# Assignment 3
# CS 747

Saavi Yadav
170020003

# TASK 1

## Implement Windy Gridworld as an episodic MDP.

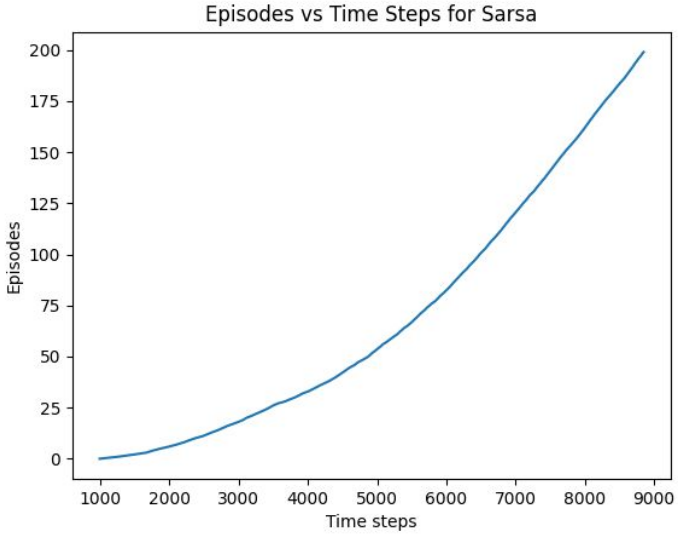
The GrindyWorld maze has been set as an mdp with rows = 7 and columns = 10.
Start = (3,0)
End = (3,7)
No of actions = 4 (Up, Down, Left, Right)
Discount factor = 1
Exploration Rate = 0.1
Alpha = 0.5
Winds = [0,0,0,1,1,1,2,2,1,0]

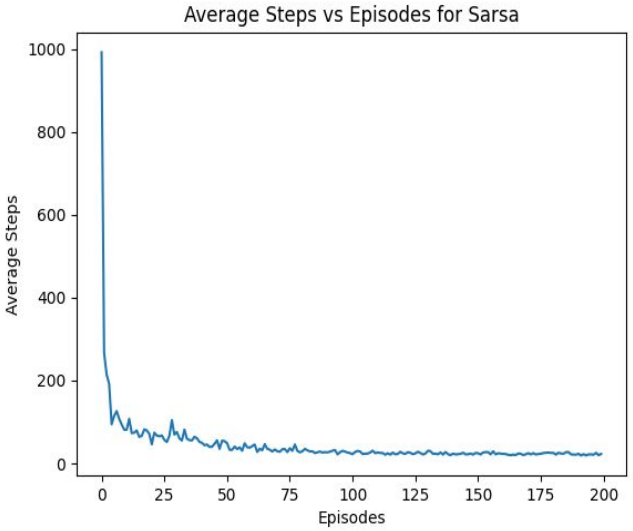
Structure of code:-
**getAction**: Gets the greedy action for the given state. With epsilon probability chooses a random action and with 1-ε probability, will choose the greedy action.
**getTarget**: Returns the target value given the method(i.e. Sarsa, expected sarsa and Q-learn).
**getIterations**: Returns the no of iterations required for a given episode.
**Sarsa**: Implements a sarsa agent with 10 random seeds and plots the average.
If an agent tries to get out of boundary, it remains in that state and attains a reward of -1.

