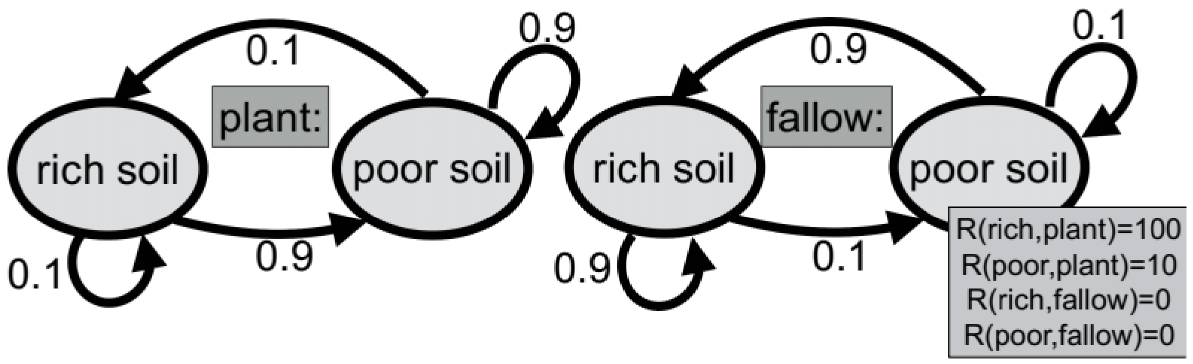
# Practical-7

**Aim: Write a python program to implement Q-value iteration algorithm for solving the Markov Decision Process (MDP) problem in the farmer example.**

- In the farmer example, the farmer has two states (rich and poor) and two actions (plant and fallow). The goal is to find the optimal policy that maximizes the expected long-term reward for the farmer under the given transition probabilities and reward values.
- Create Python Program to implement farmer example with gamma factor 0.9 value.
- Create Python Classes: FarmerEnvironment, Q
- Represent the Q-values using a separate Q class, which initializes the Q-values to zero and provides a string representation for printing the Q-values.
- Define the FarmerEnvironment class to encapsulate the environment details, including the states, actions, transition probabilities, rewards, discount factor, and Q-values.
- In the FarmerEnvironment class, implement the q_value_iteration method to perform the Q-value iteration algorithm using the Bellman equation. Implement the Q-value iteration algorithm to iteratively update the Q-values for each state-action pair until convergence.
- Implement the get_optimal_policy method to extract the optimal policy by selecting the action with the maximum Q-value for each state.
- Create an instance of the FarmerEnvironment class, perform Q-value iteration, and print the resulting Q-values and optimal policy.



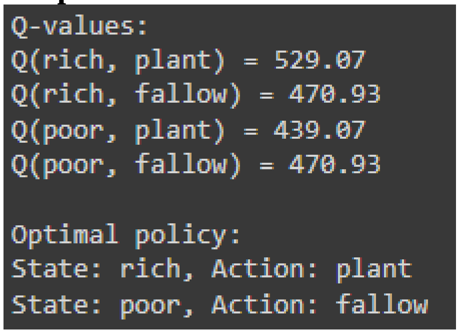
Value of action $a$ in state $s$ if make the best actions in future

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s') \max_{a'} Q^*(s',a')$$

What's the best policy?

$$\pi_{Q^*}(s) = \arg\max_a Q^*(s,a)$$

**Output:**



```
Q-values:
Q(rich, plant) = 529.07
Q(rich, fallow) = 470.93
Q(poor, plant) = 439.07
Q(poor, fallow) = 470.93

Optimal policy:
State: rich, Action: plant
State: poor, Action: fallow
```