```matlab
function Q1()
    % Define the ODE
    f = @(t, u) sin((u + t).^2);
    tspan = [0, 4];
    u0 = -1;

    % Reference solution using ode45 with high precision
    options = odeset('RelTol', 1e-12, 'AbsTol', 1e-12);
    [~, u_ref] = ode45(f, tspan, u0, options);
    u_ref_end = u_ref(end);

    % Values of n (number of steps)
    n_values = [2, 6, 20, 63, 200, 632, 2000];
    num_n = length(n_values);

    % Initialize error arrays
    errors_euler = zeros(num_n, 1);
    errors_rk4 = zeros(num_n, 1);

    for i = 1:num_n
        n = n_values(i);
        h = (tspan(2) - tspan(1)) / n;

        % Improved Euler Method (Heun's method)
        u_euler = u0;
        t = tspan(1);
        for j = 1:n
            k1 = f(t, u_euler);
            u_pred = u_euler + h * k1;
            k2 = f(t + h, u_pred);
            u_euler = u_euler + h * (k1 + k2) / 2;
            t = t + h;
        end
        errors_euler(i) = abs(u_euler - u_ref_end);

        % Runge-Kutta 4th Order (RK4)
        u_rk4 = u0;
        t = tspan(1);
        for j = 1:n
            k1 = f(t, u_rk4);
            k2 = f(t + h/2, u_rk4 + h/2 * k1);
            k3 = f(t + h/2, u_rk4 + h/2 * k2);
            k4 = f(t + h, u_rk4 + h * k3);
            u_rk4 = u_rk4 + h * (k1 + 2*k2 + 2*k3 + k4) / 6;
            t = t + h;
        end
        errors_rk4(i) = abs(u_rk4 - u_ref_end);
    end

    % Display the results
    fprintf('n\tImproved Euler Error\tRK4 Error\n');
```

```
    for i = 1:num_n
        fprintf('%d\t%.4e\t\t%.4e\n', n_values(i), errors_euler(i), errors_rk4(i));
    end
end
```