**COSC 4368 - Group Project**

*PD World*

Alexander Drake, Sara Ciesielski, Julio Santos, Khalyl Smith

**Abstract**

The Pickup-Dropoff World (PD World) problem seeks to teach two agents to pick up objects and deliver them to dropoff spaces efficiently using Reinforcement Learning. This can be achieved by teaching agents to navigate toward attractive paths through training them using Q-Tables. We’ll be utilizing and comparing two Reinforcement Learning methods to calculate Q-Values, Q-Learning and SARSA. We’ll also run four experiments to evaluate how our agents perform in finding solutions. The main focus is to see how Reinforcement Learning helps AI agents adapt to worlds with changing states and potentially non-finite state spaces.

**Introduction**

Reinforcement learning is among the three primary machine learning paradigms that aims to train machines to understand their environment and take actions that maximize cumulative rewards. It does this by utilizing the Markov Decision Process (MDP). MDP provides a framework for modeling decision making, but in the context of reinforcement learning, it trains an AI system to take actions, evaluate its rewards, and recognize patterns to find satisfactory solutions. However, with this limited feedback, essentially leaving the agent to learn on their own, can AI develop sufficient strategies to solve problems through reinforcement learning? We’ll be answering this question by training AI agents to solve the Pickup-Dropoff World problem using reinforcement learning. The purpose of this paper is to bring insight on the capabilities of reinforcement learning agents as well as administer reinforcement learning agents in order to create the most efficient path in a 2-agent setting. This paper also provides an overview of how using 4 different experimental evaluation methods will show the proficiency of completing the transportation performance puzzle.

**The Problem**

Pickup-Dropoff World (or PD-World) is a problem that places two agents in a finite state-space with pickup and dropoff spaces for delivering items. Agents will navigate the world to pick up items one at a time and deliver them to the dropoff spaces. Reinforcement learning assists in training agents to seek after attractive paths to maximize cumulative rewards. For this problem, we’ll be implementing and evaluating two fundamental reinforcement learning algorithms, Q-Learning and SARSA. The main takeaway for our solutions are to evaluate the AI agent’s performance and most importantly, their adaptability to the world they are in and for worlds that might change while the agents are working to solve the problem.

**Q-Learning vs. SARSA**

Q-Learning and State-Action-Reward-State-Action (SARSA) are two fundamental algorithms in reinforcement learning that assists agents in learning their environment and ultimately decide on the best action to take to maximize their rewards. Both methods are tasked to update a Q-Table that represents the potential future rewards that can be acquired by taking a certain action during a specific state. Their purpose is to help agents focus on maximizing their long-term rewards. Q-Learning and SARSA utilizes the Bellman Equation and Temporal Difference formulas to calculate Q-Values, however where they are different is their policies.

Q-Learning is a greedy policy because it chooses the maxima state-action pair from the next state. The greedy policy is also known as an off-policy since the learning agent evaluates the Q-Value according to an action that is derived from another policy. Meanwhile, SARSA is known as a behavior policy (or on-policy), where the learning agent evaluates the Q-Value according to the current action. In terms of performance, Q-Learning’s best performance is optimal while SARSA’s is not [1]. This is because Q-Learning learns an optimal greedy trajectory while SARSA doesn’t. This can lead to SARSA having sudden drops in performance, however over the smoothed performance, SARSA tends to perform better than Q-Learning in the long run [1]. Q-Learning often evaluates actions in undesirable states within the learning agent’s trajectory more than SARSA, which can lead to the agent receiving more negative rewards than when using SARSA. Although this might suggest that SARSA is the most optimal, both can prove to solve reinforcement learning problems efficiently. It’s best to evaluate both and determine which is better for a specific problem and for specific state-space sizes.

**Q-Table Structure**

Our method B Q-table had the following parameters in the state space: a boolean representing whether or not the agent had food, all permutations of pickup or dropoff square availability, and the current position of the agent. Since we don’t need to worry about dropoff squares when we don’t have food and don’t need to worry about pickup squares when we do have food, we only stored the pickup square availability in the has\_food = False part of the Q-table and only stored dropoff square availability in the has\_food = True part of the Q-table. This resulted in a total of 500 states:

States =

(has\_food=True) \* dropoff\_availability \* agent\_position

+ (has\_food=False) \* pickup\_availability\_permutations \* agent\_position

States = 1 \* 2^4 \* 25 + 1 \* 2^2 \* 25 = **500**

Method A had significantly more states due to the requirement of storing the other agent’s position in the state space as well. This effectively multiplied the size of the q-table by 25 since there were 25 possible positions for the other agent to be in. The total number of states was **12,500.** Technically, the amount of states could be reduced slightly due the agents being unable to occupy the same square, so the state space could have actually been multiplied by 24 instead of 25 giving us 12,000 states, but we decided not to include this consideration in our Q-table to simplify the construction and readability of the table. It also has no effect on the performance of the agents since these extra states will never be visited.

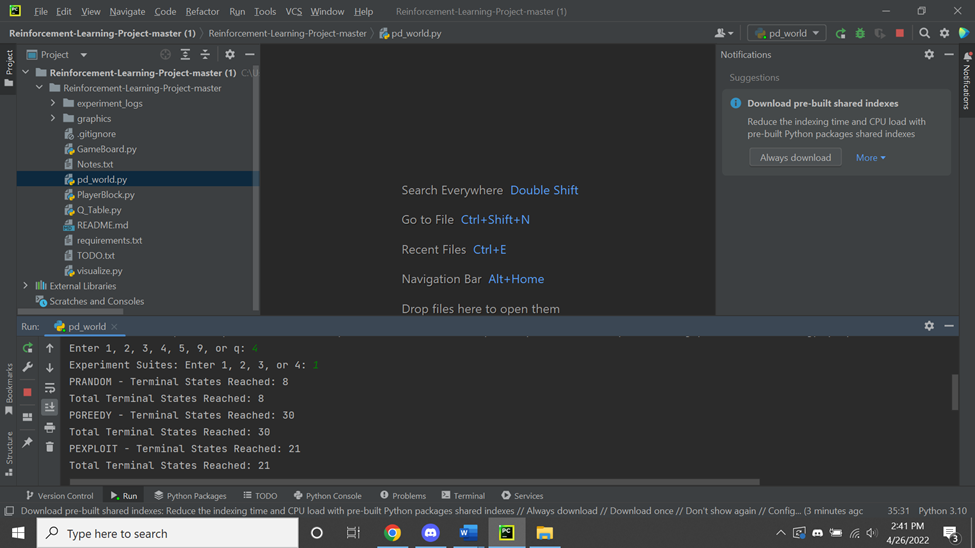
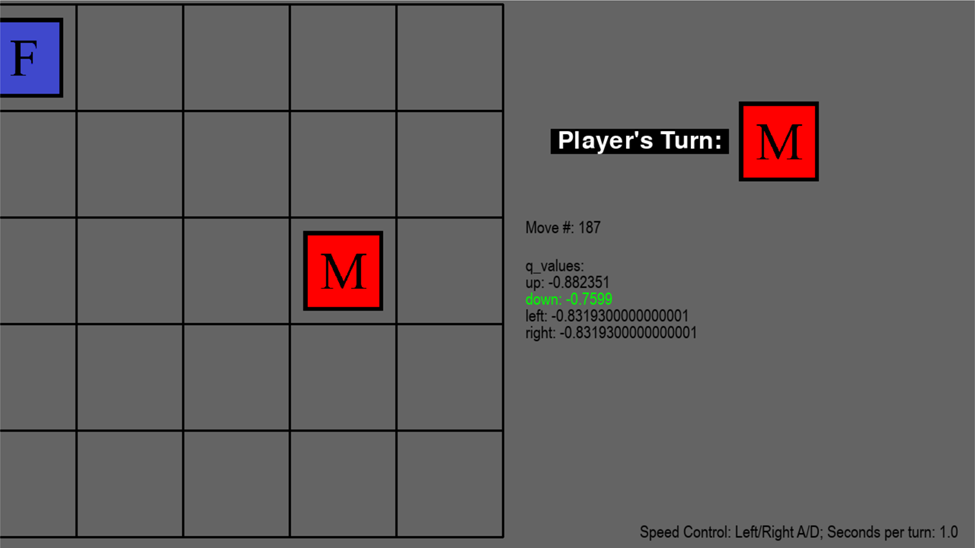
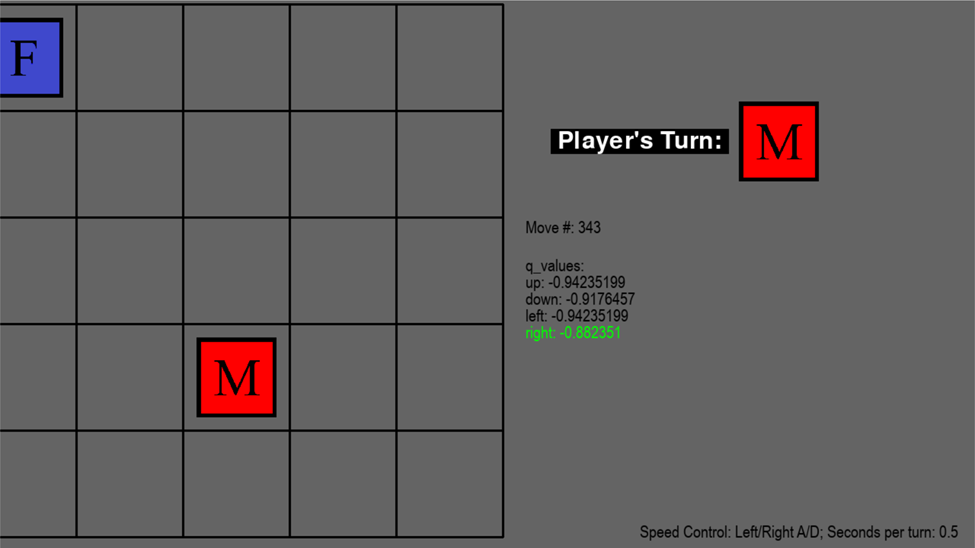
**Methods**

