Vishal I B, 917809310
Reinforcement Learning - Homework 2

Algorithm:

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 1x10. The value of the 10th state (terminal state) is 0. The outermost loop basically implements the statement: when k tends to infinity the sequence **V_k** tends to **V_π**. 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.

The IB_homework2 was written to calculate the policies and call the function to return the value.

The following are the policies that were implemented using the above algorithm:

a) Aggressive policy

In this policy, the player always bets the maximum amount, that is, the entire amount the player has. So for every state, the player bets the value of the state with 0.9 probability of winning and 0.1 probability of losing. When the player wins he receives a positive reward and reaches a higher state, and receives a negative reward and reaches a lower state when he loses.

Policy: It is a 2-D matrix of probabilities. Rows represent the actions and the columns represent the states. It is a diagonal matrix of ones representing the aggressive policy, that is, the probability for action i from state j is 1 where i=j. Betting the maximum amount in every state.

States:	1	2	3	4	5	6	7	8	9	10
Values for policy one :	9.4976	9.664	6.72	8.96	4	4.8	5.6	6.4	7.2	0

b) Conservative Policy

In this policy, the player bets 1 dollar irrespective of the state he is in. So the player always has a reward of either 1 or -1 based on the probability 0.9 and 0.1 respectively.

Policy: It is the matrix of probabilities of the actions for the states. In this case, it is a matrix with the first row with ones, indicating the action one irrespective of the state.

States:	1	2	3	4	5	6	7	8	9	10
Value for policy two :	7.88889	7.87654	6.98628	5.99848	4.99983	3.99998	3	2	1	0

c) Random Policy

In this policy, the player picks a random number to bet based on a uniform distribution. So for each state, the probability of the action is a random number from a uniform distribution.

Policy: It is the matrix of probabilities of the actions for the states. In this case, it is a matrix with the lower triangle containing the probabilities. Probabilities for state j is 1/j for all the actions from 1 to i(i= j).

States :	1	2	3	4	5	6	7	8	9	10
Values :	8.65309	8.72566	8.15151	7.61073	6.67058	5.62608	5.05305	4.75018	4.61379	0