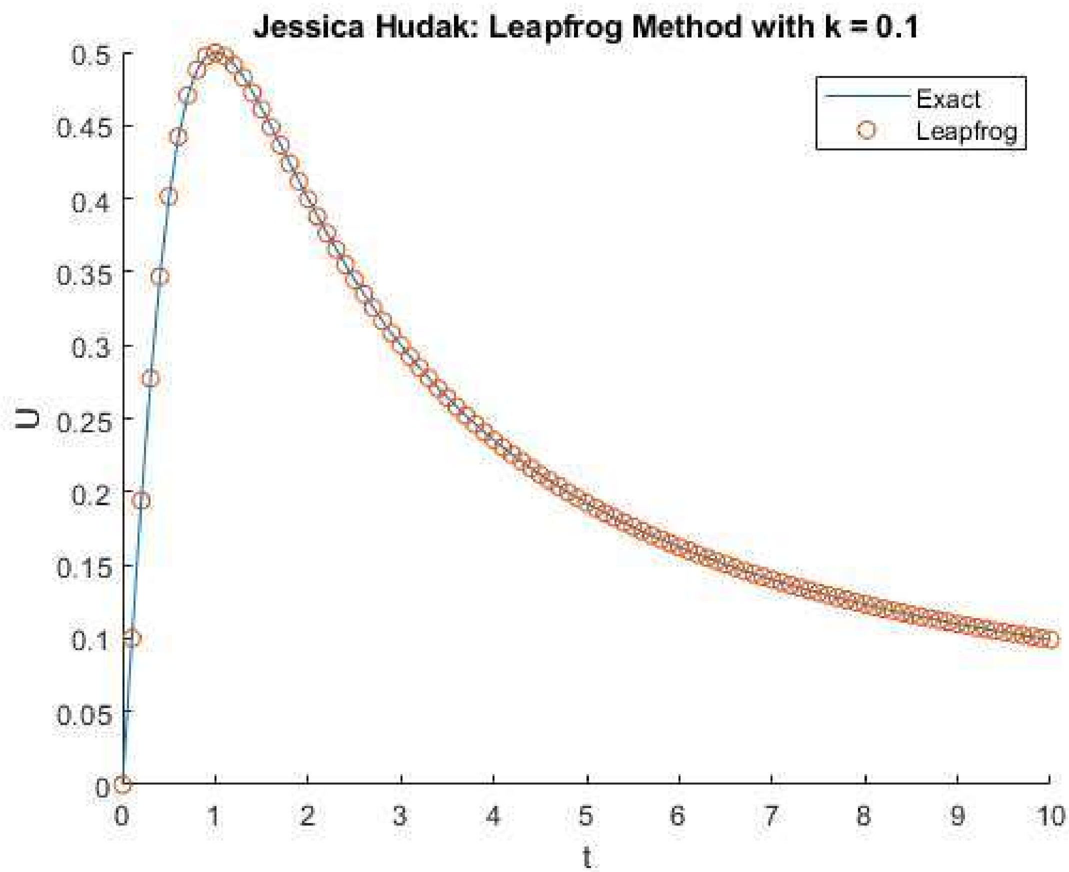
```
%Jessica Hudak
%Heat Transfer
%PROBLEM 2

%Part A: Solve IVP using the Leapfrog method

clear, clc;
% Create time array
t0 = 0;
t_end = 10;
% Initialize the time-step size
k = 0.1;
N = t_end/k;
t = linspace(t0,t_end,N+1);
% Initial value
U(1) = 0;
U_LP(1) = 0;
u_exact(1) = 0;
% Compute U_exact(n+1)
for n = 1:N
    u_exact(n+1) = t(n+1)/(1+t(n+1)^2);
end
%Use forward euler to get second term for leapfrog
U(2) = U(1)+k*(1/(1+t(1)^2)-2*(U(1)^2));
U_LP(2) = U(2);
%Back to leapfrog
U_LP(3)=U_LP(1)+2*k*(1/(1+t(2)^2)-2*(U(2)^2));
for n = 3:N
    U_LP(n+1) = U_LP(n-1)+2*k*(1/(1+t(n)^2)-2*(U_LP(n)^2));
end
%Plot results
hold all
plot(t,u_exact,t,U_LP, 'o');
legend('Exact','Leapfrog');
title(['Jessica Hudak: Leapfrog Method with k = ', num2str(k)]);
xlabel('t');
ylabel('U');
```




```
%Part B: Solve IVP using the Runge Kutta method

close all; clear; clc;
t0 = 0;
t_end = 10;
k = 0.1;  %represents "change in time"
N = t_end/k;
t = linspace(t0,t_end,N+1);
u_exact(1) = 0;

%Find the exact solution again
for n = 1:N
    u_exact(n+1) = t(n+1)/(1+t(n+1)^2);
end

%Compare exact solution to what the Runge Kutta function spits out
U = RungeKutta(t,k);
plot(t,u_exact,t,U, 'o');
legend('Exact','Runge Kutta');
title(['Jessica Hudak: Runge Kutta with k = ', num2str(k)]);
xlabel('t');
ylabel('U');

%Fuction that makes Runge Kutta work
function[U] = RungeKutta(t,k)
df = @(t,u) 1/(1+t^2) - 2*(t/(1+t^2))^2;

%Initial Condition
U(1)=0;
```

```
%Begin Runge Kutta
for i=2:length(t)

    %First, look at regular time
    %Then half step
    t_1(i) = t(i-1)+k/2;
    %Another half step
    t_2(i) = t(i-1)+k/2;
    %Finally, full step
    t_3(i) = t(i-1)+k;

   %Find the 4 slopes
    K1 = df(t(i-1)); %K1 = f(ti, yi), so depends on regular time
    K2 = df(t_1(i)); %K2 = f(ti+half step, yi due to this half step)
    K3 = df(t_2(i)); %K3 = f(ti+next half step, yi due to this half step)
    K4 = df(t_3(i)); %K4 = f(ti+full step, yi due to this full step)

    %Combine into final equation
    U(i) = U(i-1) + k/6 *(K1 + 2*K2 + 2*K3 + K4);

end
out = U;
end
```