# TASK 2

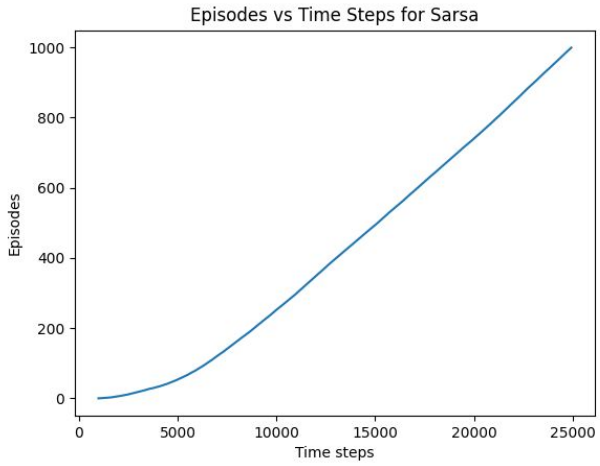
## Plots:-

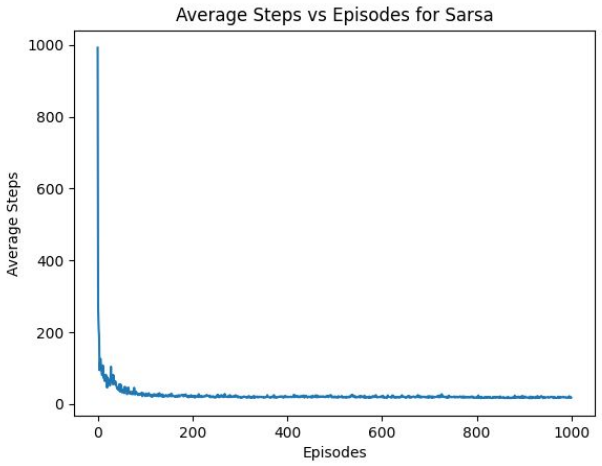
Time step = 200 episodes



Time step = 1000 episodes




## Observations:-

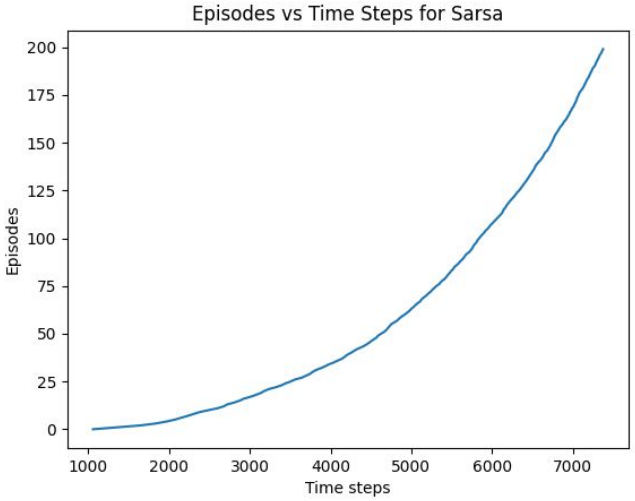
The no of time steps increase exponentially in the beginning and then linearly as episodes are increased. This indicates that the agent has learnt the best possible policy to navigate the grid.

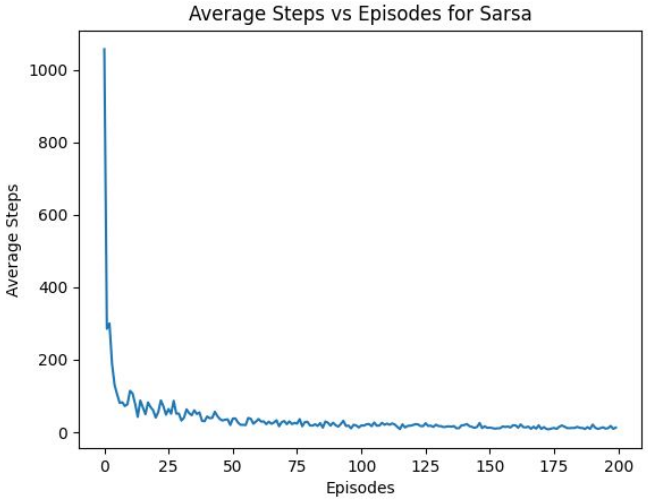
# TASK 3

For king's move the total number of actions are 8 which include the original actions as well as diagonal moves.

