Q1. Answer the following questions:

a. While training, why do the rewards randomly dip to a low number?

Ans: Based on the observations from the generated Rewards vs Trials graph, it can clearly be observed that the rewards do indeed randomly dip to a low number at irregular intervals. Following are the reasons that independently or interplayingly causing this to happen:

- Terminal State Penalties: If the agent's action or the environment's stochastic behavior causes it to land on a bomb, this might lead to a substantial dip in the reward for that particular trial.
- Exploration due to Epsilon-Greedy Policy: The QAgent employs an epsilon-greedy policy to balance exploration and exploitation. With this strategy, the agent occasionally selects a random action rather than the one it believes to have the highest Q-value. This means that even though the agent has learned a good policy, it sometimes makes random moves that can result in negative rewards, especially a huge dip if it lands on a bomb.
- Stochastic Environment Actions: The environment has a stochastic element determined by rho. With probability rho, the action taken by the agent is randomly chosen, independent of the agent's policy. This can lead to situations where the agent inadvertently lands on a bomb or misses the gold.
- Learning Process: The learning process is not monotonic; there can be fluctuations as the agent updates its Q-values based on new experiences. Even though the overall trend is improving, some trials can end with lower rewards due to exploration or stochastic effects.

With a simple implementation of an epsilon decay strategy, stabilization during training and reduction in the frequency of drastic dips in rewards can be observed.

b. Do the arrows plotted in q_values.png make sense? What do they represent?

Ans: The arrows in the q_values.png image represent the best action to take from each state according to the Q-table learned by the Q-learning algorithm. Each arrow points in the direction of the highest Q-value for its respective state, indicating the direction the agent believes will lead to the highest cumulative future reward from that state.

Following are the elements representation:

- Blue Arrows: Each arrow shows the direction of the best action to take from that particular grid cell. For instance, an arrow pointing up means the best action from that state (grid cell) is to move up.
- Red Squares: These represent the bomb locations, where the agent would receive a large negative reward.
- Yellow Square: This is the goal state where the agent receives a large positive reward (the gold location).
- Black Background: This is the navigable space or the environment in which the agent operates.

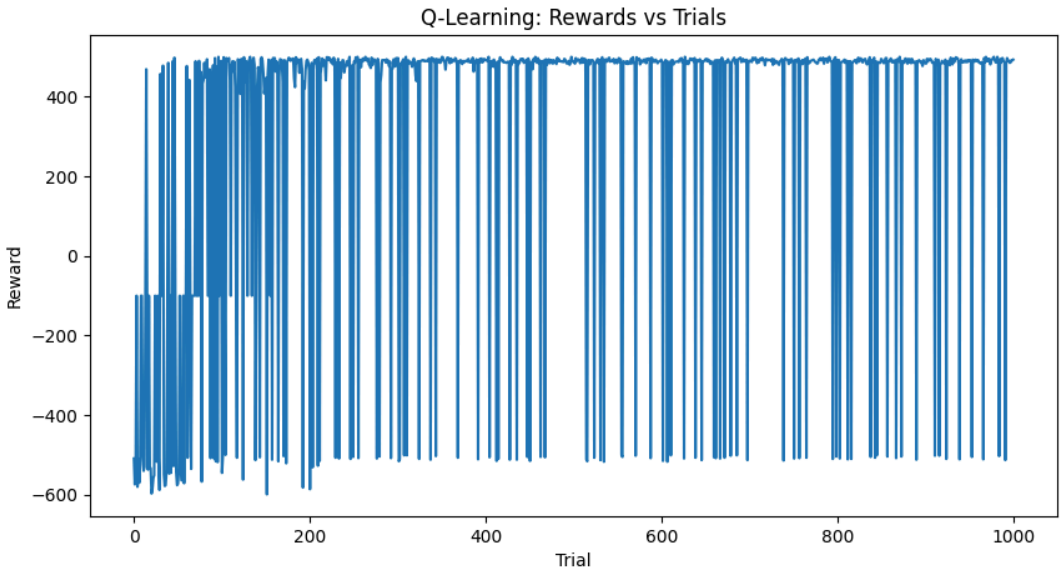
Since the arrows are mostly pointing towards the goal and avoiding bombs, it means that the Q-learning algorithm has successfully learned a good policy for this environment. If there were arrows pointing towards the bombs or away from the goal, it would indicate that the agent still has more to learn or that the epsilon value is still high enough that exploration is heavily favored. There are some suboptimal arrows due to the stochastic nature of the environment, where some states have not been visited enough to get an accurate Q-value.