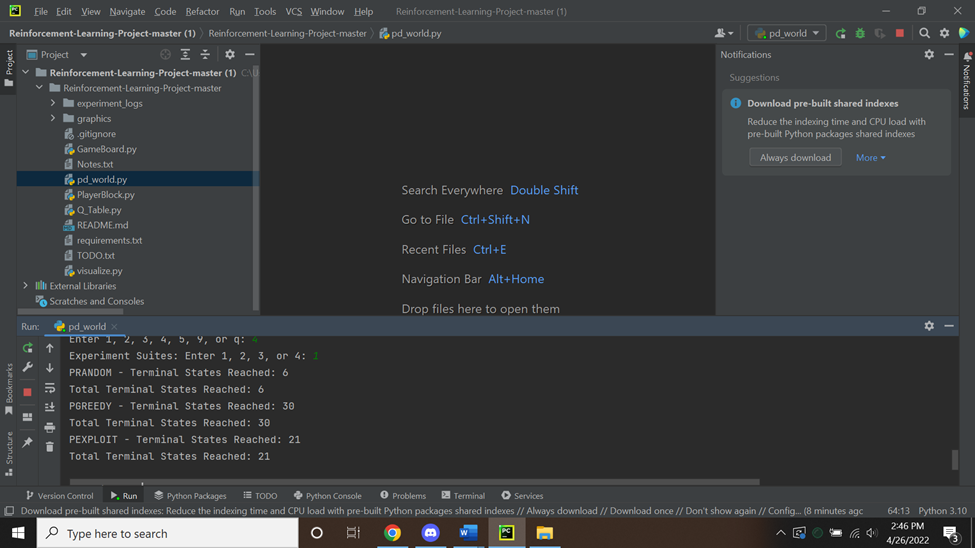
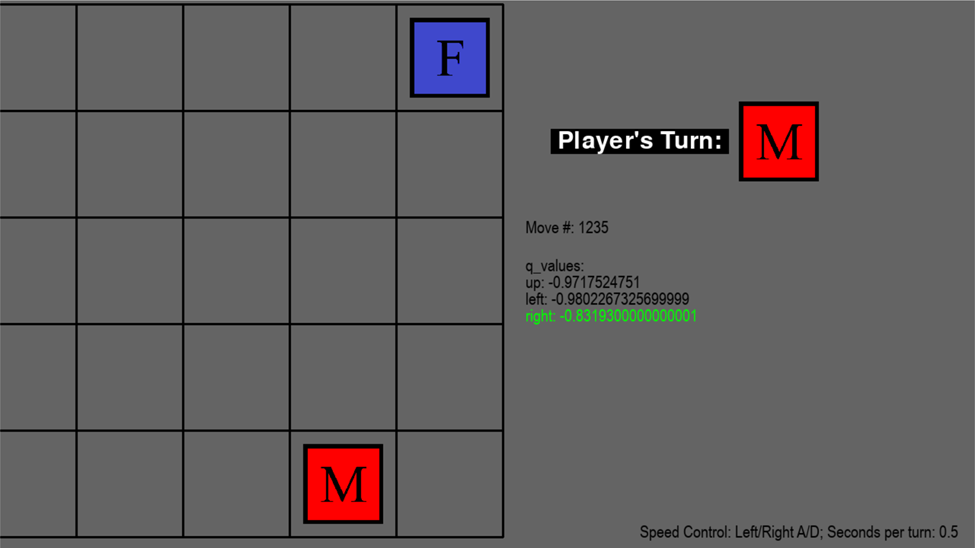
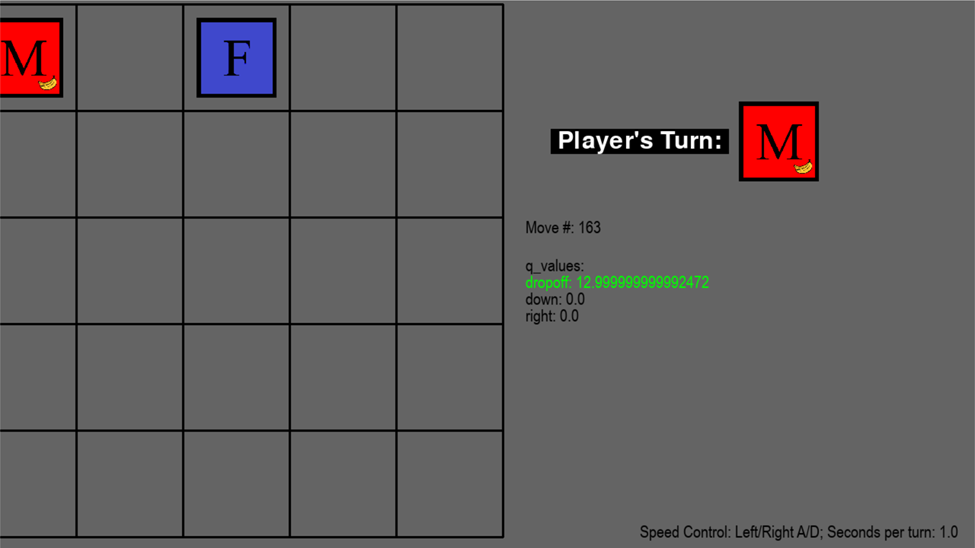
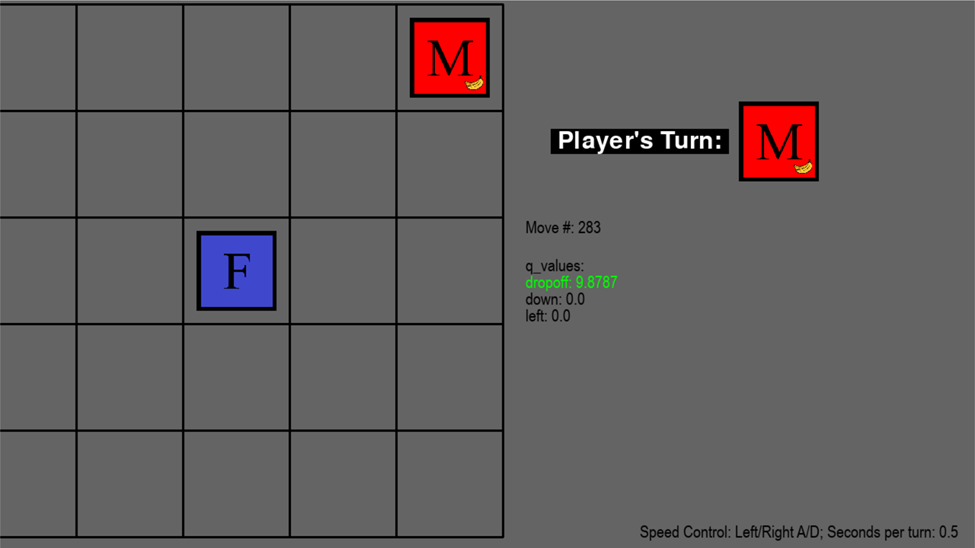
We tried both methodologies outlined in the rubric. Our primary focus was method B which involved both agents using the same Q-table. We also implemented method A in which each agent used their own separate Q-tables and stored the other agent’s position in the state space. The following experiments described are using method B, and we will touch on experiments 2 and 3 with method A near the end.