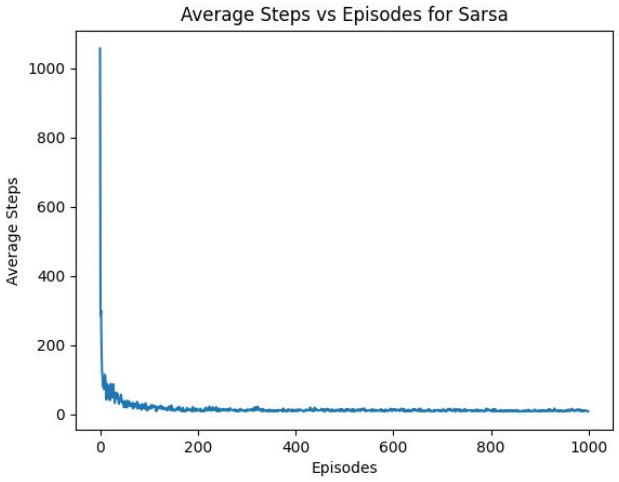
## Plots:-

Time step = 200 episodes



Time step = 1000 episodes


## Observations:-

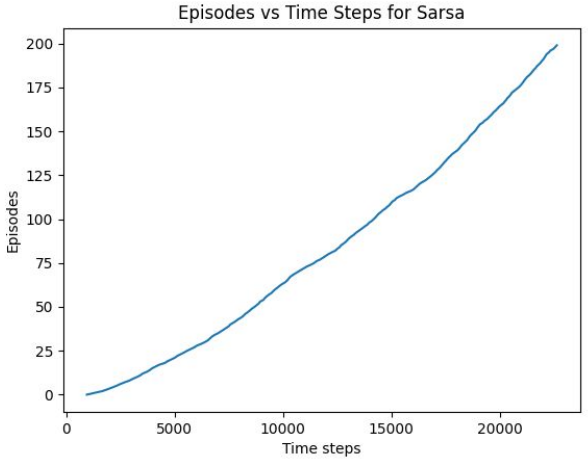
- The no of time steps increase exponentially in the beginning and then linearly as episodes are increased. This indicates that the agent has learnt the best possible policy to navigate the grid.
- The number of episodes required to achieve linear is higher for king's move as compared to without as optimal policy has to be learnt for 8 actions.
- Also, since diagonal moves are possible, the number of times steps required have decreased as compared to earlier.

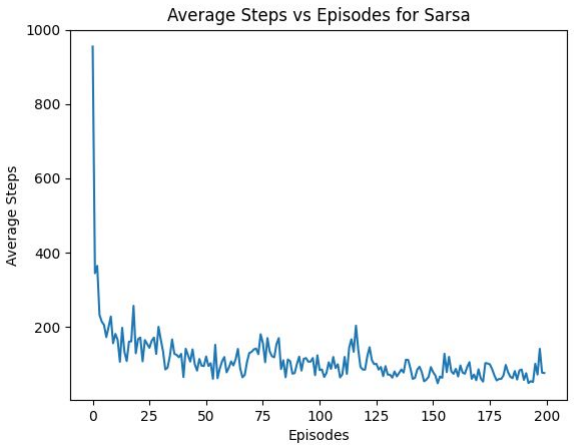
# TASK 4

For stochasticity, the wind will be [w-1,w,w+1]. Out of these, at any location one value of wind will be randomly selected with equal probability.

