Polynomial 多項式報告

41243103 林采儀

2024/11/22

解題說明

實作一個多項式類別 (Polynomial)，並支持多項式的加法、乘法與求值操作。此外，要求重載輸入 (>>) 與輸出 (<<) 運算子，以提升多項式的使用便捷性。

設計與實作細節

類別結構

1. Term 類別
   - **目的**：用來表示多項式中的單一項式。
   - **屬性**：
     - coef：項式的係數。
     - exp：項式的指數。
   - **功能**：
     - 提供基本的存取方法來操作係數與指數。
     - 提供朋友類別支援，協助在 Polynomial 類別中進行操作。
2. Polynomial 類別
   - **目的**：用來表示和處理多項式。
   - **屬性**：
     - termArray：動態陣列，存儲多項式中的所有 Term 物件。
     - capacity：動態陣列的容量，用於在項目數超過當前容量時自動擴展陣列。
     - terms：當前多項式中的項數。
   - **功能**：
     - **新增項目 (NewTerm)**：檢查多項式是否已達容量，若已滿則自動擴展陣列容量。
     - **加法 (Add)**：將兩個多項式的同指數項進行合併，並返回一個新的多項式。
     - **乘法 (Mult)**：對兩個多項式進行逐項相乘，合併相同指數的項，返回一個新的多項式。
     - **排序 (BubbleSort)**：使用冒泡排序演算法按指數從大到小排列多項式項。
     - **合併(combine)**: 合併結果多項式中相同指數的項次。
     - **求值 (Eval)**：給定變數 x，計算多項式在該點的值。
     - **運算子多載 (<<, >>)**：實現多項式的輸入與輸出，方便使用者進行使用。

優點

1. **抽象化資料結構設計**：
    - 使用 Term 類別封裝每一個項式，讓程式的結構更加清晰，易於維護。
    - Polynomial 類別提供多項式的操作，例如加法、乘法和求值，使數學運算簡潔明了。
2. **彈性記憶體管理**：
    - termArray 使用動態陣列來存儲多項式項目，能根據需求自動調整容量，提供靈活的記憶體管理機制。
    - 在拷貝建構子和運算子重載中，使用 new 和 delete 確保正確分配與釋放記憶體，避免記憶體洩漏。
3. **多載運算子支援**：
    - 使用 >> 和 << 這兩個運算子進行多項式的輸入和輸出，這樣的設計符合 C++ 語言的習慣，讓使用者能夠輕鬆操作。
4. **功能完整**：
    - 提供多項式的加法、乘法、排序和求值功能，能夠滿足多項式運算的大多數需求。

缺點

1. **排序效率低**：
    - 當多項式的項目數量較多時，泡沫排序（BubbleSort）的時間複雜度為 $O(n^2)$，效率較低。
2. **合併過程性能不足**：
    - 目前在多項式排序後，會對每一對項目進行逐項合併，這樣的複雜度為 $O(n^2)$。隨著項目數量增加，合併過程的計算時間會迅速增長。
3. **動態記憶體管理成本**：
    - 雖然動態調整陣列容量提高了靈活性，但在頻繁進行項目新增操作時，陣列擴展會帶來額外的開銷。這樣的設計對記憶體管理效率要求較高，若頻繁擴展陣列，可能會對效能產生影響。

設計手稿(初稿)

HW2 Polynomial

多項式的結構

class Polynomial

*TermArray	
capacity	(陣列容量)
terms	(陣列已使用容量)

class Term

coef	exp
⋮	⋮

加法 Add()

$P_1(x)$: $3x^3 + 2x^2 + x$

$P_2(x)$: $2x^3 - 2x + 5$

$5x^3 + 2x^2 - x + 5$

* 比較最高次項指數，若相同則合併係數，否則將指數較大的項直接加入結果中。
* 持續處理所有項直到兩個多項式的所有項目都被遍歷。

乘法 Mult()

$P_1(x)$: $3x^2 + x$

$P_2(x)$: $x^2 + 2x$

$P_1(x)$ \ $P_2(x)$	x^2	$2x$
$3x^2$	$3x^{2+2}$	$(2\times3)x^{2+1}$
x	x^{2+1}	$2x^{1+1}$

$\Rightarrow 3x^4 + 7x^3 + 2x^2$

* 逐項相乘，計算每對項目的係數積和指數和。
* 若結果中有相同指數的項目，則合併它們的係數。

排序 BubbleSort()

* 泡沫排序
* 逐一比較相鄰兩項指數，若指數不符合排序要求(大→小)，則交換位置。

新增項目 NewTerm()

* 檢查 terms 是否等於 capacity (當前儲存項數是否達到陣列最大容量)
* 若容量不足，擴展 2 倍容量。
  1. 建立一個新陣列
  2. 將舊陣列內容 copy 到新陣列。
  3. 刪除舊陣列，釋放記憶體。
* 儲存新項目(係數和指數)