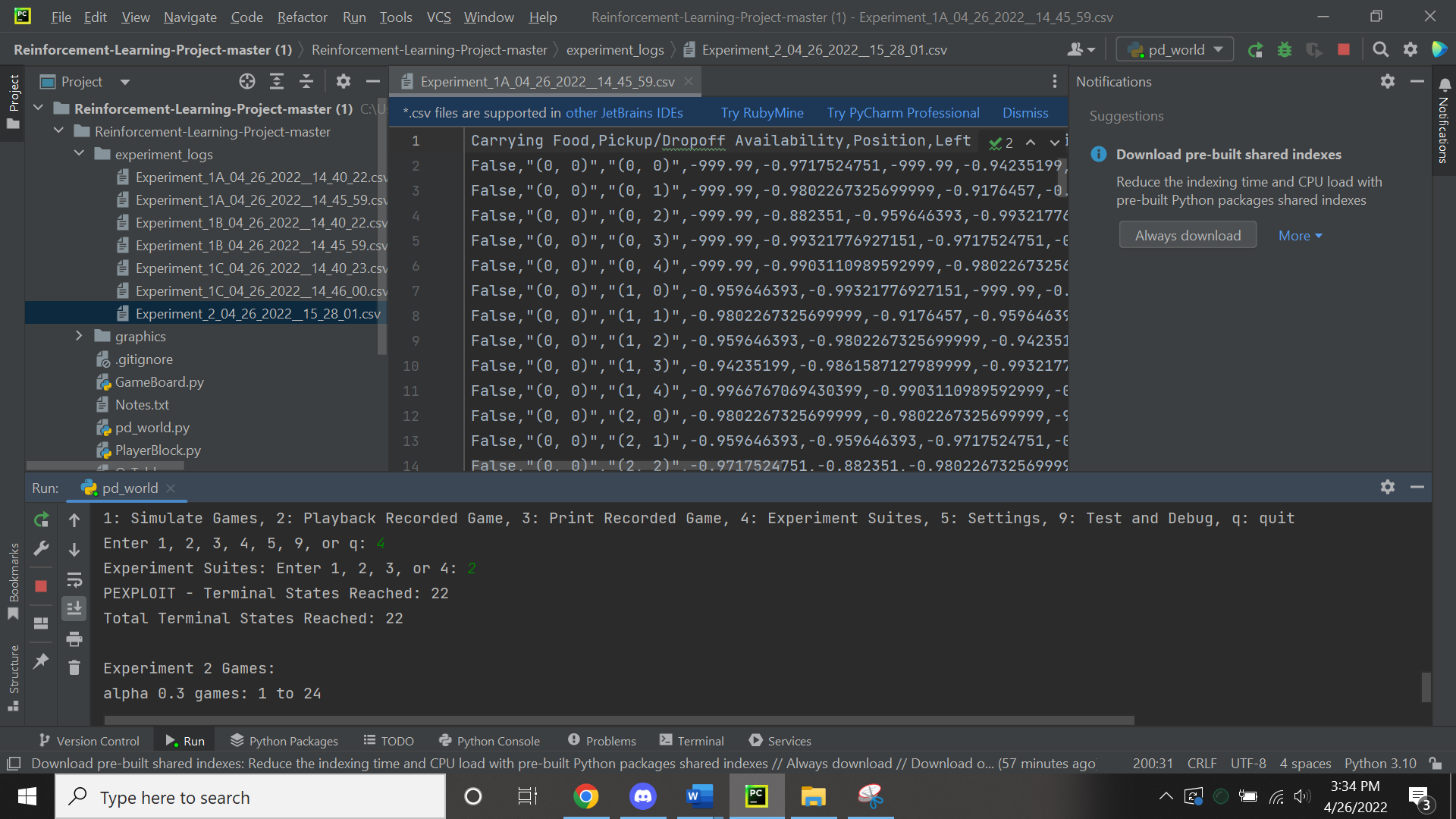
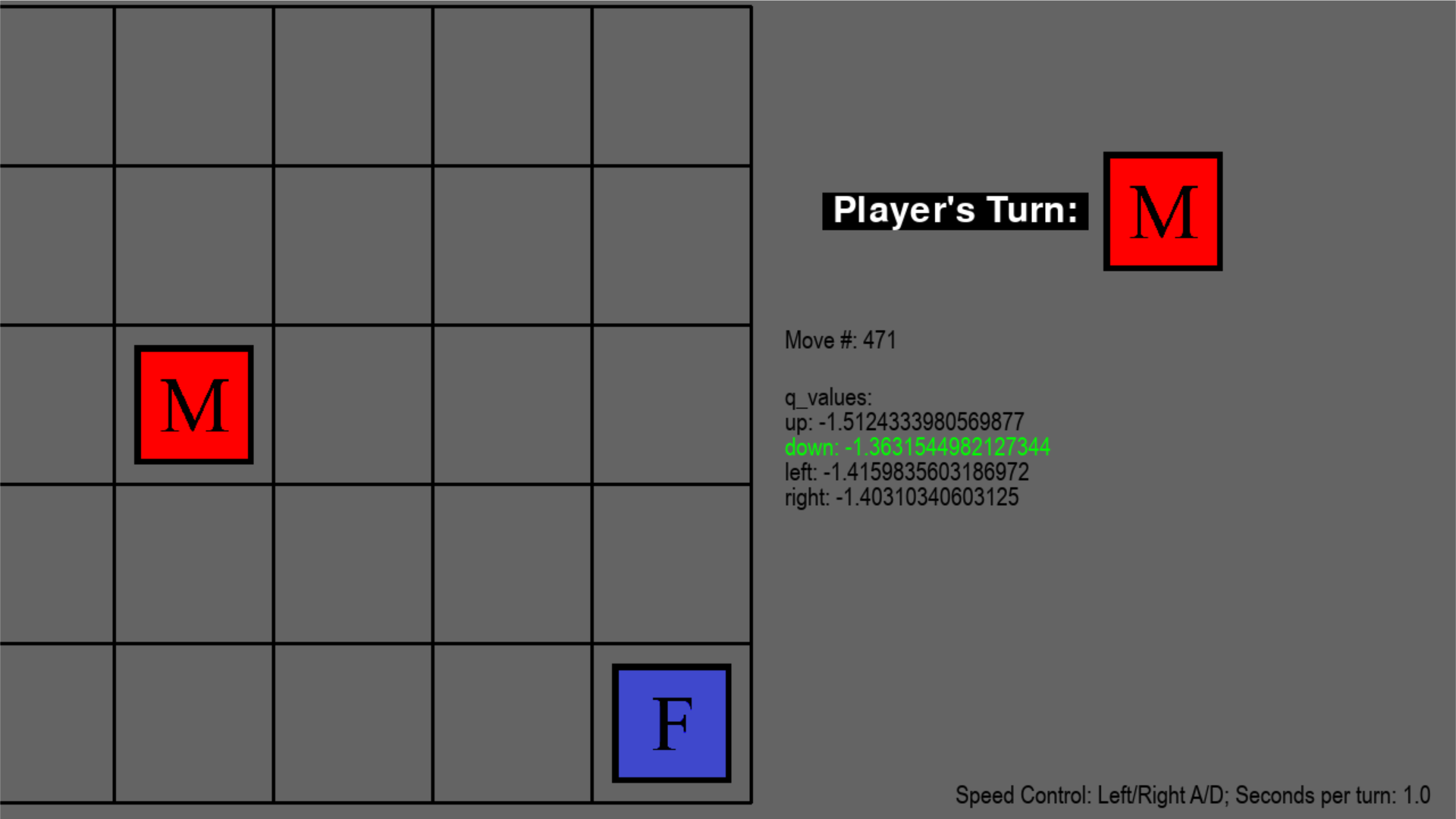
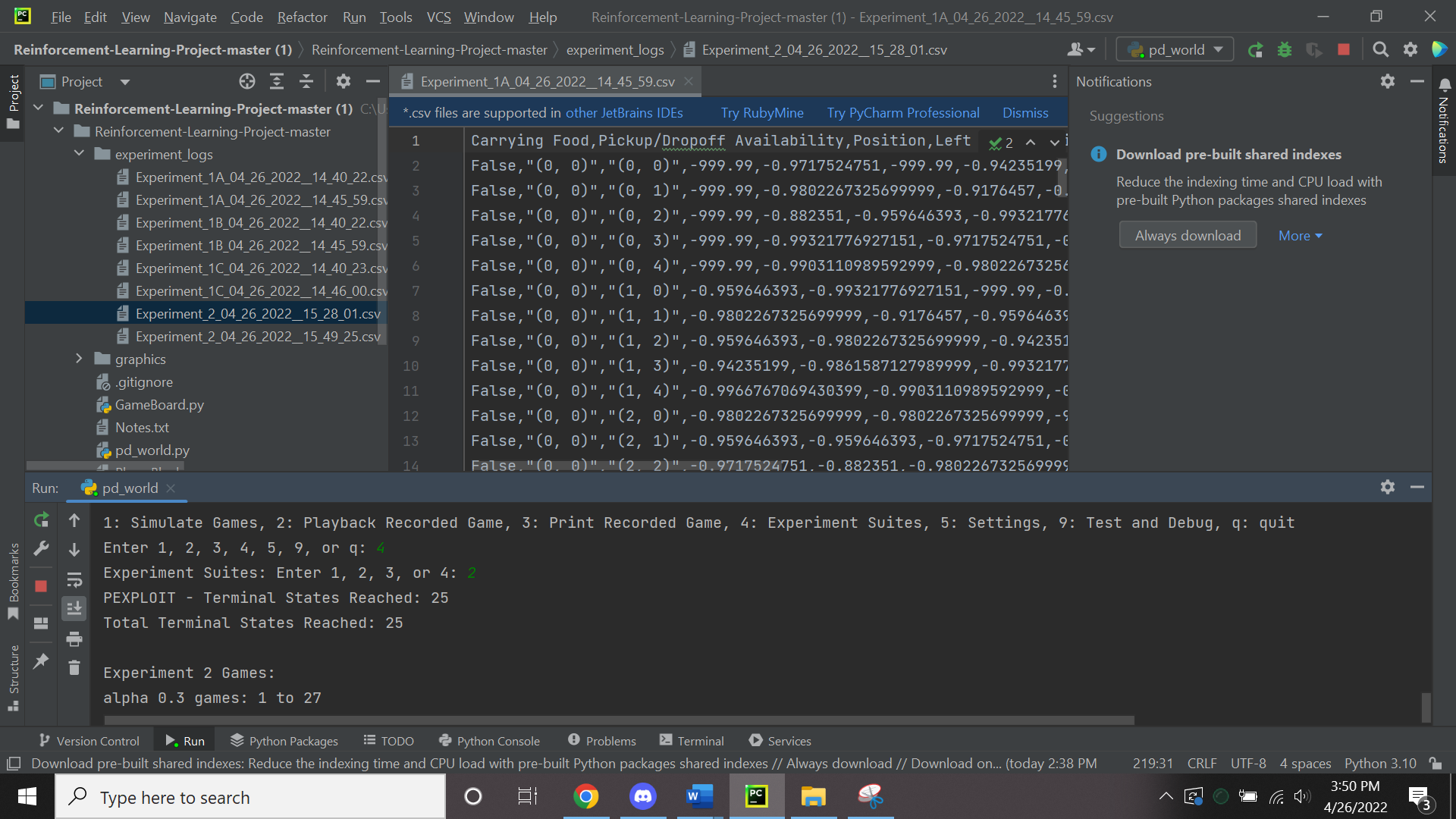
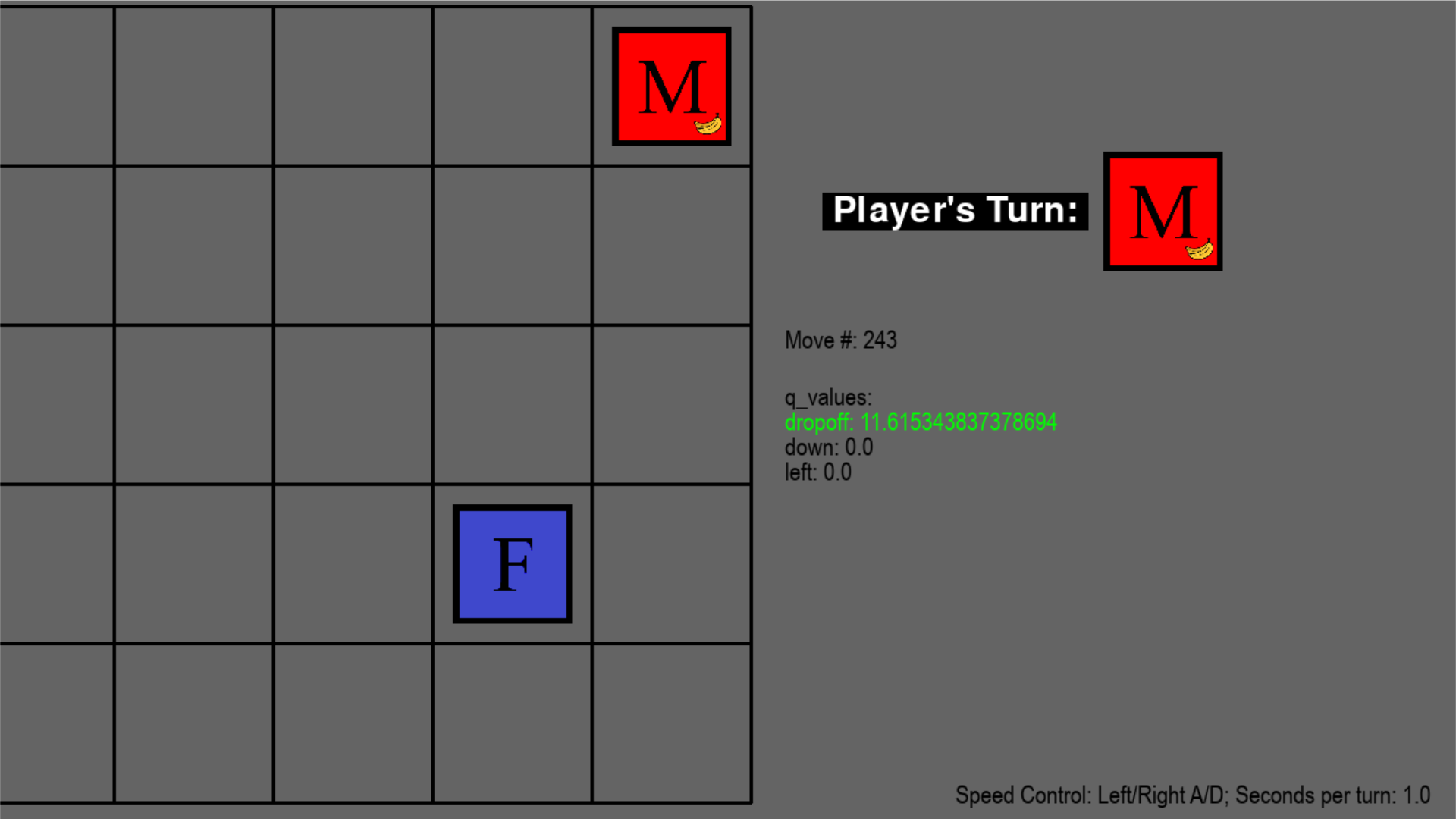
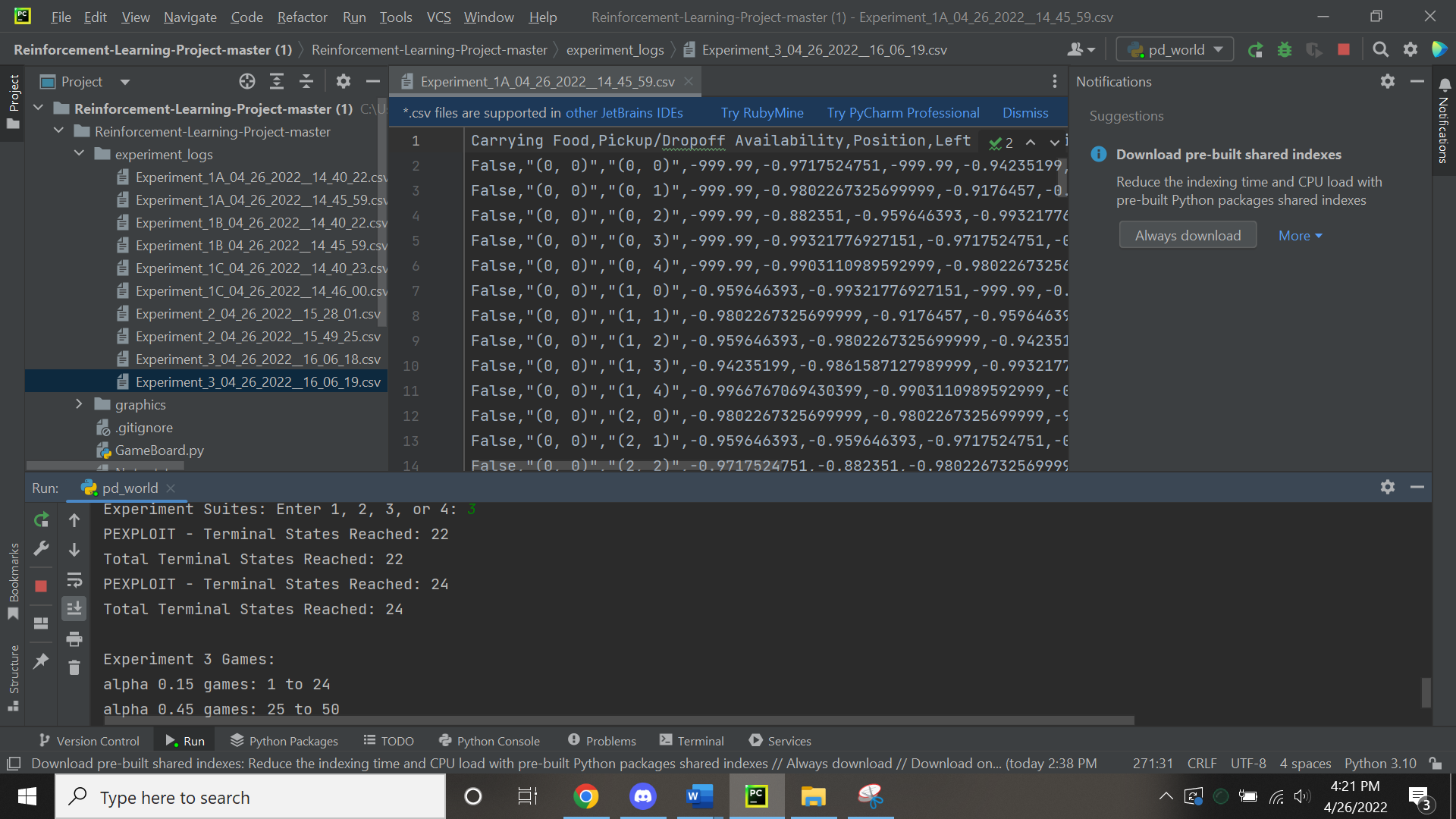
Definition of a step: we defined a step as either one of the agents taking one move. In other words, if both agents move, that counts as two steps, not one. We also counted picking up and dropping off as a separate step, even though it happens no matter what when available. (I.e. an agent without food does not enter a pickup square and pick up on the same turn.)