c. Does the Q Agent always follow the optimal path with respect to Q values in the videos generated? Why might it not be doing so sometimes?

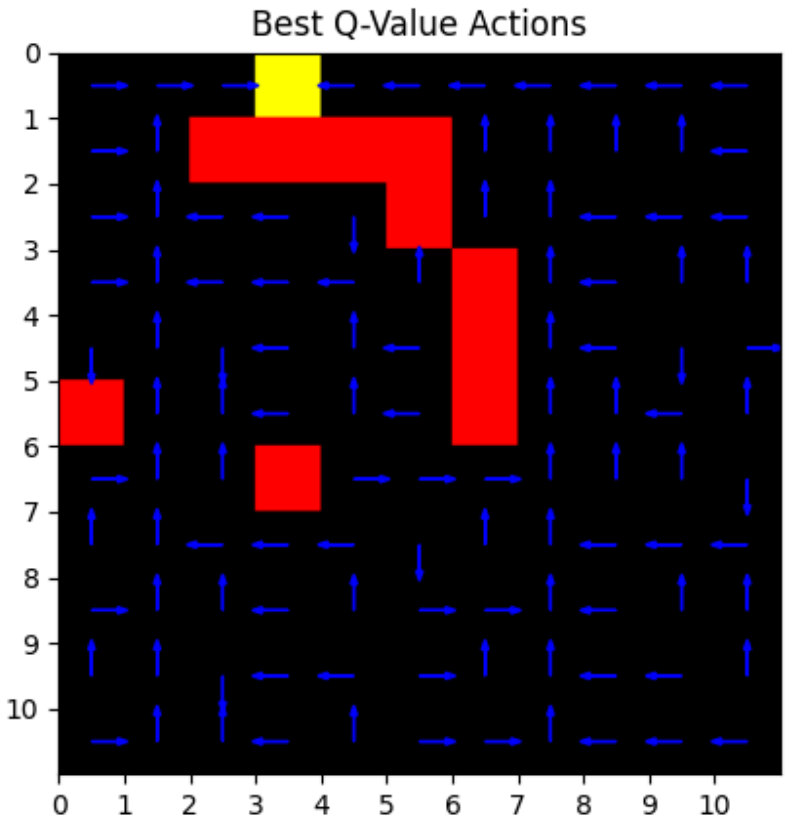
Ans: The Q Agent most of the time does follow the optimal path with respect to Q values in the videos generated, but there are instances where it ignores the learned policy and explore due to following factors:

- The agent uses an epsilon-greedy policy to balance exploration and exploitation. This means that with a probability of epsilon, the agent will choose a random action instead of the one with the highest Q value. This randomness helps the agent to explore the state space and avoid getting stuck in suboptimal policies, but it leads to seemingly random or suboptimal decisions.
- According to the GridWorld setup, the environment has a stochastic element where actions taken by the agent are substituted with a random action with probability rho. This means that even if the agent tries to take the optimal action, the environment might override this with a random one, leading to non-optimal paths.

