Suppose learning updates occurred after all moves, including exploratory moves. If the step-size parameter is appropriately reduced over time, then the state values would converge to a set of probabilities. What are the two sets of probabilities computed when we do, and when we do not, learn from exploratory moves? Assuming that we do continue to make exploratory moves, which set of probabilities might be better to learn? Which would result in more wins?

SOLUTION: If learning updates occurred after all moves, including exploratory moves, and the step-size parameter is appropriately reduced over time, the state values would converge to a set of probabilities. These probabilities would represent the likelihood of winning the game from each possible board state.

When we do not learn from exploratory moves, the probabilities would be biased towards the moves that are most likely to lead to a win, based on the current learned values. This would result in a more exploitative playing style that favors safe moves and avoids risks.

On the other hand, if we do learn from exploratory moves, the probabilities would be more evenly distributed and would reflect the algorithm's exploration of different strategies. This would result in a more exploratory playing style that takes risks and tries different strategies.

Assuming that we do continue to make exploratory moves, it would likely be better to learn from them, as this would allow the algorithm to explore different strategies and potentially discover better ones. While an exploitative playing style may result in some wins, it is more likely to get stuck in suboptimal moves and miss out on better strategies.

Therefore, learning from exploratory moves would likely result in more wins overall, as it would allow the algorithm to discover better strategies and make more informed decisions. However, it is important to note that the step-size parameter must be appropriately reduced over time to ensure that the state values converge to the correct probabilities and that the algorithm does not get stuck in a local optimum.