Polynomial Interpolation Programming Assignment
Numerical Analysis

Goal: Create a project which, given a set of points, creates a polynomial function which interpolates those points. Steps to follow - in this order:

1. Download the Polynomial Interpolation App template and open the programming project:
   - Navigate to https://github.com/pattiscot/Numerical-Analysis-Student-Ready.git
   - Click the green "code" button and select "Download file"
   - Unzip the folder and open the PolynomialInterpolationApp. Simply double clicking the .zip file to open it will NOT suffice. It will not work properly in Visual Studio if you do this.
   - Double click the PolynomialInterpolationApp.sln file. This should open the project in Visual Studio. Make sure to use version dotnet 8.0 for all your programs this semester.
2. Identify and inspect the interface.
   - You will find the Solution Explorer on the right hand side bar of Visual Studio. Use the Solution Explorer to find the interface which you need to implement. Recall that all interfaces are named with a leading capital I. So the interface in this project is a file named IPolyInter.cs. You should NOT make any changes to any of the interface classes (or other classes except the Programs.cs class and the class you create to implement the interface). However, you can inspect the interface to see what you are supposed to do in your implementation. You can open the interface (or any file in the Solution Explorer) by double clicking the name of the file in the Solution Explorer.
3. Add the implementation class to the project
   - Right click the name of the project (the bold name in the Solution Explorer) and select ADD and then in the sub-menu, select Class...
   - Select the class option and then, in the bottom area of this window pane, name the class PolyInter.cs
   - Click Add
4. Implement the methods specified in the interface
   (a) If the implementation class you just created does not automatically open then double click its name in the Solution Explorer to open it.
   (b) Replace the word internal with the word public. This changes the class to a public class which has greater accessibility within the project.
   (c) Setup the inheritance from IPolyInter.cs by adding :IPolyInter to the end of the line "public class PolyInter" like you learned to do in one of the tutorials.
   (d) Click the little light bulb - usually on the left hand side of the screen - and tell it to implement the interface. This should copy all the methods that I specified in the interface to your implementation and give you space to write your code.
5. Determine what you need to program by reading the interface.
   - Note that you should not make any changes to the getters and setters which are given at the top of your program.
   - Use double[] A to store the coefficients which your program calculates
   - Use double[] X to store the x values provided by the user
   - Use double[] Y to store the y values provided by the user. Note that order matters since the x values and the y values come in (x,y) pairs as points.
   - Use string Polynomial to store the string representation of your polynomial in Newton's Form (specified in the textbook). Round all the coefficients to 5 decimal places when writing the string. Do not round the coefficients in any other place though! **If you cannot calculate the coefficients for a polynomial, then your polynomial should be the string "No Solution".**

- Set bool IsPossible to true if you are able to find a polynomial interpolation for the set of points. Set IsPossible to false if you were unable to find a polynomial interpolation for this set of points.
- void Coef() returns nothing and accepts nothing in the call. It calculates the coefficients using the improved divided difference method and saves them to double[] A.
- double Eval(double t) returns a value of type double and accepts as input a value t of type double. This method evaluates the polynomial at some value t. Return double.NaN if you were unable to create the interpolating polynomial.
- You, in future uses of this program, will need to be able to write a string representation of the polynomial to the console. This is not implemented in the pseudocode so you will need to figure out how to do this.

6. You can use the Program.cs file for all your I/O. This can be used to help debug your program.
7. Submit your program
    - Create a zip file of your PolyInter.cs file.
    - Using the link to OICLearning posted on eclass, navigate to the Polynomial Interpolation App assignment.
    - Submit this zip file on OICLearning. Wait a little while (up to a few minutes) to see your grade. You can resubmit as many times as you want until the deadline is passed. I will record the score from your last submission in the gradebook on eclass - so make sure that this is your highest score.