The danger of polynomial interpolation without care was first observed by C. Runge in 1901. He attempted to approximate the function

$$f(x) = \frac{1}{1 + 25x^2}$$

over the interval $[-1, 1]$ by polynomials that interpolate $f(x)$ at equally spaced points. Repeat his experiment by MATLAB to see what kind of danger he was referring to.

1. Using the Newton formulation, write a program in the form

   $$y_{lookup} = \mathbf{table}(x, y, x_{lookup})$$

   that works as a "table-lookup" device. That is, **table** creates internally a polynomial that interpolates points (x_i, y_i) given in the data base of vectors x and y. It then looks up the corresponding values y_{lookup} at any specified locations x_{lookup}.

2. Use your program to plot a few interpolation polynomials of degrees, say, 6, 12, and 20, respectively for the Runge's function. (Note that this is global interpolation.)

3. Report on how the error estimate formula (7.9) in our note could be used to explain the above phenomena.

4. MATLAB has a command **interp1** that does the local polynomial interpolation. (Note that the degree of the polynomial is usually low and fixed.) Compare results from using **interp1** and your **table**. *MATLAB has another command* **polyfit** *that does polynomial fitting based on the idea of least squares. However, when the degree of the polynomial and the number of data to be fitted are exactly the same, the least square fitting becomes interpolation. The problem is that* **polyfit** *construct the polynomial by using our first approach, i.e., solving a linear system, which we do not recommend. You may use this* **polyfit** *to check your answers in this project, but I still want you to write your own program* **table** *by using the Newton formulation.*