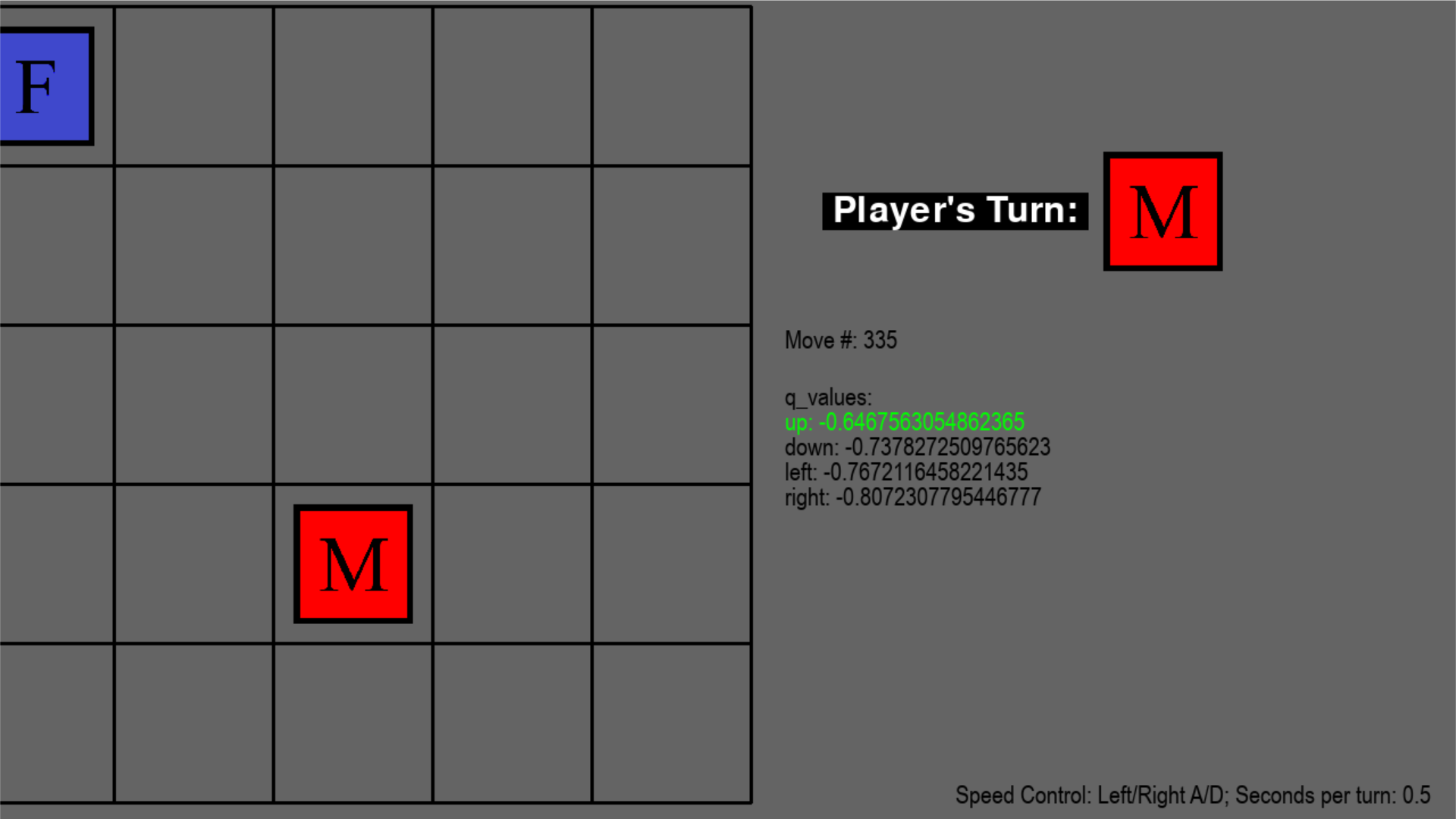
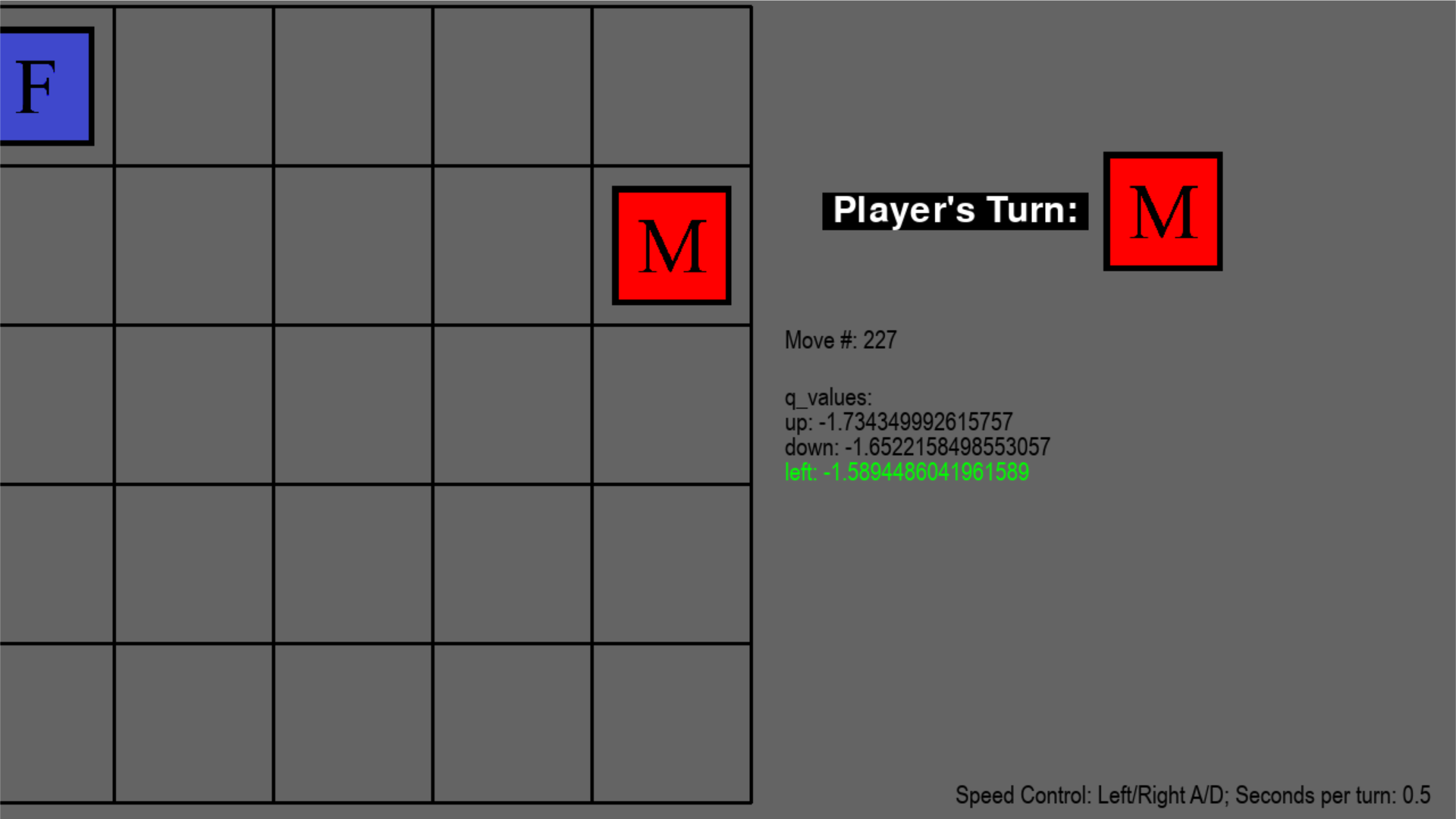
**Experiments**

