Implement

Create the core functions for building a simulation of the system – in other words, write an appropriate "slope" function that can be used with `ode45`.

Note that once you've done a lot of work on paper, creating the simulation is pretty straightforward. Here's what I came up with as a first pass:

```
function res = skateboard_slope_func(t,W)
        % unpack
        R = W(1:2);  % unpack r and theta
        V = W(3:4);  % unpack v_r and omega

        % define parameters
        m = 70 % mass of skateboarder, kg
        MR= 70  % mass of ramp, kg
        L = 5    % total length of ramp, meters
        g = 10 % meters per second^2

        % calculate the slopes
        dRdt = V; % The simple first order DEs
        dVdt = acceleration_func(t,W); % The complex first order DE's we calculated

        % pass out the results
        res = [dRdt;dVdt];

    function res = acceleration_func(t,W)
    % Calculate second derivative of r and second derivative of theta
         r = W(1);
         theta = W(2);
         rdot = W(3);
         thdot = W(4);
         rdotdot = -g*sin(theta) + r*thdot^2;
         thdotdot = (-m*g*r*cos(theta) - 2*m*r*rdot*thdot)/(m*r^2 + 1/12*MR*L^2);
         res = [rdotdot;thdotdot];
    end

end
```

Now, there is still a LOT of work to be done here. This does not have any of the event catching you might need around the ramp hitting the ground or around the skateboarder launching off the end of the ramp. But this code at least could form a core...