Every person may have a unique way of solving a maze problem. Some people may mentally visualize a path and solve it before starting it. Other people may just randomly take paths and learn from their mistakes. This second scenario is similar to the algorithm that an intelligent agent would use to solve the maze problem. A human would learn if something is not working and would be able to avoid that same mistake. This means that each step they make is thought through and not completely random. Though it may be random in the beginning, there will be fewer and fewer options to be chosen once one move is made.

The steps the intelligence agent would take would be like those of a human learning from mistakes. However, the more that the intelligence agent works with the maze, the more it knows the rewards and penalties. Instead of looking for just the finish of the maze, it works by looking for smaller rewards that bring it to the optimal reward at the end. In this Treasure Hunt game, the pirate is played by an intelligent agent. Though we can see every move the intelligent agent is making, we can still see some of the final decisions and how many runs it took for the intelligent agent to learn how to solve it. The algorithm allows it to know what penalties it has already encountered. With a reward system, the agent can guess which way to go next. The algorithm does have a bit of a randomizing factor added within it, which then helps to create a more human-like choice.

The agent is developed using a point system to help with the rewards and penalties. This will allow the agent to help determine what moves were the most optimal to make. In the treasure hunt game, there is a status class that is being created within it that will allow the game to know when to end if needed. If the point system that the agent receives goes below a set minimum value, then the game will end. For the agent to win, it will have to solve the puzzle and reach the part of the environment that has a proclaimed treasure.

In this particular game, there is both exploitation and exploitation. The difference is how the intelligent agent decides what move to make next. Exploitation is the may decision factor which utilizes the information the agent gained from training it had received. The secondary would be the exploration factor, randomly selecting which way to go. This factor is set to 0.1 in which to allow the agent to find the way of the treasure but also learn new information on a random basis. The 0.1 value will have the agent choose a random direction to move about every ten moves.

Reinforcement learning is used for many different things but still has a similar goal. It goes through a series of training to gain a human-like sense of learning to collect data. The purpose of this training and learning is for the agent to achieve the main goal. In the sense of this game, the agent(private)’s main goal is to navigate a path to find the treasure. Utilizing reinforcement learning, it is able to, in time, find its way to the treasure by using data it learned for which path has bigger rewards and will ultimately get it to the goal.

Q-learning's main operation is reinforcement learning, which was discussed previously in this paper. The initial state is given to the neural network for implementing it. It is then able to process any possible decisions that it is able to make and whether it will receive a reward or a penalty. All these possible decisions will be made based on a given q value within the algorithm. Using this algorithm with the q value input within the game will allow the pirate or agent to make an action that will potentially allow it to receive the greatest reward. To help choose the best action to take, all previous results from previous actions are stored for later usage. Using this data, it will seem as if the pirate knows how to navigate around the obstacles and walls within the environment.

**Citations**

Beysolow, T. (2019). *Applied Reinforcement Learning with python: With Openai Gym, tensorflow and keras*. Apress.

Gulli, A., & Pal, S. (2017). *Deep learning with keras: Implementing deep learning models and neural networks with the power of python*. Packt Publishing.

Lamba, A. (2018, September 3). *An introduction to Q-learning: Reinforcement learning*. Medium. https://medium.com/free-code-camp/an-introduction-to-q-learning-reinforcement-learning-14ac0b4493cc

Singh, S. (2022, October 7). *A comprehensive guide to neural networks in Deep Q-Learning*. A Comprehensive Guide to Neural Networks in Deep Q-learning. https://www.turing.com/kb/how-are-neural-networks-used-in-deep-q-learning