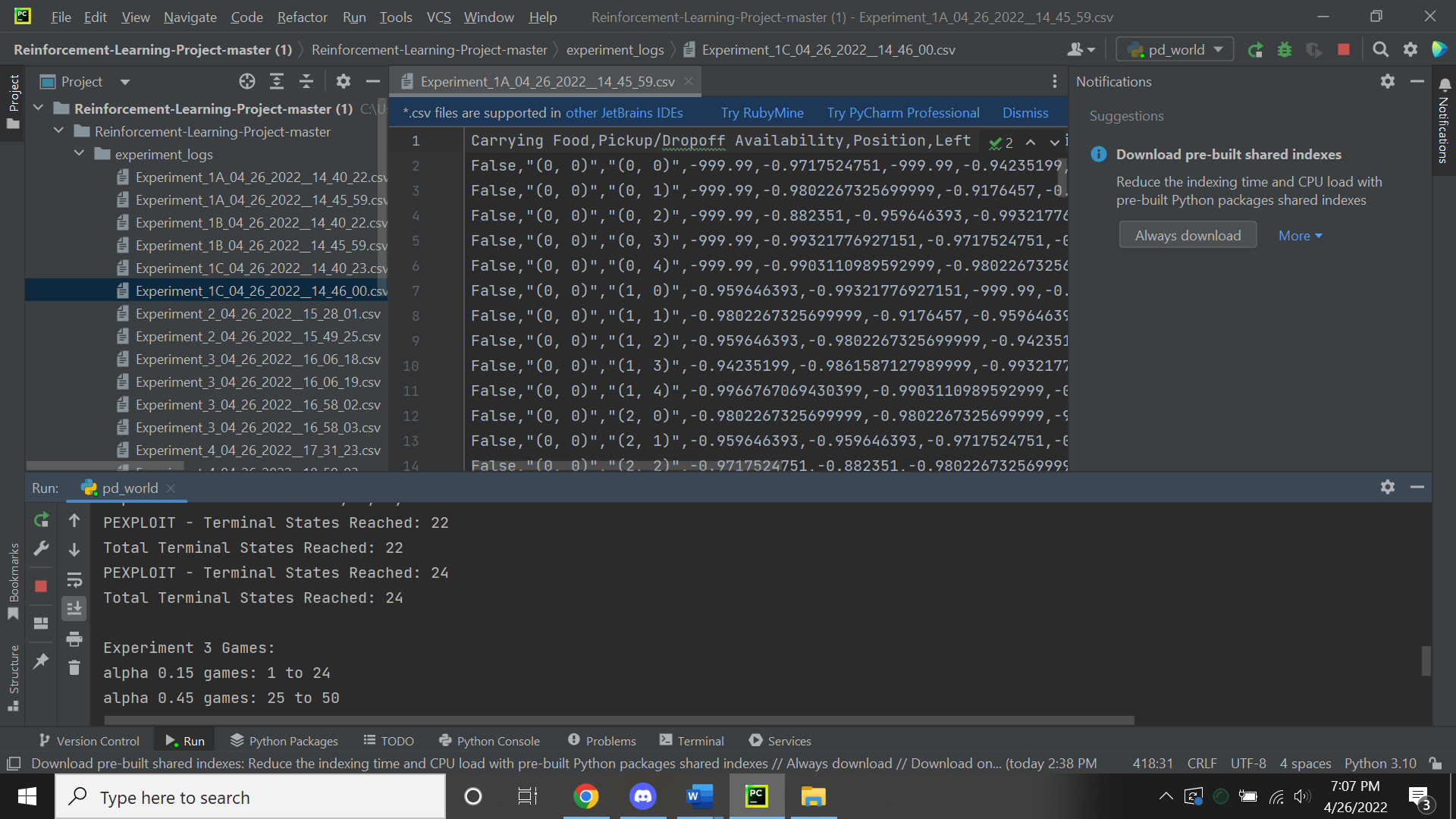
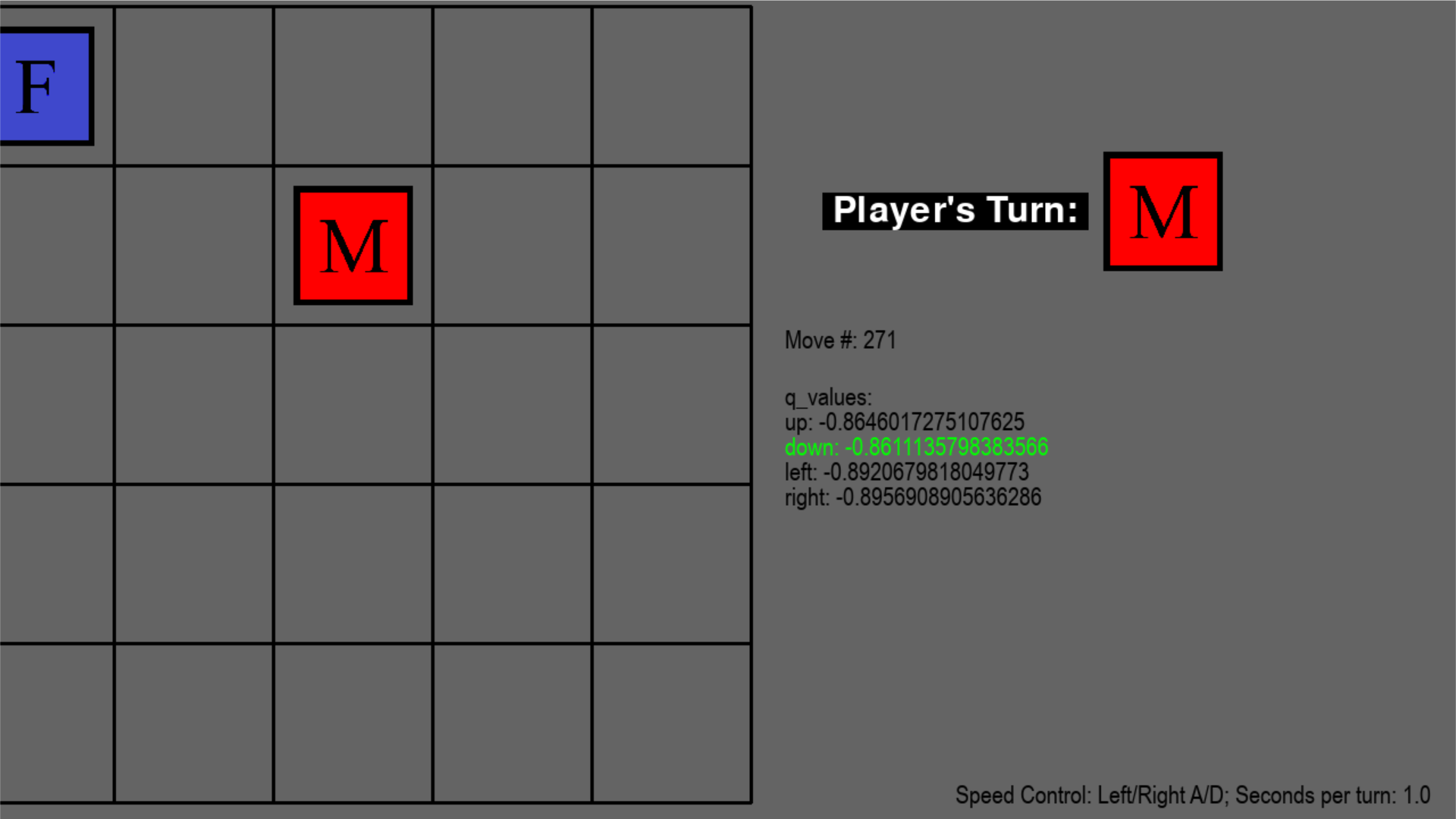
For this section, we’ll be running four experiments to train two AI agents in the PD-World and evaluating their performance and coordination. In addition to Q-Learning and SARSA, we’ll also be using three different policies that can affect the choice of each agent during their turns, PRANDOM, PEXPLOIT, and PGREEDY. These policies will affect what operator to choose when pickup or dropoff isn’t applicable. For context with the results, with our definition of steps outlined in the Methods section, we estimate that with perfect, optimal, and synergistic play from both agents, the minimum number of steps is between 150 and 160.

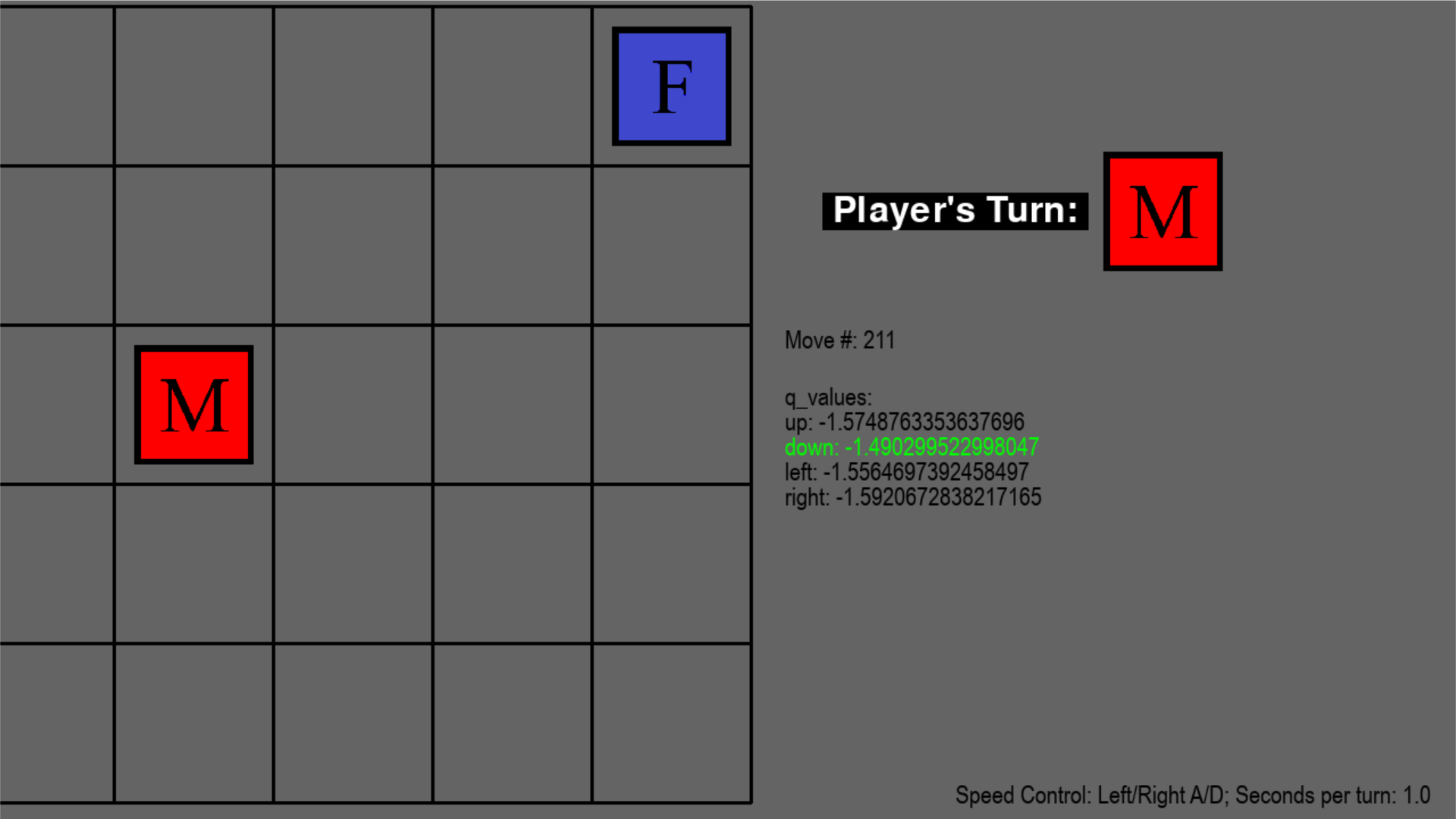
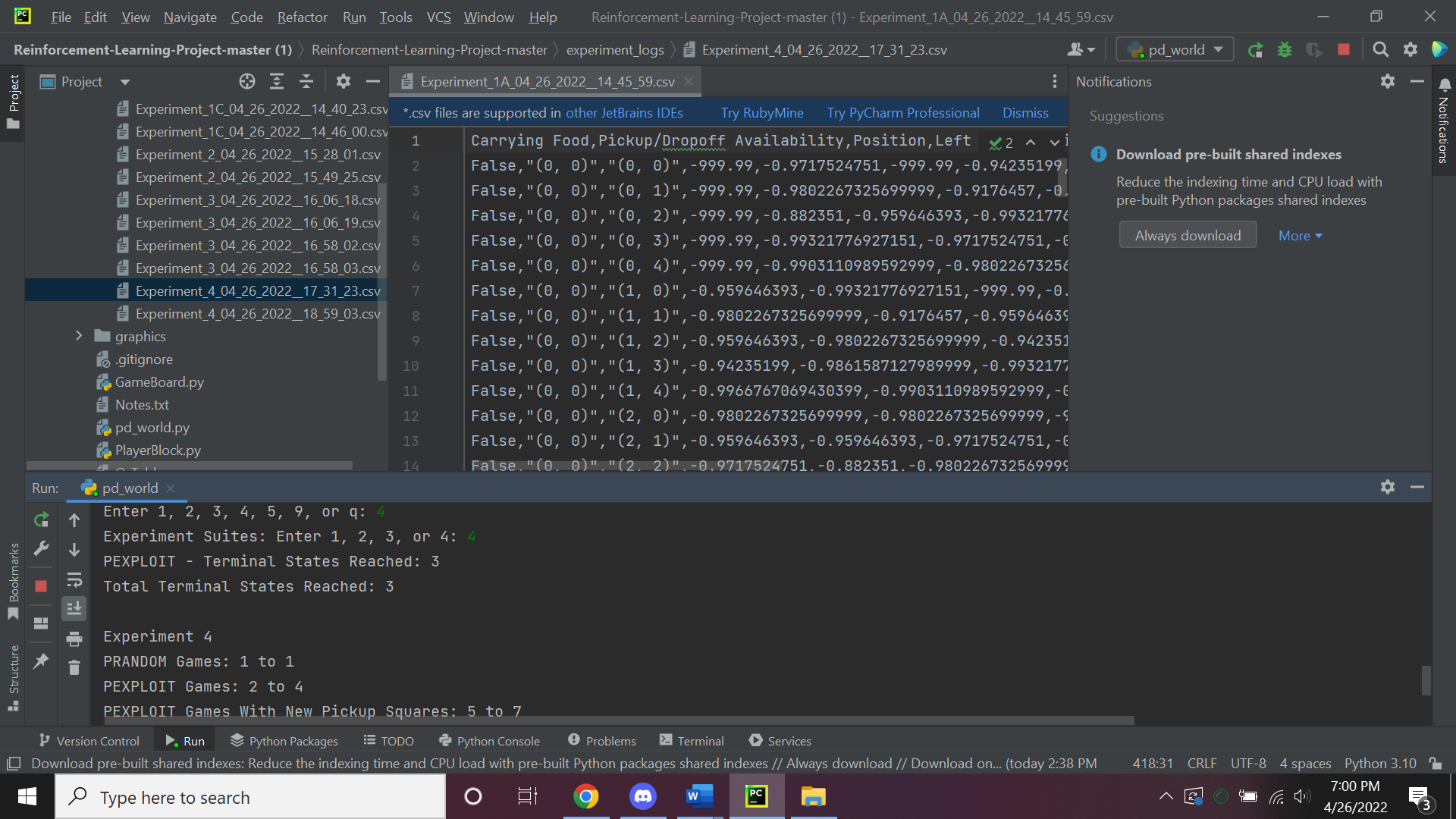
**Experiment 1: Result 1**Terminal Results:**  
PRANDOM:** Pathing had little to no efficiency and both agents did not take the shortest paths to the goals. There were little to no blocking paths for each other. Around half of the moves was spent to get the bananas in the last available goal. By far, this experiment took the longest to complete. The total moves to complete this puzzle was 1084. (image below)  
**PGREEDY:** Very efficient pathing for both male and female agents. They both took the shortest paths in order to get the banana to the goal. Pathing was repetitive and there was no blocking as well in order to deliver the banana. Both agents remembered the paths they took to efficiently deliver bananas. This experiment completion time dropped dramatically. The fastest of the experiments by far. The total moves to complete this puzzle was 188. (image below)  
  
