MATLAB Programming NYCU Fall 2022 Assignment#3 due 2022/12/6

In this assignment, your task is to implement a class called **`SparsePoly`**. Some of the functionalities are based on those in Assignment #2, and you can reuse some of your code.

The properties of the class consist of a vector **`degree`** and a vector **`coefficient`**. The two vectors should be of the same length, will the corresponding elements being the degree and coefficient of a term, respectively. The rules are similar to those in Assignment #2:

- The terms are in descending order of degrees.
- No two terms can have the same degree.
- Only terms with nonzero coefficients are included. The only exception is when the polynomial is zero, in which case both **`degree`** and **`coefficient`** contains scalar zero.

Apply set access control to the properties as (**`SetAccess=private`**). This is to ensure the consistency of the values in **`degree`** and **`coefficient`**. The only way to set the properties is through the constructor.

Class methods to implement:

- Constructor (only implemented to return a scalar object):
  - No input argument: The default constructor. A zero polynomial is returned.
  - Two input arguments: Both should be vectors of the same length, with one being the degrees and the other being the coefficients. Do proper input argument checking here.
- Operator overloading functions **`plus`**, **`minus`**, and **`times`**, for addition, subtraction, and (elementwise) multiplication, respectively. These should take two **`SparsePoly`** arrays of the same size, or one array and one scalar, as inputs.
- Function **`eval`**: It takes a **`SparsePoly`** array and an array of x values at which the polynomials are evaluated. The output is a cell array of the same size as the input **`SparsePoly`** array. Each element of the cell array is a **`double`** array of the same size of the array of x values, containing the evaluated values for the corresponding **`SparsePoly`** object.
- Function **`plot`**: It takes a **`SparsePoly`** array and a vector of x values at which the polynomials are evaluated and plotted. Call function **`eval`** for the purpose of evaluation. For plotting, simply used line plots. Plot the curves of all the polynomials in a single axes using **`hold on`**, and let MATLAB select the colors automatically.
- Function **`disp`**: For text display.
  - When the input argument is a scalar: List the degrees and coefficients in the form like
    ```
    x^3+5x^2-2
    ```
  - When the input argument is a non-scalar array: Just print out the class name and array size like:
    ```
    SparsePoly array (size 3x2)
    ```
- (Optional) An extra method just for making a polynomial consistent with the rules. This method should have private access, so it should be placed in a separate methods block like
  ```
  methods (access=private)
      …
  end
  ```

Submission: Submit your code (m file) through E3. As this is a class file, the file name is the same as the class name. There will be a three-day grace period after the due date, during which there will be a 10%/day deduction for your grade.

A "copy detection" will be applied to your submissions, and those found to have copied assignments will receive zero points for the assignment.

Your code should include sufficient comments. This will be part of the grade. Include your name and ID at the top of your code.

There will be demo session with the TAs (date/time to be announced later).