g++ B.cpp interpolation.cpp -o output\_executable   
./output\_executable

g++ C.cpp interpolation.cpp -o output\_executable   
./output\_executable

g++ D.cpp interpolation.cpp -o D\_program

./D\_program

#ifndef M\_PI

#define M\_PI 3.14159265358979323846

#endif

C:\Users\Dave\AppData\Local\Programs\Python\Python313\Scripts;%USERPROFILE%\AppData\Local\Microsoft\WindowsApps;%PyCharm Community Edition%;C:\Users\Dave\AppData\Local\Programs\Microsoft VS Code\bin;C:\Users\Dave\AppData\Roaming\TinyTeX\bin\windows;

None of the codes interfere with each other right? I'm afraid that there are multiple mains which would prevent the code from running: This my teacher's comment about my first programming assignmnet: Makefile编译无法通过（因为你的所有cpp文件里都有main函数）  
  
A. Implement the Newton formula in a subroutine that produces the value of the interpolation polynomial p\_n (f; x\_0, x\_1, …, x\_n; x) at any real x, where n \in \matbb{N^+} , x\_i’s are distinct, and f is a function assumed to be available in the form of a subroutine.

B. Run your routine on the function

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

for x \in [-5, 5] using x\_i = -5 + 10 \frac{i}{i/n}, i = 0, 1, …, n, and n = 2, 4, 6, 8. Plot the polynomials against the exact function to reproduce the plot in the notes that illustrate the Runge phenomenon.

C. Reuse your subroutine of Newton interpolation to perform Chebyshev interpolation for the function f(x) = \frac{1}{1 + 25x^2} for x \in [-1, 1] on the zeros of Chebyshev polynomials T\_n with n = 5, 10, 15, 20. Clearly the Runge function f(x) is a scaled version of function in B. Plot the interpolating polynomials against the exact function to observe that the Chebyshev interpolation is free of the wide oscillations in the previous assignment.

D. A car travelling along a straight road is clocked at a number of points. The data from the observations are given in the following table, where the time is in seconds, the displacement is in feet, and the velocity is in feet per second.

Time 0 3 5 8 13

displacement 0 225 383 623 993

velocity 75 77 80 74 72  
(a) Use a Hermite polynomial to predict the position of the car and its speed for t = 10 seconds.

(b) Use the derivative of the Hermite polynomial to determine whether the car ever exceeds the speed limit of 55 miles per hour, i.e., 81 feet per second.

E. It is suspected that the high amounts of tannin in mature oak leaves inhibit the growth of the winter moth larvae that extensively damage these trees in certain years. The following table lists the average weight of two samples of larvae at times in the first 28 days after birth. The first sample was reared on young oak leaves, whereas the second sample was reared on mature leaves from the same tree.

Day 0 6 10 13 17 20 28

Sp1 6.67 17.3 42.7 37.3 30.1 29.3 28.7

Sp2 6.67 16.1 18.9 15.0 10.6 9.44 8.89

1. Use Newton’s formula to approximate the average weight curve for each sample.
2. Predict whether the two samples of larvae will die after another 15 days.

F. The roots of the following equation constitute a closed planar curve in the shape of a heart: x^2 + (\{frac{3}{2} y - \sqrt{|x|}}^2 = 3.

Write a program to approximate the heart by cubic Bezier curves and plot your approximant, i.e. the piecewise cubic polynomials. Choose m = 10, 40, 160 in Algorithm 2.74 and produce three plots of the heart function. Your knots should include the characteristic points.  
  
Algorithm 2.74:  
Algorithm 2.74 (Curve fitting via Cubic Bézier curves). A curve \gamma: (0,1) → \mathbb{R}^D with given tangent vector \gamma^' can be approximated by cubic Bézier curves as follows.

(a) Select m + 1 consecutive marker points p\_0, ..., p\_m on \gamma and calculate tangent vector at these markers.

(b) For each j = 0,..., m - 1, calculate q\_0 := p\_j, q\_1 := \frac{1}{3} \gamma^’ (p\_j) + p\_j, q\_2 :=p\_{j+1} - \frac{1}{3} \gamma^’ (p\_j), q\_3 := p\_{j+1} and determine a cubic Bézier curve (2.59) from q\_j ’s.

(c) Connect the m cubic Bézier curves.

    
 