**PEXPLOIT:** Both agents took efficient pathing at the beginning. Took a bit longer to complete the puzzle. Massive increase in blocking each other from placing bananas in the goal. Male agent blocked the female agent more. The 2nd fastest completion for the puzzle. The total moves to complete this puzzle was 344. (image below)   


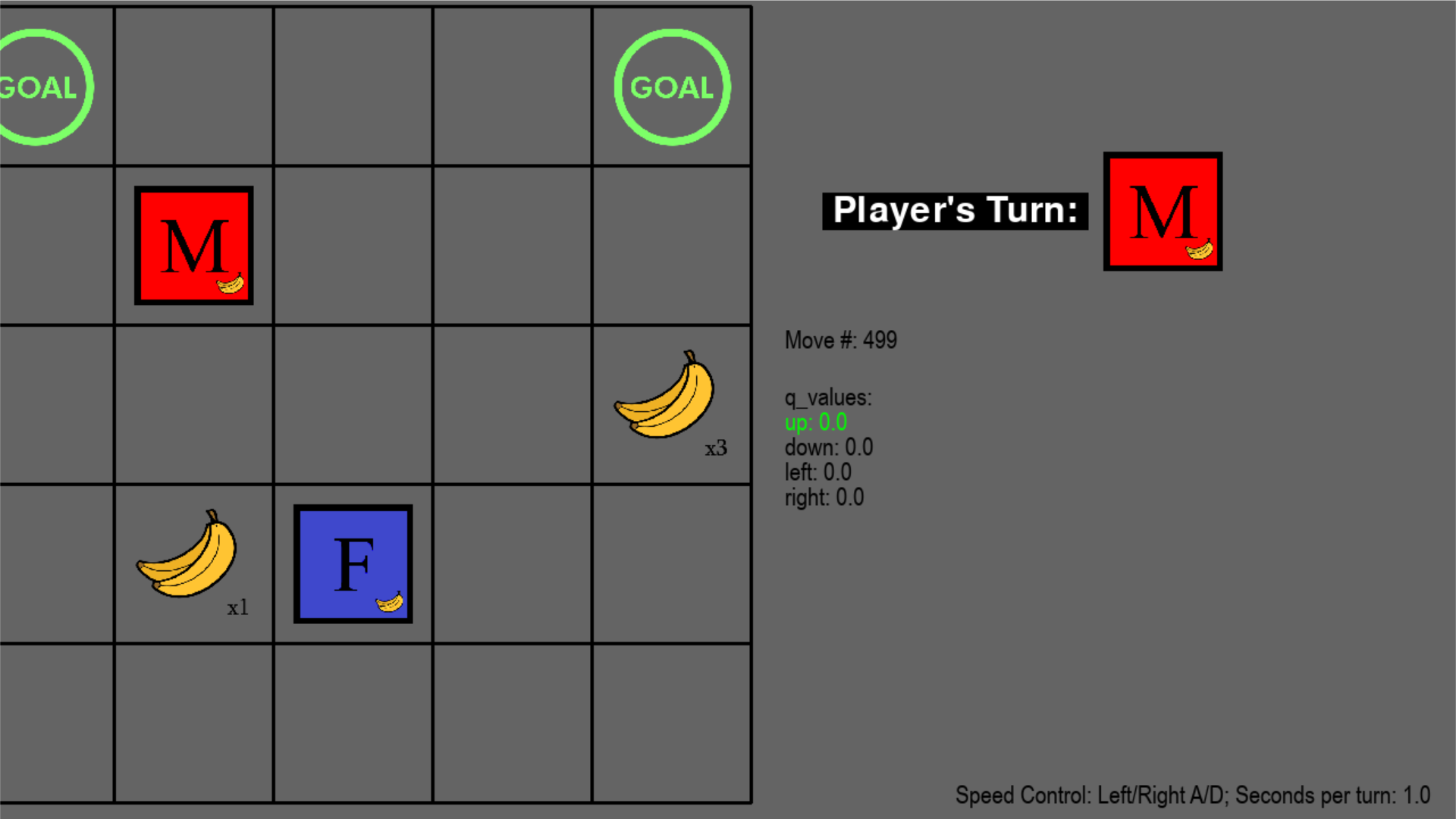
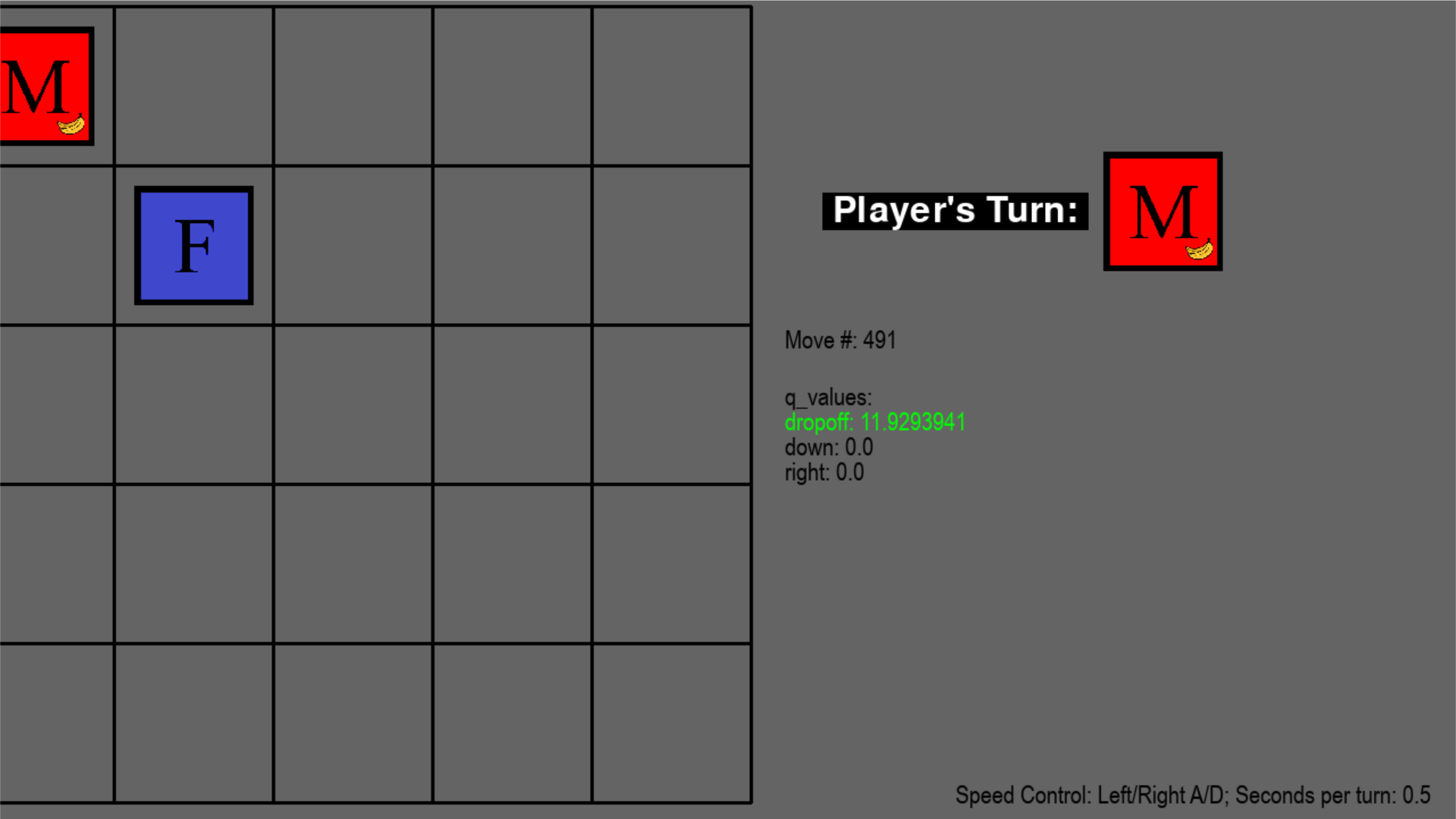
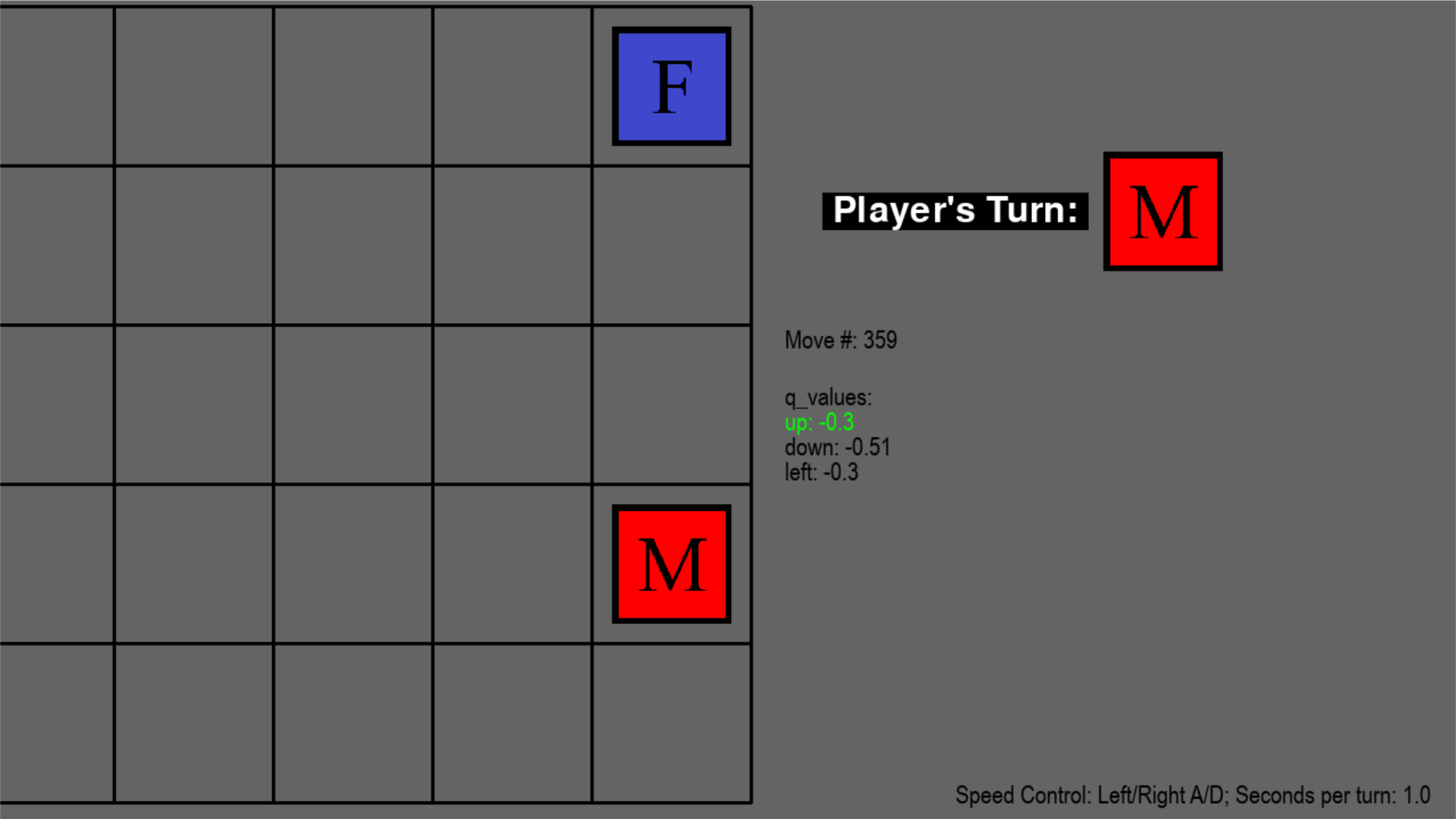
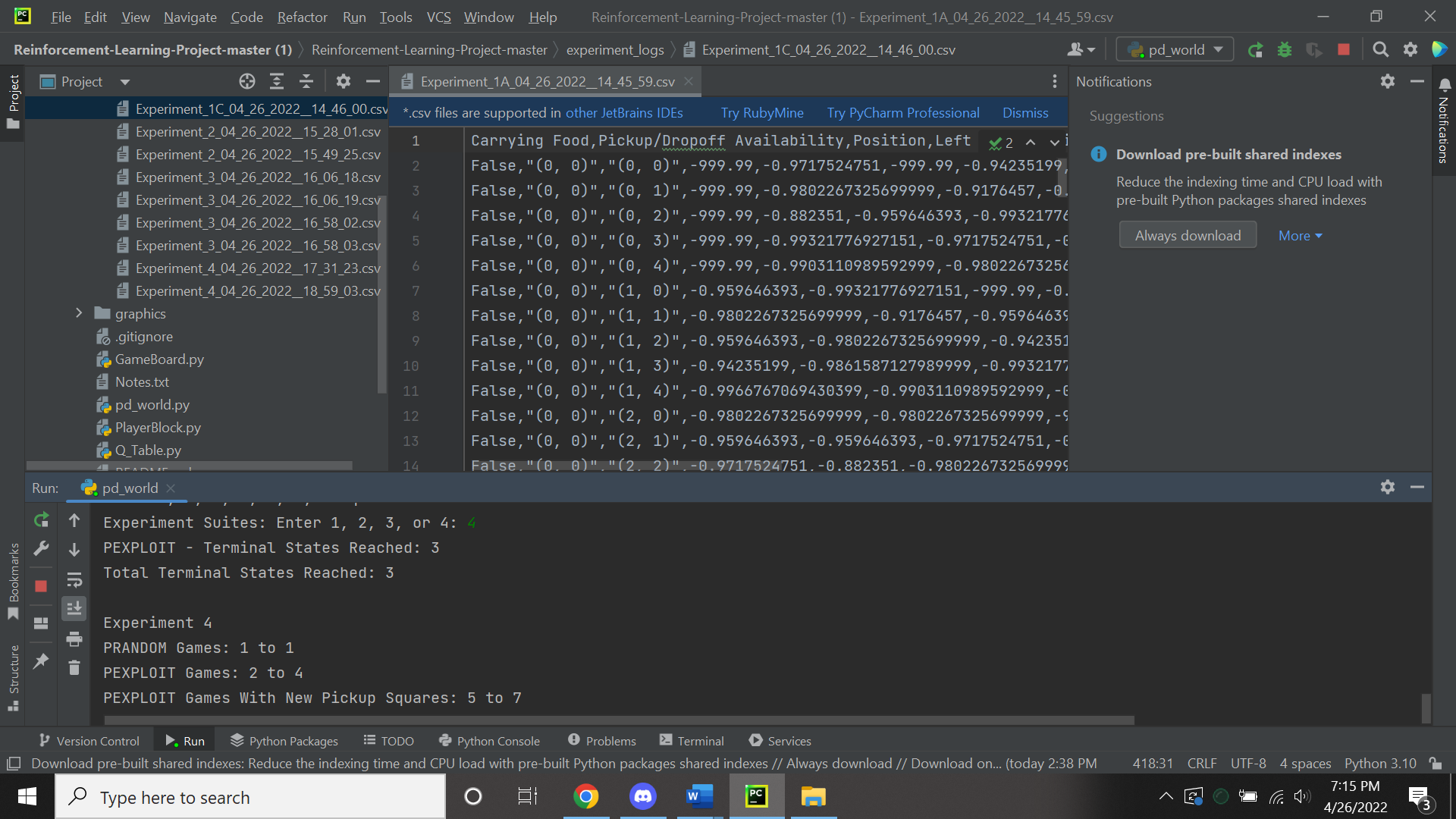
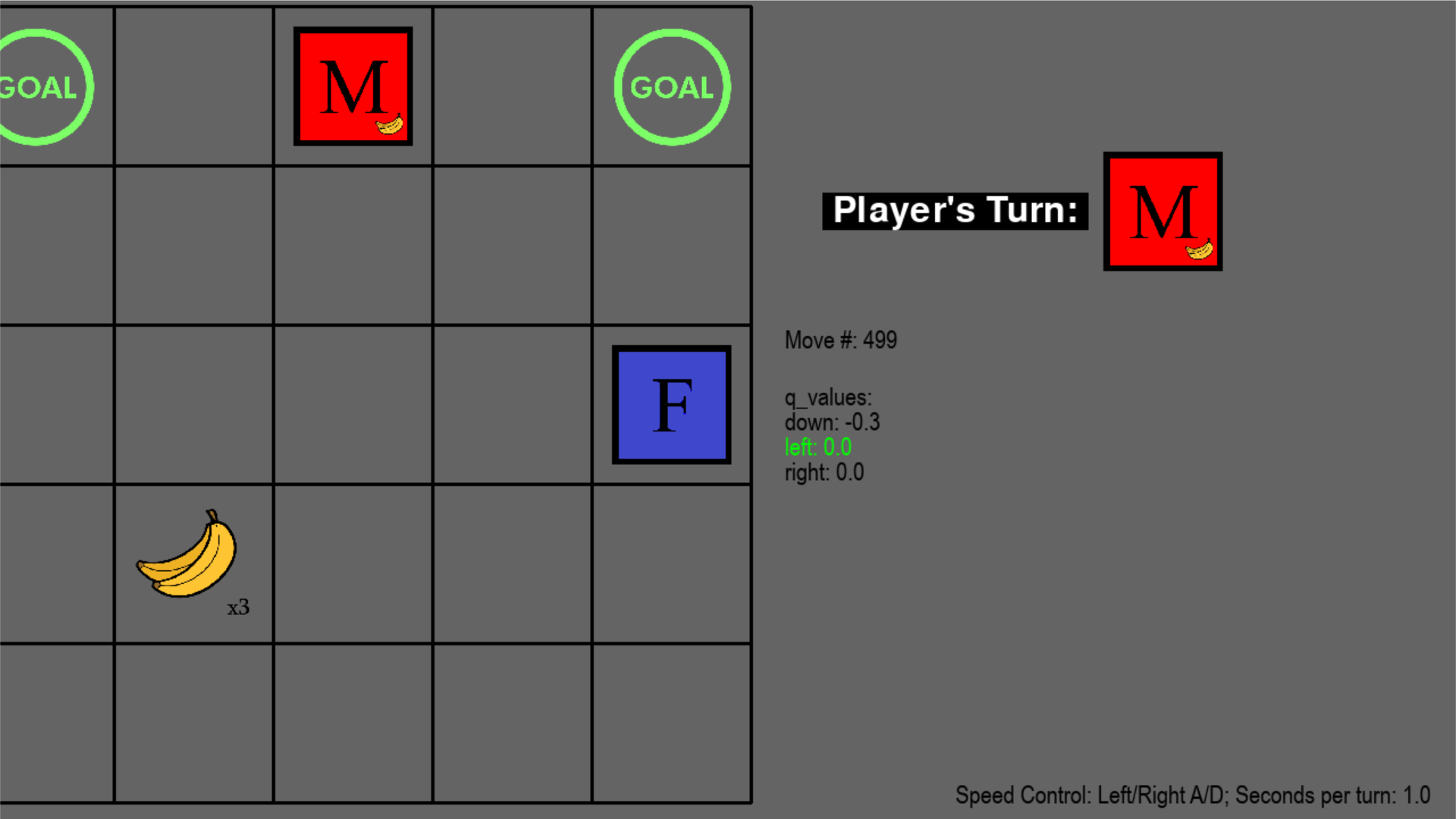
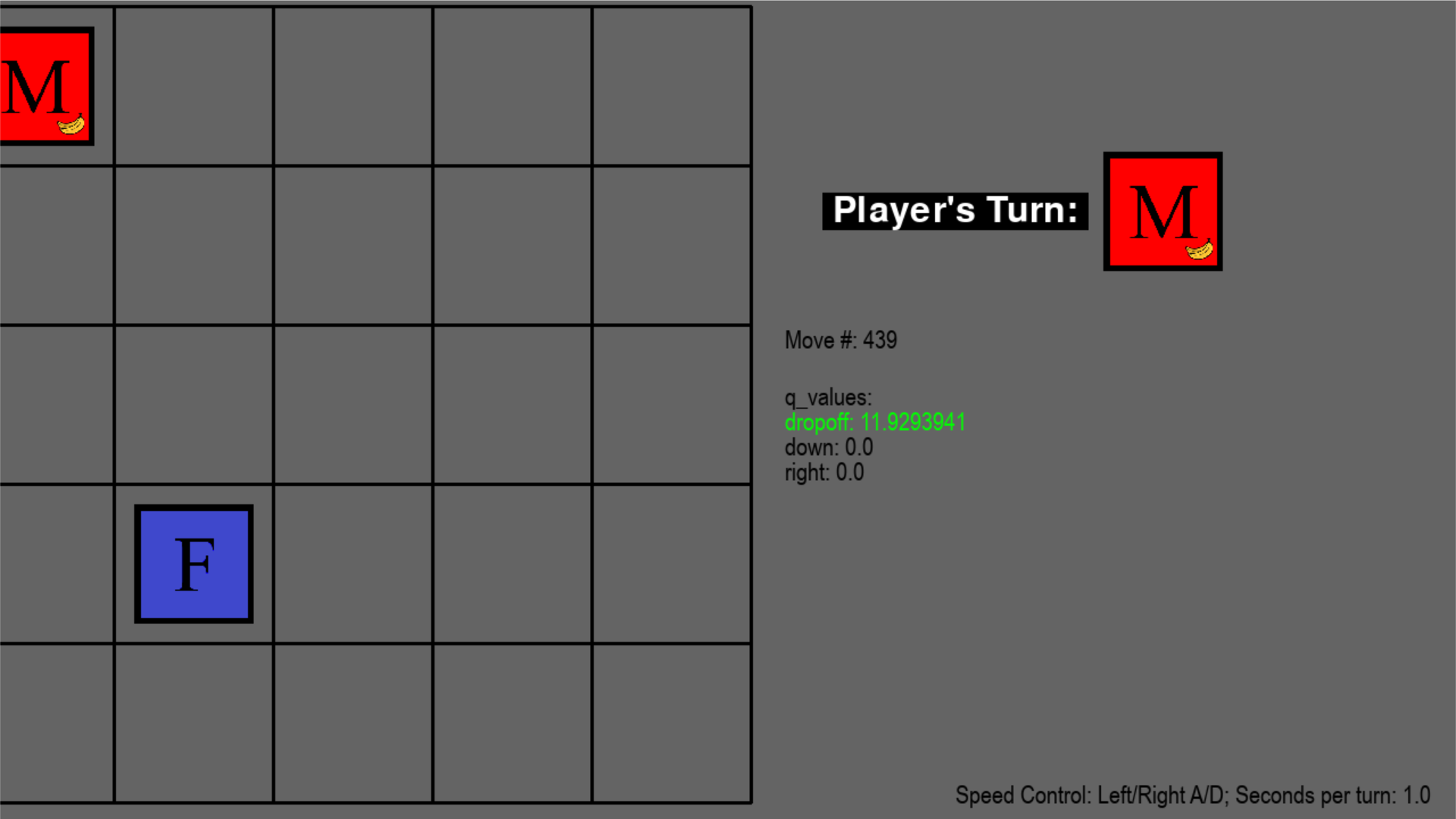
