```
% function handle for logisticPop.m
f = @logisticPop;

% pass logisticPop to eulersMethod with step size 0.5, 0.1, and 0.01
popLow = eulersMethod(f, 0.5, 0, 30, 6);
popMed = eulersMethod(f, 0.1, 0, 30, 6);
popHigh = eulersMethod(f, 0.01, 0, 30, 6);

% calculate analytical solution
actual_population = arrayfun(@ivp, linspace(0,30,61));

% calcualte errors for the three step sizes used to calculate this problem
errorpopLow = absoluteError(popLow(2, :), actual_population);
errorpopMed = absoluteError(popMed(2, :), arrayfun(@ivp, ...
    linspace(0,30,301)));
errorpopHigh = absoluteError(popHigh(2, :), arrayfun(@ivp, ...
    linspace(0,30,3001)));

% plot figure that models mountain lion population
figure
plot(popLow(1, :), popLow(2, :), popMed(1, :), popMed(2, :), ':', ...
    popHigh(1, :), popHigh(2, :), '--', linspace(0,30,61), ...
    actual_population, '-.')
title('Mountain Lion Population')
xlabel('Time (years)')
ylabel('Population (Dozens)')
legend('Euler: h= 0.5', 'Euler: h= 0.1', 'Euler: h = 0.01', ...
    'Exact Solution')

%plot figure that puts the abolsute error of the numerical solutions on
%the y-axis
figure
semilogy(linspace(0,30,61), errorpopLow, linspace(0,30,301), errorpopMed, ...
    linspace(0,30,3001), errorpopHigh)
title('Absolute Error of Numerical Solutions')
xlabel('Time (years)')
ylabel('Log(Absolute Error)')
legend('Error: h= 0.5', 'Error: h= 0.1', 'Error: h = 0.01')

%function handle for the logistic harvesting equation
f2 = @logisticHarvesting;

% find the threee equilibrium solutions
z0 = fzero(f2, 0.8);
z1 = fzero(f2, 1.85);
z2 = fzero(f2, 5.44);

% pass logistic harvestignng to eulers method with four different population
% sizes
pop84 = eulersMethod(f2, 0.1, 0, 30, 84);
pop24 = eulersMethod(f2, 0.1, 0, 30, 24);
pop18 = eulersMethod(f2, 0.1, 0, 30, 18);
```

```
pop6 = eulersMethod(f2, 0.1, 0, 30, 6);

% plot figure that has the directional field, the four solutions to the
% logistic harvesting function, and the equilibirum solutions
figure
hold on
plot(pop84(1, :), pop84(2, :), pop24(1, :), pop24(2, :), ':', pop18(1, :), ...
    pop18(2, :), '--', pop6(1, :), pop6(2, :), '-.')
dirfield(f2, 0:1:30, 0:0.5:10, 'Mule Deer Population')
yline(z0)
yline(z1)
yline(z2)
xlabel('Time (years)')
ylabel('Population')
legend('Euler: h= 0.5', 'Euler: h= 0.1', 'Euler: h = 0.01', ...
    'Actual Solution')

% function handle for harvesting equation
h = @harvesting;

% simulate harvesting equation as population grows exponentially large
harvesting_simulation = arrayfun(h, linspace(1, 10000, 1000));

% plot simualtion of harvesting function and p value
figure
plot(linspace(0, 1.0, 1000), harvesting_simulation);
yline(1.2)
xlabel('Prey Population Size (Dozens)')
ylabel('Prey Harvested (Dozens)')
title('Harvesting Function as Prey Population Increases')
axis([-0.005 0.12 1.195 1.201])
```

plot equilibrium solutions for lotka volterra system in question 3.2.2

```
figure
hold on
equX = [(2.5/1.4)];
equY = [(1.5/1.1)];
scatter(equX, equY,'g', 'filled')
scatter(0,0, 'g')
flow
xline(0, 'red')
yline(0, 'black')
yline((1.5/1.1), 'red')
xline((2.5/1.4), 'black')

% fucntion handle for lotkla volterra
f3 = @lotkaVolterra;

% pass lotka volterra to ODE45, varibale is a 2 by x array with time series
% and popualtion information
[t, P] = ode45(f3, [0,30], [0.5, 1]);

% plot phase portrait
```

```
figure
flow
hold on
plot(P(:,1), P(:,2))

% plot the components in time
figure
plot(t, P(:,1), t, P(:,2));
xlabel('Time (years)')
ylabel('Population')
```

lotka volterra logistic function handle

```
f4 = @lotkavolterraLogistic;

% pass lotka volterra logisic equation to ode45 with two sets of parameters
[t1, P1] = ode45(f4, [0,30], [5,1]);
[t2, P2] = ode45(f4, [0,30], [1,5]);

% plot the phase portrait of the lotka volterra logistic system with
% horizontal and vertical nullclines on a slope field
figure
flow
hold on
xline(0, 'black')
yline(0,'red')
yline((1.5/1.1), 'black');
legend('Slope Field', 'h-nullcline', 'v-nullcline')
% anonymous function for one of the clines
f5 = @(x2) ((2.5-(2.5*0.5*x2))/1.4);
hnull = arrayfun(f5, linspace(-1, 6, 100));
plot(P1(:,1), P1(:,2), P2(:,1), P2(:,2), linspace(-1, 6, 100), hnull, 'red');


% plot components in time
figure
plot(t1, P1(:,1), t1, P1(:,2));
xlabel('Time (years)')
ylabel('Population')
legend("Population 1", "Population 2")

% plot components in time for second set of initial conditions
figure
plot(t2, P2(:,1), t2, P2(:,2))
xlabel('Time (years)')
ylabel('Population')
legend("Population 1", "Population 2")
```

Published with MATLAB® R2021b