Example: SARSA for Grid World

Consider a simple 3x3 grid world where an agent can move left, right, up, or down. The grid has a reward of -1 for each step and a reward of $+10$ for reaching the goal state. The discount factor γ is set to 0.9.

The SARSA (State-Action-Reward-State-Action) algorithm updates the Q-values based on the observed transitions (state, action, reward, next state, next action) and uses an epsilon-greedy policy for exploration and exploitation.

Let's initialize the Q-values for each state-action pair to zero:

State	Left	Right	Up	Down
S1	0	0	0	0
S2	0	0	0	0
G	0	0	0	0

Now, let's start the SARSA algorithm. We'll use an epsilon-greedy policy with $\epsilon = 0.1$ for exploration.

1. **Initialization**: Start at a random state (e.g., S1).
2. **Action Selection**: Use the epsilon-greedy policy to select an action (e.g., with probability $\epsilon = 0.1$ choose a random action, otherwise choose the action with the highest Q-value).
3. **Action Execution and Observation**: Execute the selected action and observe the reward and the next state.
4. **Q-Value Update**: Update the Q-value for the current state-action pair using the SARSA update rule:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma Q(s',a') - Q(s,a)\right)$$

   where:

   - α is the learning rate,
   - r is the observed reward,
   - γ is the discount factor,
   - s' is the next state,
   - a' is the next action.
5. **State Transition**: Move to the next state.
6. **Repeat Steps 2-5 Until Goal State is Reached or Maximum Number of Steps is Reached**.
7. **Policy Improvement**: After training, the policy can be improved by selecting the action with the highest Q-value in each state.

After training, the Q-values will converge to approximate the optimal Q-values for each state-action pair, and the agent can use these Q-values to make decisions in the grid world environment.