\*- coding: utf-8 -\*-

"""

Created on Sun Sep 24 21:48:07 2023

@author: Admin

"""

#SOLVING FROZEN LAKE USING VALUE ITERATION - FIND AN OPTIMAL POLICY

import gymnasium as gym

def value\_iteration(env):

num\_itns = 1000

threshold = 1e-20

gamma = 1.0

value\_table = np.zeros(env.observation\_space.n)

for i in range(num\_itns):

updated\_val\_tab = np.copy(value\_table)

for s in range(env.observation\_space.n):

Q\_values = [sum([prob \* (r + gamma \* updated\_val\_tab[s\_])

for prob, s\_, r, \_ in env.P[s][a]])

for a in range(env.action\_space.n)]

value\_table[s] = max(Q\_values)

if (np.sum(np.fabs(updated\_val\_tab - value\_table)) <= threshold) :

break

return value\_table

def extract\_policy(value\_table):

gamma = 1.0

policy = np.zeros(env.observation\_space.n)

for s in range(env.observation\_space.n):

Q\_values = [sum([prob \* ( r + gamma \* value\_table[s\_])

for prob, s\_, r, \_ in env.P[s][a]])

for a in range(env.action\_space.n)]

policy[s] = np.argmax(np.array(Q\_values))

return policy

#main prgm

import numpy as np

env = gym.make('FrozenLake-v1', render\_mode = 'human')

env.reset()

env.render()

optimal\_value\_function = value\_iteration(env)

optimal\_policy = extract\_policy(optimal\_value\_function)

print(optimal\_policy)

env.close()