```
%% MECH 6326 - Homework 4
% Author: Jonas Wagner
% Date: 2023-04-14
% Much credit to collaboraters:
% Devshan, Leon, Alyssa
% (+ Chat GPT/Bing Chatbot)

%% Problem 1
% part a
clc; clear; close all

a = load('robot_nav.mat');
[i_ind, j_ind, ~] = find(a.X); % Unpack the structure data
map = -1* a.X;
grid_size = size(map,1);

map(1,end) = 1; % End point
map(1,1:end-1) = -1;
map(2:end,end) = -1;

% Scenarios:
u_mat = [
    0.8 0.2; % North North
    0.6 0.4; % North East
    0.2 0.8; % East North
    0.4 0.6]; % East East

N = 82;

J = zeros(grid_size,grid_size,N+1); % Value function
J(:,:,N+1) = map;

pi_Star = zeros(grid_size,grid_size,N); % Optimal policy


for k=N:-1:1 % backwards time recursion
    for i=1:grid_size
        for j=1:grid_size
            if map(i,j) ~= 0            % Hit something
                if (i < grid_size) && (j < grid_size)   % Hit obstacle
                    J(i,j,k) = -1;
```

```
                    pi_Star(i,j,k) = 0;
                elseif i == grid_size           % Hit top boundary
                    Ji = zeros(2,1);
                    for u = 3:4
                        Ji(u) = map(i,j) + u_mat(u,:)*[J(max(1,i-1),j,↙
k+1) J(i,min(j+1,grid_size),k+1)]';
                    end
                    [J(i,j,k), pi_Star(i,j,k)] = max(Ji);
                elseif j == grid_size           % Hit right boundary
                    Ji = zeros(2,1);
                    for u = 1:2
                        Ji(u) = map(i,j) + u_mat(u,:)*[J(max(1,i-1),j,↙
k+1) J(i,min(j+1,grid_size),k+1)]';
                    end
                    [J(i,j,k), pi_Star(i,j,k)] = max(Ji);
                end
            else                            % Did not hit anything
                Ji = zeros(4,1);
                for u = 1:4
                    Ji(u) = map(i,j) + u_mat(u,:)*[J(max(1,i-1),j,k+1) J↙
(i,min(j+1,grid_size),k+1)]';
                end
                [J(i,j,k), pi_Star(i,j,k)] = max(Ji);
            end

        end
    end
end

figure;
imagesc(J(:,:,1));
title("Value function");
saveas(gcf,'figs/pblm1_valFunc.png')

figure;
imagesc(pi_Star(:,:,1));
title("Optimal Policy");
saveas(gcf,'figs/pblm1_optPolicy.png')

% part b
% initialize variables
```

```
N = 1000; % number of Monte Carlo simulations
success_count = 0; % count successful simulations
success_trajectories = {}; % store successful trajectories
obstacles = [i_ind j_ind]; % obstacles matrix

% Monte Carlo simulation
for i = 1:N
    % initialize a new simulation
    x = 1; % start state
    y = 1;
    trajectory = [x y];

    % simulate until reaching the goal or crashing
    while ~(x == 41 && y == 41) && ~ismember([x y], obstacles, 'rows')
        % choose an action based on the optimal policy
        action = pi_Star(42-x, y);
        if action == 1 % move right
            if rand() < 0.8 % move in the intended direction
                y = min(41, y+1);
            else % move in the unintended direction
                x = min(41, x+1);
            end
        else % move up
            if rand() < 0.6 % move in the intended direction
                x = min(41, x+1);
            else % move in the unintended direction
                y = min(41, y+1);
            end
        end

        % add the new state to the trajectory
        trajectory = [trajectory; x y];
    end

    % check if the simulation was successful
    if x == 41 && y == 41
        success_count = success_count + 1;
        success_trajectories{end+1} = trajectory;
    end
end
```

```
% estimate the success rate
success_rate = (success_count / N)*100;
fprintf("Success rate: %.2f%%", success_rate);
fprintf("\n");

% plot a successful trajectory
figure;
hold on;
for i = 1:size(success_trajectories{1}, 1)-1
    x1 = success_trajectories{1}(i, 1);
    y1 = success_trajectories{1}(i, 2);
    x2 = success_trajectories{1}(i+1, 1);
    y2 = success_trajectories{1}(i+1, 2);
    plot([y1 y2], [x1 x2],'b', 'LineWidth', 2);
end
scatter(1, 1, 50, 'ko', 'filled');
scatter(41, 41, 50, 'go', 'filled');
scatter(j_ind, i_ind, 50, 'r*');
xlim([1 41]);
ylim([1 41]);
title("Sample Successful Trajectory");
saveas(gcf,'figs/pblm1_sampleTraj.png')


%% Problem 2
clear; close all;% clc

% Problem Data
A = [.4 -.3 0 .6;
     .1 .7 .2 0;
     .5 .2 -.8 .1;
     0 .3 -.4 9];
B = [.1 .1;
     .1 .3;
     0 .1;
     .2 0];
Q = eye(4);
R = eye(2);
w = [.1; -.1; .3; -.3];
W = .1*eye(4);
N = 30;
```

```
% Calculation of Cost
P(:,:,31) = Q;
for t = N:-1:1
    P(:,:,t) = Q + A'*P(:,:,t+1)*A - A'*P(:,:,t+1)*B*inv(R + B'*P(:,:,
t+1)*B)*B'*P(:,:,t+1)*A;
end

x0 = [0;0;0;0];
constants = 0;
for i = 1:30
    constants = constants + trace(P(:,:,t+1)*W);
end

initial_cost = x0'*P(:,:,1)*x0 + constants
initial_coefficients = -inv(R + B'*P(:,:,2)*B)*(B'*P(:,:,2)*A*x0 + B'*P
(:,:,2)*w)
```