

# *NSYSU-MATH Data Structure – Spring 2024*

## Homework 1

### Design: Designing a Polynomial Class

#### Data Preparation

---

For this assignment, you will find a zip file named `HW1.zip` containing template files and public test data. Your task is to implement a `Polynomial` class in either Python or C++. The directory structure and contents are as follows:

1. Python Implementation (Py/ directory):
  - ✓ `Polynomial.py`: Implement your `Polynomial` class here.
  - ✓ `test.py`: Contains public test cases for your implementation.
  - ✓ `benchmark.py`: A template for conducting benchmark analysis.
2. C++ Implementation (Cpp/ directory):
  - ✓ `Polynomial.cpp`: Implement your `Polynomial` class here.
  - ✓ `Polynomial.h`: The header file for your `Polynomial` class.
  - ✓ `main.cpp`: Contains public test cases for your implementation.
  - ✓ `benchmark.cpp`: A template for conducting benchmark analysis.

#### Description

---

This assignment is divided into three main parts:

1. Environment Setup:
  - ✓ Choose either C++ or Python as your programming language.
  - ✓ Set up your programming environment accordingly.
2. Class Implementation:
  - ✓ Implement a new class named `Polynomial` in the provided template file. For Python, use `Polynomial.py`. For C++, use `Polynomial.cpp`.
  - ✓ The specifications for the `Polynomial` class will be provided in the subsequent sections.
3. Time Complexity Analysis:
  - ✓ Analyze the time complexity for the following operations in your `Polynomial` class: **Addition, Subtraction and Multiplication**. Report the worst-case time complexity using Big O notation.
  - ✓ Use the benchmarking method introduced in class to validate your analysis. Implement your analysis in the provided template (`benchmark.py` for Python or `benchmark.cpp` for C++).

Note: You may assume that all basic operations on **lists** (or **vectors** in C++) have constant time complexity for the purpose of this analysis.

## ADT

### Polynomial ADT

**Data:** A list (vector) that stores coefficients stores in **descending order** from left to right. An integer that records the degree of polynomial

### Operation:

1. **Initialize:** Creates a new polynomial that is constructed using the given coefficients. It needs a list of coefficients and returns the polynomial.
2. **Addition:** Add two polynomials and return the resulting polynomial:  $(x^2 + 3x + 2) + (x + 2) = x^2 + 4x + 4$
3. **Subtraction:** Subtract one polynomial from the other and return the resulting polynomial:  $(x^2 + 3x + 2) - (x + 2) = x^2 + 2x$
4. **Multiplication:** Multiply two polynomials and return the resulting polynomial:  $(x^2 + 3x + 2) \times (x + 2) = 2x^3 + 5x^2 + 8x + 4$

## Specifications

1. Class name: `Polynomial`
2. Attribute name: `_degree`, `_coeff` (They should be private)
3. Method: Constructor (list of coefficients), `+`, `-`, `*` and negation. You should implement them using operator overloading. Note a custom `print()` method for the class is already implemented. Do not modify this method.
4. Use a **list** (in Python) or a **vector** (in C++) to store the coefficients.
5. Coefficients should be stored in descending order of power (from left to right). For a polynomial with highest power  $x^n$  it will contain  $n + 1$  terms (Input sequences may contain leading zeros; these should be removed).  
Ex:  $3x^4 + 2x^3 + x^2 + x$  (Input will be `[3,2,1,1,0]` or `[0,3,2,1,1,0]` ...)

0	1	2	3	4
3	2	1	0	0

$$-2x^4 + x^2 + 0.5$$

0	1	2	3	4
-2	0	1	0	0.5

$$x + 1$$

0	1
1	1

6. Please combine the terms that have the same powers.
7. The input coefficients can be integers or floating-point numbers.
8. You can only use standard [Python](#) or [C++](#) library and do not use `reverse()` or `[::-1]` method for `list` and `vector`.

## Deliverables

---

1. Deadline: 2024/3/17 (Sun.), 11:59 PM. Hand in the following two items to the cyber universities. Please see our [Facebook group](#) for the late policy and rules.
2. Report:
  - ✓ Describe your programming environment and provide instructions on how to set it up.
  - ✓ Explain the design of your program and the data structures used. Discuss what you have learned from completing this homework.
  - ✓ Provide a detailed analysis of the time complexity (Big O notation) and benchmark results for the Addition, Subtraction, and Multiplication operations in your implementation.
3. Program Source Files:
  - ✓ Submit your source files in a zip file. **Ensure that you follow the provided template files.**
  - ✓ Source File Comments: Each file must begin with three lines of comments indicating the Author, Date, and Purpose of the program. Include appropriate comments throughout your code for clarity.

## Grading Policy

---

- Function Correctness: 60% (45% for public test cases and 15% for hidden test cases).
- Big O and Benchmark Analysis: 20%.
- Report: 20%.

## Reference

---

1. <https://python-course.eu/oop/polynomial-class.php>
2. [https://hplgit.github.io/primer.html/doc/pub/class/. \\_class-readable003.html](https://hplgit.github.io/primer.html/doc/pub/class/. _class-readable003.html)
3. <https://web.ntnu.edu.tw/~algo/Polynomial.html>
4. <https://gist.github.com/birshert/8965693055464cb8b4e4cb16d6306fc8>