the next phase (e.g., END_TURN or an extra build phase if applicable).
   - **Validation:** `Game.getCurrentPhase() == GamePhase.END_TURN`

---

3. Victory Check

Pre-Conditions

1. The move action must have been completed.
   - **Validation:** `Game.getCurrentPhase() != GamePhase.MOVE`
2. Victory conditions must be checked for both the active player and the opponent.

Post-Conditions

1. If the Minotaur's move fulfills the default victory condition (worker reaches tower height 3), the game ends.
   - **Validation:** `Game.isGameEnded() == true && Game.getWinner() == currentPlayer`
2. If no victory condition is met, the game continues to the next phase.
   - **Validation:** `Game.isGameEnded() == false`

---

4. Tradeoffs and Assumptions

1. **Assumption:** The opponent's worker can always be pushed as long as the push location is valid.
2. **Tradeoff:** The rules rely on the state validations at the board level, ensuring that illegal moves are caught early in the game logic.

Rationale

The behavior contract ensures that:

- The Minotaur card’s unique abilities are implemented with clarity and precision.
- Illegal moves and builds are prevented through strict pre-condition checks.
- Post-conditions verify that the game state transitions correctly after each action.