```
% do_ex_9_1_exps - performs coparison of the rate of learning with various number p
%
% Written by:
% --
% John L. Weatherwax                2007-12-03
%
% email: wax@alum.mit.edu
%
% Please send comments and especially bug reports to the
% above email address.
%
%-----

close all;
clc;

% sample_discrete.m
addpath( genpath( '../../../FullBNT-1.0.4/KPMstats/' ) );

% the learning rate:
alpha = 1e-1;

% the probability of a random action (non-greedy):
epsilon = 0.1;

% the discount factor:
gamma = 0.95;
%gamma = 1.0;

% get our maze:
MZ = mk_ex_9_1_mz(0); [sideII,sideJJ] = size(MZ);

% the beginning and terminal states (in matrix notation):
s_start = [ 3, 1 ];
s_end   = [ 1, 9 ];

MAX_N_STEPS=30;
%MAX_N_STEPS=1e3;
%MAX_N_STEPS=1e4;
MAX_N_STEPS=1e5;
MAX_N_STEPS=1e6;
%MAX_N_STEPS=10e6;

nPSV = [ 0, 5, 50 ];
fhs = zeros(length(nPSV),1); fhl = cell(length(nPSV),1); aets = cell(length(nPSV),1);
colors='rbk';
figure; hold on;
for npi=1:length(nPSV),
  nPlanningSteps = nPSV(npi);
  fprintf( 'the number of planning steps = %10d ...\n',nPlanningSteps);
  tic
  [Q,ets,numFinishes] =
dynaQ_maze(alpha,epsilon,gamma,nPlanningSteps,@mk_ex_9_1_mz,s_start,s_end,MAX_N_STEPS);
  toc
  aets{npi} = ets;

  % optionally perform some basic soothing to this signal:
  if( 0 )
    fhs(npi) = plot( 1:length(ets), ets, ['-',colors(npi)] );
```

```
  else
    filterLength = 10; % <- length of running average ... to smooth the signal
    tmp = filter( ones(filterLength,1)/filterLength, 1, ets );
    tmp = tmp(11:end); % <- drop the first ten elements ...
    fhs(npi) = plot( 1:length(tmp), tmp, ['-',colors(npi)] );
  end

  fhl{npi} = sprintf('%d planning steps',nPlanningSteps);
end
legend( fhs, fhl ); grid on;
xlabel( 'episode number' ); ylabel( 'number of timesteps until solution' );
axis( [0,500,0,70] );
fn = 'dyna_q_various_learning_rates'; saveas( gcf, fn, 'png' );

return;


% compute the (negative) cost to go and the optimal (greedy with respect to the state-
value function) policy:
pol_pi = zeros(sideII,sideJJ); V = zeros(sideII,sideJJ);
for ii=1:sideII,
  for jj=1:sideJJ,
    sti = sub2ind( [sideII,sideJJ], ii, jj );
    [V(ii,jj),pol_pi(ii,jj)] = max( Q(sti,:) );
  end
end

plot_mz_policy(pol_pi,MZ,s_start,s_end);
title( 'policy (1=>up,2=>down,3=>right,4=>left); start=green; stop=red' );
fn = sprintf('dyna_maze_policy_nE_%d',MAX_N_STEPS); saveas( gcf, fn, 'png' );

figure; imagesc( V ); colormap(flipud(jet)); colorbar;
title( 'state value function' );
fn = sprintf('dyna_maze_state_value_fn_nE_%d',MAX_N_STEPS); saveas( gcf, fn, 'png' );

return;
```