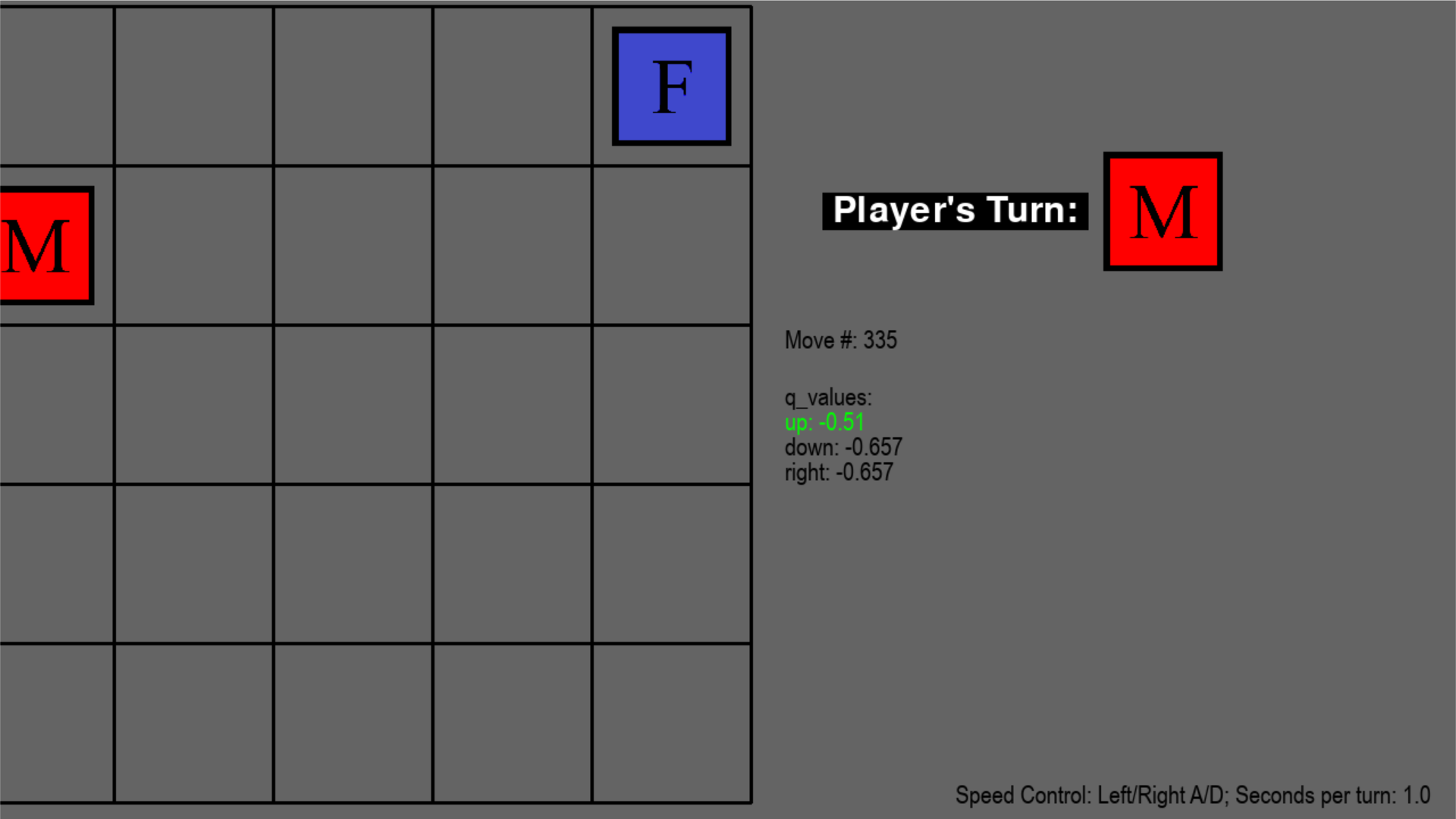
**Experiment 1 Results 2**Terminal Results: **  
PRANDOM:** Completion time took slightly longer. Again, the pathing is still not efficient and there was little to no blocking each other’s path ways. Around half the moves were spent getting the bananas in the last available goal. The total moves to complete this puzzle was 1236.   
**PGREEDY:** Slightly faster completion. Smooth transitioning around each other to take the shortest path possible. Both agents again took repetitive paths to deposit the bananas to the goals. The total moves to complete this puzzle was 164.  
  
