The artifact that I selected for the algorithm and data structures enhancement was the DEEP Q-learning pirate game project from CS- 370. The original program was a reinforcement learning environment where the pirate agent is tasked to explore a grid in search of a treasure. The agent learned through reward feedback. It used a DQN structure with simple state representation, limited exploration logic, and little to no long-term memory or stability mechanisms. It did demonstrate the fundamentals of Q-learning, but it lacked any advanced algorithmic techniques that can make reinforcement learning more stable and efficient. I have chosen to enhance the project by adding a target network, reshaping the reward system, improving the training loop, and adding some evaluation metrics so that the model's learned performance is measured more accurately.

I chose this artifact for the EPortfolio because it will show off my ability to work with algorithmic design and data-driven decision structures. Reinforcement learning is built on algorithms, state modeling, and iterative computation, which makes it an ideal project to highlight competency in this category. The enhancements that I made to the project require a deeper understanding of how the learning agents store, process, and evaluate the state action pairs. It also shows knowledge of how algorithmic stability will impact the quality that we see from the trained model. I added a replay buffer using a deque structure so that the agent was able to learn from past experiences and not just the immediate transitions. I also added a target network to prevent unstable oscillations in Q-value updates, which makes the project reflect a more modern look found in projects like this. The environment was improved with distance-based reward shaping and penalties for revisiting past cells. These enhancements show algorithmic reasoning, data structure usage, performance optimization, and model evaluation techniques that are a step above what was in the original project.

In the planning I intended to demonstrate my ability to design and evaluate computing solutions using algorithmic principles while balancing the trade-offs in performance, structure, and computational efficiency. I believe I have met and, in some cases, exceeded my initial plans. I was able to implement all of the enhancements including replay buffers, performance tracking, and tunable parameters, and even expanded the project with more improvements like a greedy evaluation function and better reward engineering. This turned the simple academic sample into something much closer to a real reinforcement learning project. No changes were necessary to my coverage plan.

While working on this project, I learned some valuable information on how sensitive these systems are to the different design choices. Small adjustments to the reward structure had a noticeable impact on how quickly and effectively the agent learned. I also learned more about balancing algorithmic tradeoffs like how big the replay buffer should

be, how frequently the target network should update, and how to structure the penalties so the agent avoids loops and does not get stuck. A challenge I faced was with testing different combinations until I could get consistent behavior from the agent. It was also challenging to restructure the training loop so that it was clear and efficient when using the new replay logic. Through these enhancements I was able to see how algorithmic design impacts computational behavior, and I saw my confidence in improving these systems increase.