Q2. Attach q_learning_training.png
Ans:



Q3. Attach q_values.png
Ans:



Q4. Report the Average rewards obtained by the Random Agent and Q agent in the benchmark runs (this should be printed by default)
Ans: Following is the screenshot and table of the average rewards for bother agents during benchmark runs:

```
(16662Env) gordian@shahram95:~/Desktop/16662_S24/16_662_HW4$ python q_learning.py
Benchmarking Random Agent for 500 trials
Average reward random agent: -447.732

Visualizing Random Agent
Saving 101 frames to visualizations/random_agent_0.mp4
Finished after 100 steps with total reward of -100.0
Saving 16 frames to visualizations/random_agent_1.mp4
Finished after 15 steps with total reward of -514.0
Saving 43 frames to visualizations/random_agent_2.mp4
Finished after 42 steps with total reward of -541.0

Training Q Agent for 1000 trials

Benchmarking Q Agent for 500 trials
Average reward Q agent: 485.57

Visualizing Q Agent
Saving 11 frames to visualizations/q_agent_0.mp4
Finished after 10 steps with total reward of 491.0
Saving 15 frames to visualizations/q_agent_1.mp4
Finished after 14 steps with total reward of 487.0
Saving 13 frames to visualizations/q_agent_2.mp4
Finished after 12 steps with total reward of 489.0
```

Agent Type	Average Reward
Random Agent	-447.732
Q Agent	485.57

Q5. How do the following hyperparameters empirically alter the performance of the Q Agent? Why do you think so?

a. **Q_ALPHA**
b. **Q_GAMMA**
c. **Q_EPSILON**
d. **ENV_RHO**

Ans: Following is the ablation study for varying hyperparameters (with drastic changes) to gauge the empirical effect based on the causation of these hyperparameters:

Experiment	Q_ALPHA	Q_GAMMA	Q_EPSILON	ENV_RHO	Avg. Rewards	Generated visualizations
Default	0.1	0.99	0.1	0.01	485.57	Link
Q_ALPHA Low	0.01	0.99	0.1	0.01	143.328	Link
Q_ALPHA High	1.0	0.99	0.1	0.01	484.768	Link
Q_GAMMA Low	0.1	0.1	0.1	0.01	42.232	Link
Q_GAMMA High	0.1	1.0	0.1	0.01	480.906	Link
Q_EPSILON Low	0.1	0.99	0.01	0.01	481.596	Link
Q_EPSILON High	0.1	0.99	0.5	0.01	376.118	Link
ENV_RHO Low	0.1	0.99	0.1	0.001	489.088	Link
ENV_RHO High	0.1	0.99	0.1	0.99	-437.92	Link

- **Q_ALPHA (Learning Rate)**
  - **Effect:** Determines how much new information overrides old information, hence a higher valued learning rate means that the agent will rapidly adopt the newly acquired

information, possibly overriding substantial existing knowledge. A lower learning rate means that the agent learns more slowly, integrating new knowledge more cautiously.
    - **Empirical Impact:** Setting the learning rate higher than the default, the agent converges faster for the given number of trials without experiencing adverse instability. When set too low, the learning process is slower (as observed in the rewards vs trials graph for training), and the agent does not converge to the optimal policy within the predefined number of episodes.

- **Q_GAMMA (Discount Factor)**
    - **Effect:** This dictates the importance of future rewards. A higher discount factor makes the agent value future rewards more strongly, encouraging it to think long-term. A lower discount factor makes the agent short-sighted, where it prioritizes immediate rewards.
    - **Empirical Impact:** Setting this close to 1, the agent was clearly encouraged to take actions that may have long-term benefits. And setting it closer to 0, the agent developed a myopic policy, optimizing for immediate gains and missing out on better long-term rewards.

- **Q_EPSILON (Exploration Rate)**
    - **Effect:** Determines the likelihood of the agent taking a random action, as opposed to the "best" action as dictated by the current Q-table. This balances exploration (finding new strategies) and exploitation (using known strategies).
    - **Empirical Impact:** A higher epsilon means more exploration, which prevented the convergence to an optimal policy during training as observed in the rewards vs trial curves. A lower epsilon reduces exploration in favor of exploitation, which clearly led to faster convergence during training but surprisingly didn't exhibit the increased risk of getting stuck in suboptimal policies.

- **ENV_RHO (Random Action Probability)**
    - **Effect:** Reflects the stochasticity of the environment. With a higher rho, the environment is more unpredictable as it increases the chances of a taken action being replaced with a random one.
    - **Empirical Impact:** A high rho can lead to a more robust policy that's effective under a variety of conditions, but within our scenarios it made training more challenging and increased the variance in obtained rewards. A low rho means the environment's response to actions is more deterministic, which simplified the learning but this could result in policies that don't generalize well to changes in the environment.

