The University of Texas
Computer Science

313E Programming Assignment 10
Polynomials

Setup

This assignment may be completed individually or with a pair programming partner. Be sure to include your and your partner's name and EID in the Python file header. If you are working alone, you may delete one of them.

Complete these steps to prepare for the Assignment.

Check	Description
▢	Download poly.py and poly.in. You will be working on the poly.py file. You may not change the file names. Otherwise, the grading script will not work.
▢	Place both files in the same folder/directory.

If you would like to turn on autosave in VSCode, click on **File -> Autosave.**

Problem Description

A polynomial is an algebraic expression defined as the sum of terms, each made up of a coefficient and a variable (indeterminate) raised to the power of a positive number. An example of a polynomial with two variables is $x^2 + 2y - 10$. In this example, there are three terms: x^2, $2y$, and 10.

For this assignment, you can assume that the polynomials that we will be representing will have integer coefficients and exponents. The coefficients can be positive or negative. The zero-coefficient terms are not shown in the polynomial expression. The exponents are zero or greater.

Requirements

For this programming assignment, you will represent a polynomial as a linked list. We have redefined the **Node** class (which defines the nodes in the linked list). Based on this, you will be working on the **LinkedList** class. You will implement two main arithmetical functions on polynomials: addition and multiplication. You may add as many helper functions as needed.

There are a few restrictions you must follow:

1. Your program uses the provided text file poly.in to create your final output. You will be using input redirection to read from this file.
2. **You may not change the names or parameters of the functions listed. They must have the functionality as given in the specifications. You can always add more functions than those listed.**

3. **You must use Linked Lists to solve this programming assignment.**
4. **You may not import any additional external libraries in your solution.**

You will modify the following functions:

def insert_term(self, coeff, exp)
Insert the term with the coefficient ***coeff*** and exponent ***exp*** into the polynomial.
If a term with that exponent already exists, add the coefficients together.
You must keep the terms in descending order by exponent.

def add(self, p)
Add a polynomial ***p*** to the polynomial and return the resulting polynomial as a new linked list.

def mult(self, p)
Multiply a polynomial ***p*** with the polynomial and return the product as a new linked list.

def __str__(self)
Return a String representation of the polynomial. The expected format is defined in the **Output** section.

def main()
In this method, you will read data from the file poly.in from standard input (stdin). Using the data, you will create polynomial p and polynomial q. You will get the sum of p and q and print the sum. Lastly, you will also get the product of p and q and print the product.

Input
Your program uses the provided text file **poly.in** as input. You will write the code to read from this file in **main()**. **poly.in** will contain data for just two polynomials separated by a single blank line.

```
3
3 4
-2 3
5 2

3
5 3
-6 1
2 0
```

For p: The first line of data will be **n**, the number of terms in the **first** polynomial. It will be followed by **n** lines of data. Each line will have two integers separated by whitespace. The first integer will be the coefficient and the second integer will be the exponent.

There will be a new line to separate p and q.

For q: The new line will be followed by a single integer **m** representing the number of terms in the **second** polynomial. It will be followed by **m** lines. Each line will have two integers separated by whitespace. The first integer will be the coefficient and the second integer will be the exponent.

The code can assume that the file is structured properly.

Output
Your output will be the sum and product of the two polynomials in the following format:

```
(3, 4) + (3, 3) + (5, 2) + (-6, 1) + (2, 0)
(15, 7) + (-10, 6) + (7, 5) + (18, 4) + (-34, 3) + (10, 2)
```

Each term in the resulting polynomial will be represented as a String of the format:
`(coefficient, exponent)`

The terms in the polynomial should be combined using: **+**

The first line will be the sum, and the second line will be the product. Both the sum and the product will be in descending order of the exponents. There should not be any zero **coefficient** term in the output.

Running Your Program
You can run the program by typing the following command into your terminal:

```
python3 poly.py < poly.in
```

Alternatively, if you are using a Windows computer and are having trouble with input redirection <, you can run:

```
cat poly.in | python3 poly.py
```

Grading

Visible Test Cases - 75/100 points
The program output exactly matches the expected output for each function. To receive full credit, the program must produce the correct output based on all requirements in this document and pass all the test cases.

Hidden Test Cases - 15/100 points
The program output exactly matches the expected output for each function. To receive full credit, the program must produce the correct output based on all requirements in this document and pass all the test cases.

Style Grading - 10/100 points
The poly.py file must comply with the PEP 8 Style Guide.

If you are confused about how to comply with the style guide, paste the error or the error ID (e.g. C0116 for missing function or method docstring) in the search bar here in the Pylint Documentation, and you should find an example for how to fix your problem.

The TAs will complete a manual code review for each assignment to confirm that you have followed the requirements and directions on this document. Deductions will occur on each test case that fails to follow requirements.

Submission

Follow these steps for submission.

Check	Description
▢	Verify that you have no debugging statements left in your code.
▢	Submit **only** poly.py (the starter code file with your updates) to the given assignment in Gradescope. This will cause the grading scripts to run.
▢	If you have a partner, make sure you are submitting it for you **and** your partner on Gradescope. See this Gradescope documentation for more details.

When the grading scripts are complete, check the results. If there are errors, evaluate the script feedback, work on the fix, test the fix, and then re-submit the file to Gradescope. You can submit as many times as necessary before the due date.

NOTE: By default, only the most recent submission will be considered for grading. If you want to use a previous submission for your final grade, you must activate it from your submission history before the due date.

Academic Integrity

Please review the Academic Integrity section of the syllabus. We will be using plagiarism checkers, and we will cross-reference your solution with AI-generated solutions to check for similarities. Remember the goal in this class is not to write the perfect solution; we already have many of those! The goal is to learn how to problem solve, so:

- Don't hesitate to ask for guidance from the instructional staff.
- Discuss with peers about **only** bugs, their potential fixes, **high-level** design decisions, and project clarifying questions. Do not look at, verbalize, or copy another student's code when doing so.

Attribution

Thanks to Dr. Carol Ramsey and Dr. Shyamal Mitra for the instructions template and this assignment.