Vishal I B, 917809310
Reinforcement Learning - Homework 3

Algorithm:

Policy iteration:

The policy evaluation algorithm basically evaluates the value for a given policy matrix.
The in-place version of the iterative policy evaluation was used to implement this problem. The defined function takes in the policy as the argument. A small threshold theta was initialized to 0.00001 (below which the values did not seem to converge). The value array was set to a vector of zeros of size 1x9. The outermost loop basically implements the statement: when k tends to infinity the sequence **Vk** tends to **Vπ**. I used two nested loops inside the main loop, for the states and the actions. Based on the Bellman equation, the value for every state was calculated as the sum over the actions. The inner loop was iterated over the actions (1 to 9) and the outer loop over the states (1 to 9).for every action, the reward is basically the action taken or the bet placed. The values are updated for four different conditions of the next state based on the rewards. Based on these conditions, the value of the next state was set to 0 or the next state value. Once the values were calculated for all the states, they were stored in the value array and the delta value was calculated as max(delta, |Vk+1(s) - Vk(s)|). This is basically comparing the new values with the old ones. The outermost loop is conditioned on the value of delta, that is if the value of delta is a very small positive value, then the iteration halts. For every iteration, the value vector gets updated. A very small value of theta allows the values to converge.

Once the value converges, now we move on to the policy improvement. In policy improvement, the policy is set to greedy based on the value. The value for every action is calculated and the action with the maximum value is selected, and the probability of that action is set to one in the policy matrix. If

there are multiple actions with the same max value, then equal probabilities are set to these actions. A flag is initially set to true. If the old policy is not equal to the new policy, then the flag is set to false. Basically, if the policy hasn't converged to the optimal policy, set the flag to false. For every new policy from the policy improvement, the policy is evaluated. If the old policy is equal to the new policy, then the policy has converged to the optimal policy. If the policy has converged to the optimal policy, the value and the policy are returned.

Policy iteration - Values: 13.9268 14.5853 13.7696 12.7901 11.7924 10.7926 9.79267 8.79268 7.79268

Policy iteration - Policy:

1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	0	0

Value iteration:

The value iteration is similar to policy iteration, but the value is not allowed to converge. After the first sweep, policy improvement is done. And during the policy evaluation part, the value is not the weighted mean, but the max over all the actions. The entire function of policy evaluation and policy iteration converge together. The probability of head here is 0.4. The

probability of winning is lower than the probability of losing. Hence, the values are negative.

Value iteration - Values: -0.552 -0.88 -1.08 -1.2 -1 -1.2 -1.32 -1.12 -0.872

Value iteration - Policy:

1	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	1	0
0	0	1	0	0	0	1	0	0
0	0	0	1	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0