Assignment 2

(2) 2) 1) → Finer discretization with 50 bins per dimention creates an enormous Q-Table of $50^4 \times$ actions V.S. $50^4 \times$ actions. With only 1k episodes, the agent can not visit enough states to meaningfully update most entries in the large table. Many Q-values remain near their initial values, so the learned policy is poor. Coarse discretization with 8 bins has far fewer states, so even with 1000 episodes, each bin gets updated multiple times. This produces more stable estimates and better short-term performance. Therefore, The plot shows coarse discretization outperforming fine discretization at 1k episodes because of coverage VS. resolution trade-off.

2) → With 10k episodes, the agent has more time to explore and update the larger Q-table. Now, finer discretization starts to pay off: more bins are visited often enough to learn accurate Q-values. Coarse discretization still converges quickly but is limited by its loss of detail (different states lumped together). The plot reflects this, fine discretization improves significantly with more training, while coarse discretization plateaus.

3)→ Advantages of discretization:- Simple, easy to implement, works for small problems.

Issues in discretization:-

i) Edge cases:- Two nearly identical states can fall into different bins, causing discontinuities in policy.

ii) State aliasing:- Very different states may fall into the same bin, hiding important differences

iii) Curse of dimensionality:- Q-table size grows exponentially with bins per dimension, making training inefficient.

∴ Discretization is a hacky workaround for continuous spaces, but not ideal for complex environment.

4)→ In theory: Yes, more bins = finer resolution = closer to reality

In practice:- The Q-table becomes astronomically large, requiring massive training episodes to fill and update meaningfully. Memory and computation costs explode. Exploration becomes ~~ineffective~~ inefficient and many bins remain unused. Therefore, Even with more epochs, the computational burden makes this impractical. That's why modern RL uses function approximation instead of huge Q-tables.