BONUS. Try setting up your own gridworld with:

a. **An easier map (fewer obstacles en route target)**
b. **A harder map (dense obstacles en route target)**

Keeping hyperparameters fixed, how does the performance of the Random Agent and Q Agent vary in these environments? Does the difference in performance increase or decrease with an

increase in the environment difficulty?

Ans: Following is the gridworld with my own environment:

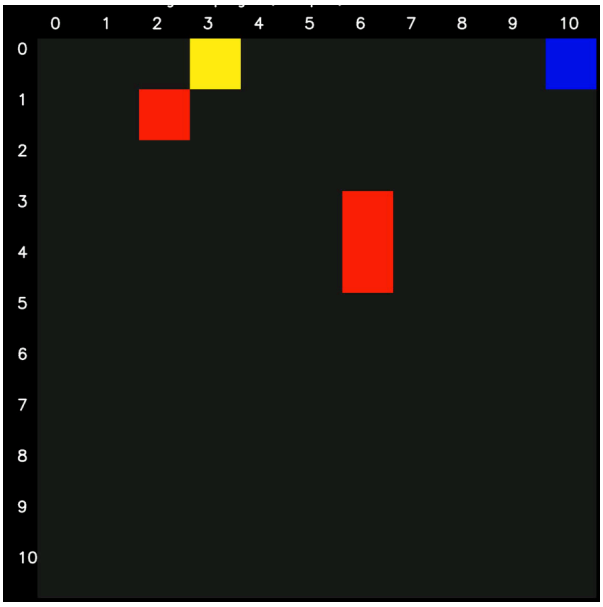


Figure (a): Easier Environment

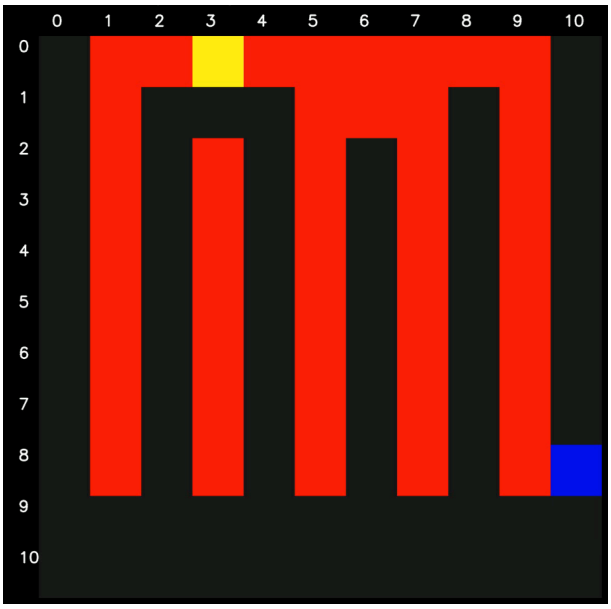


Figure (b): Harder Environment

Keeping the hyperparameters as default, i.e. Q_ALPHA = 0.1, Q_GAMMA=0.99, Q_EPSILON=0.1, and ENV_RHO=0.01 following are the observed average rewards:

Map Type	**Random Agent Average Reward**	**QAgent Average Reward**	**Generated Visualizations**
Default	-447.732	485.57	Link
Easier	-280.156	492.73	Link
Harder	-509.0	185.676	Link

Based on the observations from the table above it is evidently clear that with decreasing the difficulty, the difference between the average rewards for the Random Agent and QAgent decreased i.e. the Random Agent is less susceptible to run into a bomb location during random movement and hence increases drastically but the average reward for QAgent only improves slightly. And similarly increasing the difficulty the difference between the average rewards for the Random Agent and QAgent also decreases. A trend of decreasing performance of the Random Agent and the QAgent is observed. It is evident that for harder scenarios, the QAgent needs to be trained for a longer period in order to maintain the difference.