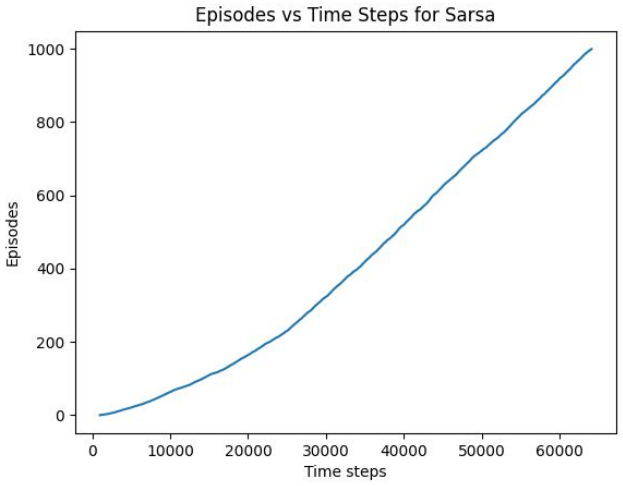
## Plots:-

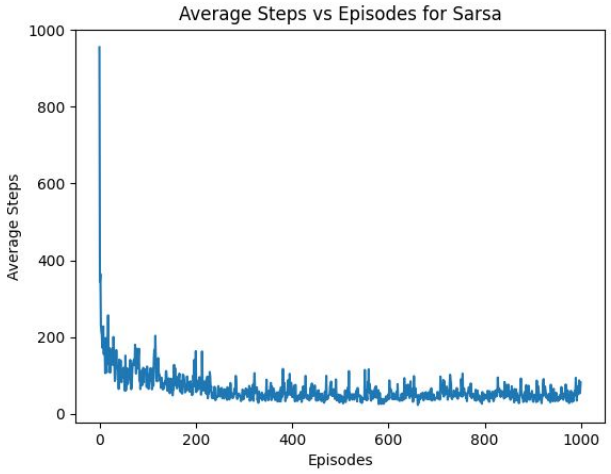
Time step = 200 episodes



Time step = 1000 episodes:-




## Observations:-

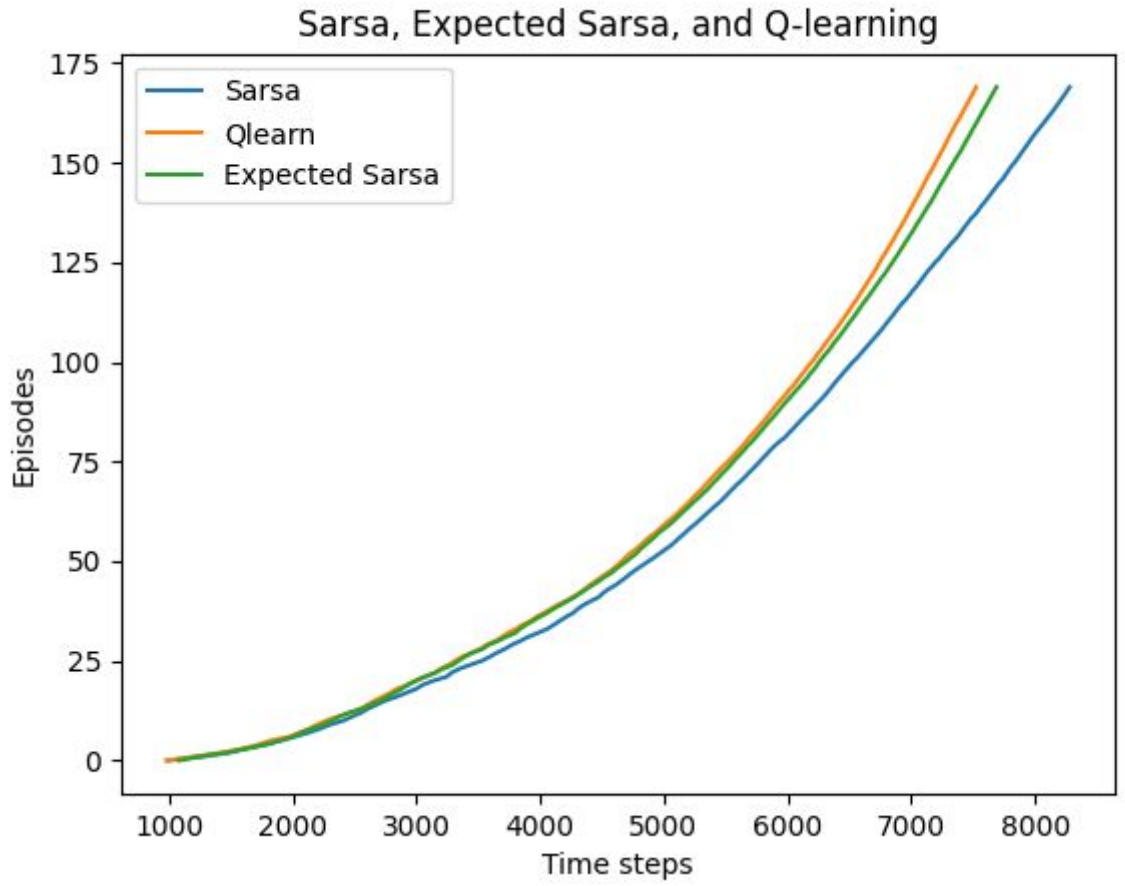
- The no of time steps increase exponentially in the beginning and then linearly as episodes are increased. This indicates that the agent has learnt the best possible policy to navigate the grid.
- The number of episodes required to achieve linear is higher for this case as compared to without as optimal policy has to be learnt for the stochastic wind.
- The number of timesteps have also increased due to the introduction of randomization, the moves predicted are not the same as what happens.

# TASK 5

Introduced method in getTarget function to accommodate other learning agents.

## Plots:-

Time step = 200 episodes




## Observations:-

- The no of time steps increase exponentially in the beginning and then linearly as episodes are increased. This indicates that the agent has learnt the best possible policy to navigate the grid.
- The number of time steps required for Q-learn is the least and sarsa is the highest among the 3.
- Q-learn selects the highest Qvalue to calculate target and hence has the highest update value leading to fewer time steps required for convergence.
- Expected sarsa has more than Q-learn as it uses ε-greedy approach to calculate target value. If ε = 0, then the time step for both Q-learn and expected sarsa would be same.