1. (Policy Gradient) Let f be an arbitrary function that maps the state space $\mathcal{S}$ to $\mathbb{R}$, π_θ be the policy induced by the parameter θ, and $\rho^{\pi_\theta}(s)$ be the state occupancy measure of π_θ. Prove that

$$\mathbb{E}_{\pi_\theta}\left[\frac{\partial \log \pi_\theta(a|s)}{\partial \theta} f(s)\right] = 0\,.$$

Here the expectation is taken from the randomness of visited state s and action a generated by policy π_θ, i.e., LHS $\triangleq \sum_{s\in\mathcal{S}} \rho^{\pi_\theta}(s) \sum_{a\in\mathcal{A}} \pi_\theta(a|s) \frac{\partial \log \pi_\theta(a|s)}{\partial \theta} f(s)$.

2. (Implementation of the Dyna-Q and Dyna-Q+ algorithms) You are required to implement the Dyna-Q and Dyna-Q+ algorithms for the maze problems. In the first **simple maze** problem shown in Figure 1, there are four actions, `up`, `down`, `right`, and `left`, which take the agent **deterministically** to the corresponding neighboring states, except when movement is blocked by an obstacle or the outer boundary of the maze, in which case the agent remains where it is. Reward is **zero** on all state-action pairs, except those transiting to the goal state, on which the reward is **+1**, i.e., $r(s, a) = 1$ when the next state $s' = G$ and $r(s, a) = 0$ otherwise. After reaching the goal state (G), the agent returns to the start state (S) to begin a new episode.

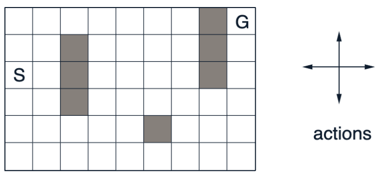

Figure 1: The simple maze problem.

In the second **blocking maze** problem shown in Figure 2, initially, there is a short path from start to goal, along the right-hand side boundary of the maze, as shown in the left subfigure of Figure 2. After **5000** time steps, the short path is "blocked", and a longer path is opened up along the left-hand side boundary of the maze, as shown in the right subfigure of Figure 2. Other basic settings are the same as the first simple maze problem.

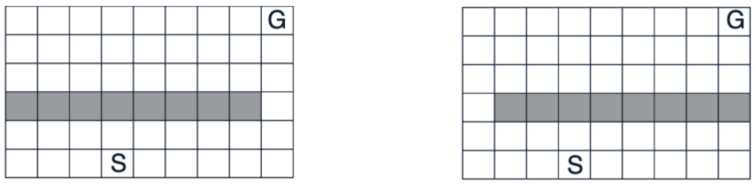

Figure 2: The blocking maze problem.

In the third **shortcut maze** problem shown in Figure 3, initially, the optimal path is to go around the left side of the boundary (left subfigure of Figure 3). After **3000** time steps, however, a shorter path is opened up along the right side, without changing the original path (right subfigure of Figure 3). And other basic settings are the same as the first problem.

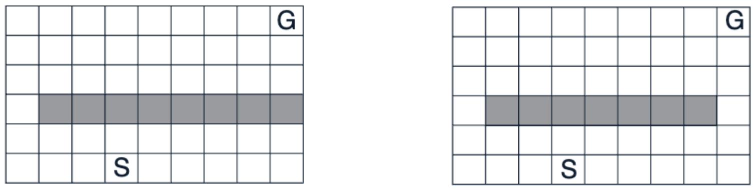

Figure 3: The shortcut maze problem.

You are required to first implement the `q_learning()` function and the `update()` function, which instantiates the Dyna-Q algorithm, in `DynaQ.py`. Similarly, implement the `q_learning()` function and the `update()` function, which instantiates the Dyna-Q+ algorithm, in `DynaQ_Plus.py`. Run your codes to show the performances of both algorithms in the above three environments. Please answer the following questions.

(a) What are the impacts of the number of planning steps on the performances of algorithms and what is the reason?

(b) What are the differences between the performance of Dyna-Q and that of Dyna-Q+ in three environments? Please discuss the reason for these differences.