Assignment 2

1. First-Visit Monte-Carlo

First-visit MC prediction, for estimating $V \approx v_\pi$

Initialize:
  $\pi \leftarrow$ policy to be evaluated
  $V \leftarrow$ an arbitrary state-value function
  $Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Repeat forever:
  Generate an episode using π
  For each state s appearing in the episode:
    $G \leftarrow$ return following the first occurrence of s
    Append G to $Returns(s)$
    $V(s) \leftarrow$ average($Returns(s)$)

An episode is generated from a random starting state. And then for each state occurred in the episode, only the first occurrence is used to update G value.

In my implementation, total time of iteration is 35000. In every episode, I set an upper limit of 100 steps to avoid episodes that's too long.

The value and policy of each state are shown below. And it can be proved that each state would take the shortest path to destination.

−8. 7→	0. 0	−15. 4←	−25. 1←	−30. 3←	−31. 7←
−17. 3 ↑	−16. 3 ↑	−22. 6 ↑	−28. 4←	−31. 5←	−31. 9 ↓
−25. 4 ↑	−26. 2 ↑	−28. 4 ↑	−30. 5 ↑	−30. 9 ↓	−30. 1 ↓
−31. 1 ↑	−31. 6 ↑	−31. 6 ↑	−30. 6→	−28. 2 ↓	−24. 9 ↓
−34. 1 ↑	−34. 3 ↑	−32. 3→	−28. 5→	−22. 5→	−15. 5 ↓
−34. 4 ↑	−34. 3→	−31. 4→	−25. 7→	−15. 7→	0. 0

2. First-Visit Monte-Carlo

- To evaluate state s
- Every time-step t that state s is visited in an episode,
- Increment counter $N(s) \leftarrow N(s) + 1$
- Increment total return $S(s) \leftarrow S(s) + G_t$
- Value is estimated by mean return $V(s) = S(s)/N(s)$
- Again, $V(s) \rightarrow v_\pi(s)$ as $N(s) \rightarrow \infty$

The case is quite similar for first-time monte-carlo. The difference is that in one episode, each state could be updated multiple times according to the order.

The value and policy of each state are shown below. And it can be proved that each state would take the shortest path to destination.

−8. 7→	0. 0	−14. 5←	−23. 0←	−27. 5←	−29. 2←
−16. 8 ↑	−15. 9 ↑	−21. 1 ↑	−25. 8←	−28. 4←	−29. 0 ↓
−24. 1 ↑	−24. 6 ↑	−26. 3 ↑	−27. 7 ↑	−28. 1 ↓	−27. 5 ↓
−29. 1 ↑	−29. 2 ↑	−29. 0 ↑	−27. 9→	−25. 8 ↓	−23. 2 ↓
−31. 8 ↑	−31. 5 ↑	−29. 8→	−26. 1→	−20. 9→	−14. 5 ↓
−32. 5 ↑	−31. 8→	−29. 0→	−23. 5→	−14. 8→	0. 0

3. Temporal-Difference TD(0)

Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated
Algorithm parameter: step size $\alpha \in (0, 1]$
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    $A \leftarrow$ action given by π for S
    Take action A, observe R, S'
    $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
    $S \leftarrow S'$
  until S is terminal

Different from MC, Temporal Different (TD) use the Bellman equation to estimate the value, and then updates the estimate as the target value. The advantage is that it can update after every step which can be seen as online learning.

The implementation is similar, first generate random episodes and update states using TD(0) mentioned above. α is set to 0.5, and γ is set to 1.

The value and policy of each state are shown below. And it can be proved that each state would take the shortest path to destination.

−3. 5→	0. 0	−10. 1←	−15. 5←	−28. 9←	−38. 8←
−11. 8↑	−20. 9↑	−30. 8↑	−36. 3←	−34. 6↑	−37. 0↓
−18. 9↑	−26. 4↑	−32. 2←	−34. 3←	−31. 2↓	−30. 6↓
−35. 0↑	−34. 4↑	−36. 4↑	−32. 9→	−25. 4↓	−16. 4↓
−39. 9↑	−37. 4↑	−34. 7→	−34. 2→	−15. 1→	−6. 3↓
−42. 1↑	−38. 6→	−35. 7→	−26. 7→	−19. 4→	0. 0