**PEXPLOIT:** Pathing was not as efficient as greedy but still more efficient than random. Massive increase in blocking each other’s paths. Both agents did not take the shortest paths near the end in order to block each other. Pathing looked more like the pathing from the random experiment. The total moves to complete this puzzle was 284. (image below)  


**Experiment 2 Results 1**Terminal Results:  
   
**REXPLOIT:** There was efficient pathing for both agents while they deposited the banana into the same goal that was closest to them at the very beginning. The male and female agent were well coordinated and had a taking turns pattern while taking the shortest path to deliver the banana. While the number of goals lessened, the pathing became much less efficient. When 1 goal disappeared, 1 agent would efficiently take the shortest path to the next while the other agent would tend to drift away from the goal. This took much longer to complete than the version with just the q-learning algorithm. Both agents did not have the tendency to block each other but rather wait for one agent to deposit the banana and then deposit their own banana. The total moves to complete this puzzle was 472.  
**  
Experiment 2 Results 2**Terminal Results:  
  
**PEXPLOIT:** For this experiment result, the agents did not deposit in a taking turns manner for the closest goal. They worked independently during the entire puzzle compared to the first result and for the algorithm in q learning. Female and male agent did not take the shortest paths to deposit the bananas and yet the agents completed the puzzle in half the time. When there was only one goal, there were a lot of wasted moves from both agents in order to turn in the last banana. The total moves to complete this puzzle was 244. (image below)  
**  
Experiment 3 Result 1**Terminal Results:  
  
**Alpha .15 Results:** Pathing was pretty efficient overall throughout the puzzle. Both agents experienced a little bit of blocking each other’s paths. There were times where unnecessary movement At the end, there was a bit of a struggle to retrieve the last set of bananas and deposit them into the goal. The total moves to complete this puzzle was 336. (image below)

 **Alpha .45 result 1:** Both agents were efficient in pathing overall during the puzzle. There was a bit of blocking each other’s paths during the middle of the puzzle. The female and male agent again wasted moves when there was only 1 source of bananas to collect from. Pathing became less efficient as the puzzle went on. The total moves to complete this puzzle was 228.   
**  
Experiment 3 Result 2**Terminal Results:

  
**Alpha .15 Result 2:** Both agents were more efficient in their pathing in the beginning and middle of completing the puzzle. Little to no blocking each other’s paths to deposit the bananas into the goals. When there was only 1 source of bananas and 1 goal, male and female agents coordination was better than the first result. The total moves to complete this puzzle was 272.  
**  
Alpha .45 Results 2:** Very efficient pathing from both agents throughout the entire puzzle. Female and Male agents experienced more blockage from each other compared to the first result. The shortest path was taken and there were little to no wasted moves except at the very end where there was only 1 source of bananas and 1 goal available. Even then, there was minimal blocking from both agents and they were well coordinated with their movements. The total moves to complete this puzzle was 212.

**  
Experiment 4 Result 1:**Terminal results:  
  
**PRANDOM** For both agents, pathing was not well coordinated to deliver the bananas to the goals as well as the majority of the moves were wasted. Little to no obstruction with each other and they failed to collect all of the bananas before they reached the maximum amount of moves allowed. There were no notable changes in their pathing decisions. The total moves to complete this puzzle was 500.

  
**PEXPLOIT:** Over all movement is wasted throughout the entire puzzle from both agents. Female and male agent did not take the shortest paths except for in the middle of the completion. They showed very little improvement in trying to take new paths and repeated old paths instead to deposit bananas. No notable learning strategy was seen while the puzzle progressed. During the end, both agents struggled to find an efficient path to deliver the bananas to the last goal. The total moves to complete this puzzle was 492.  **  
EXPLOIT WITH NEW SQUARE LOCATIONS:** Both agents showed massive improvement in learning new paths to deposit bananas to the goals. Their pathing was efficient when the goal was nearby but when the banana sources lessened the agents trailed away trying to find new paths to arrive at the goals. Overall, the female and male agent were able to make notable changes in their pathing and act accordingly to deliver bananas.The total moves to complete this puzzle was 360.  **  
Experiment 4 Results 2**Terminal Results:  
  
**PRANDOM:**There were no notable changes compared to result 1. Both agents were terrible at pathing. No effort in learning new paths to take to deposit bananas. Female and Male agent proceeded to take old paths repeatedly. The total moves to complete this puzzle was 500.   
  
**PEXPLOIT:** Massive improvement on pathing for the Female agent. The female and male agent blocked each other multiple times throughout the beginning and middle of the puzzle. The blocking of paths proved to lessen when there was only 1 source of banana. The amount of wasted moves in order to place the last bananas dramatically increased for around half of the time to complete the puzzle. The female took the shortest paths toward the goals while the male had trouble completing the task of depositing bananas. The total moves to complete this puzzle is 440.   
**  
EXPLOIT WITH NEW SQUARE LOCATIONS:** Dramatic improvement shown for the female agent in finding the shortest path. Female agent kept being blocked by the male agent. This proved no problem because the Female agent learned new paths and made the old paths obsolete. Towards the end of completing the puzzle, both agents struggled to deposit the banana into the last goal. They both failed to find the optimal path to complete their task. The total moves to complete this puzzle was 336. ****

**EC Method A - Experiment 2/3**

In comparison to Experiment 2 without method A, the agents show a complete lack of efficient pathing in order to complete the puzzle. Both agents female and male agents’ movements were similar to the pattern of PRANDOM algorithm rather than an EXPLOIT algorithm. Not only were the turns wasted for the female and male agent, the time for completing the puzzle was double the amount compared to the first result and quadruple the amount compared to the second result. Overall, method A does not prove to surpass the first algorithm’s capabilities to solve the puzzle. In comparison to Experiment 3 without method A for alpha value at .15, there was also little to no blocking each other’s path ways. The time to complete the puzzle was doubled with the amount of turns at 528 moves in contrast to both results 1 and 2 being below 300 moves. Both agents lacked the skill of taking the shortest path. It should be noted that although method A showed to be inefficient in pathing for both experiments, only when the algorithm was left to run for over 100 thousand steps did it improve in results. For the alpha value at .45 using method A, the results were very poor in completion of the puzzle as well as the decision making to choose the best path to deliver the bananas. At the end of experiment 3 result 1 alpha value .45 that did not use method A was the pathing similar because the agents patterns were random.

**Conclusion**

The PD-World is a reinforcement learning problem that seeks to answer whether AI agents can teach themselves to navigate a finite state-space that is subject to change while the agent is delivering items from a source to a destination. We implemented a solution to this problem using Q-Learning and SARSA and evaluated the performance of our agents. In method B (both agents using the same q-table), it appears that the two agents we used for this problem performed well throughout four different experiments and training sessions with little collision between the two. In method A (each agent using a different q-table and storing the other agent’s position in the state space), the results were less favorable due to the number of states increasing so dramatically. Perhaps with 10-30x the training time, method A would’ve achieved comparable results to method B. Our discoveries found that reinforcement learning is a very powerful machine learning method and with the right hyperparameters and training, agents can reliably solve a multitude of problems on their own.

**References**

**[1]** Yang, Zhihan. “SARSA vs. Q-Learning.” *Sarsa vs. Q-Learning*, 25 July 2020,

https://zhihanyang2022.github.io/rl/sarsa\_qlearning\_comparison.