Part 5 - Heuristics in the Adaptive A* [15 points]: Implement and compare Repeated Forward A* and Adaptive A* with respect to their runtime. Explain your observations in detail, that is, explain what you observed and give a reason for the observation. Both search algorithms should break ties among cells with the same f-value in favor of cells with larger g-values and remaining ties in an identical way, for example randomly.

Across 100 grids with dimensions 101 X 101, our Adaptive A* algorithm took, on average, 2124.34 expansions to reach the target. Forward A*, on the other hand, took an average of 2203.06 expansions to reach the target, showing that Adaptive A* is 3.57% more efficient.

It is unsurprising that Adaptive A* is faster because it improves the h value after each iteration of A* by making a more informed guess. Adaptive A*'s estimate of the goal distance for a state is found by taking the goal distance of the start state and subtracting the distance from the start state to state s. As proven in the assignment write-up, this is admissible. Adaptive A* updates the heuristic value for all states expanded during an A* search and increases them, therefore makes them less likely to be visited in future A* searches. In all, this makes the Adaptive A* much more focused and more efficient. This updated heuristic also can only ever increase the h value of a cell, not decrease it. This guarantees that the algorithm never makes the agent more likely to revisit a cell during A* and only can help it or be neutral.

That being said, the agent does not know the entire grid at the start and therefore occasionally paths that seem better based on Adaptive A* actually end up being marginally worse. The agent may end up getting stuck in a hole which it was unable to foresee, leading to the “better” h values actually resulting in being less efficient. This, however, does not happen nearly often enough to offset the increase in efficiency that Adaptive A* offers.