輸出入 (<<、>> 多載)

* 使用者輸入：3 2 (表示 $3x^2$)
* >> ⇒ 儲存為 Term(3, 2)
* << ⇒ 輸出：3x^2

效能分析

NewTerm(新增項目)：

- **時間複雜度**：O(n)，需要遍歷項目以確認是否需要擴展儲存空間。
- **空間複雜度**：O(n)，由於使用動態陣列來儲存多項式的項次。

Add (加法)：

- **時間複雜度**：O(n+m)，需要遍歷兩個多項式的項次進行加法運算，其中 n 是兩個多項式中項數的總和。
- **空間複雜度**：O(n + m)，其中 n 和 m 分別是兩個多項式的項數，最終會生成一個新的多項式來存儲結果。

Mult (乘法)：

- **時間複雜度**：O(n * m)，需要遍歷兩個多項式的每一項進行乘法運算。
- **空間複雜度**：O(n * m)，由於生成的結果多項式可能包含最多 n * m 項。

Combine (合併):

- **時間複雜度**：

  最壞情況：O(n^2)，每一項次與其後所有項次進行比較，共需執行：

  最佳情況：若多項式中所有項次的指數均不相同，則無需合併，時間複雜度為 O(n)。

- **空間複雜度**：O(1)，合併過程中僅透過陣列內部操作完成

BubbleSort (泡沫排序)：

- **時間複雜度**：

  最壞情況：O(n^2)，當所有項次完全亂序時，外層迴圈執行 n−1 次，內層迴圈執行 n-i-1 次，是平方級別的操作。

  最佳情況：多項式項次已排序且使用「swapped」檢測，第一輪後即可退出，時間複雜度降為 O(n)。若未使用「swapped」，仍需執行 O(n^2) 的比較操作。

- **空間複雜度**：O(1)，排序過程中只需要常數額外空間來交換項次。

```
// 泡沫排序：將多項式按照指數從大到小排序
Polynomial Polynomial::BubbleSort()
{
    bool swapped; // 標記是否進行交換
    for (int i = 0; i < terms - 1; i++)
    {
        swapped = false;
        for (int j = 0; j < terms - i - 1; j++) // 後面的元素已排好序，減少內層迴圈次數
        {
            if (termArray[j].exp < termArray[j + 1].exp) // 指數較小則交換
            {
                Term temp = termArray[j];
                termArray[j] = termArray[j + 1];
                termArray[j + 1] = temp;
                swapped = true; // 標記進行過交換
            }
        }
        if (!swapped) break; // 若未交換，表示已排序完成，提早退出
    }
    return *this;
}
```

Eval (求值)：

- **時間複雜度**：O(n)，需要遍歷所有項次來計算多項式的值。
- **空間複雜度**：O(1)，只需要少量的額外空間來儲存中間結果。

測試與驗證

測試數據 & 測資設計

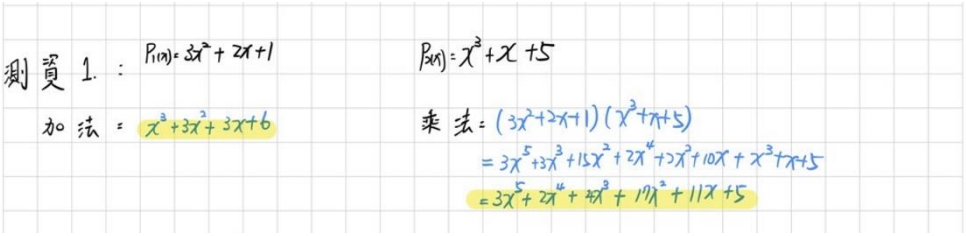
測資 1：基本加法與乘法測試

輸入：

- 多項式 1：$3x^2+2x+1$
- 多項式 2：x^3+x+5

預期結果：

- 加法結果：x^3+3x^2+3x+6
- 乘法結果：$3x^5+2x^4+4x^3+17x^2+11x+5$



實際結果：

```
Enter the first polynomial
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 3 2
Enter the coefficient and exponent for term 2: 2 1
Enter the coefficient and exponent for term 3: 1 0

Enter the second polynomial
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 1 3
Enter the coefficient and exponent for term 2: 1 1
Enter the coefficient and exponent for term 3: 5 0

P1(x) = 3x^2 + 2x + 1
P2(x) = x^3 + x + 5

P1(x) + P2(x) = x^3 + 3x^2 + 3x + 6
P1(x) * P2(x) = 3x^5 + 2x^4 + 4x^3 + 17x^2 + 11x + 5

Test variables x={ 0, 1, 2, -1 }
Add P(0) =  6          Mult P(0) =  5
Add P(1) =  13         Mult P(1) =  42
Add P(2) =  32         Mult P(2) =  255
Add P(-1) =  5         Mult P(-1) =  6
```

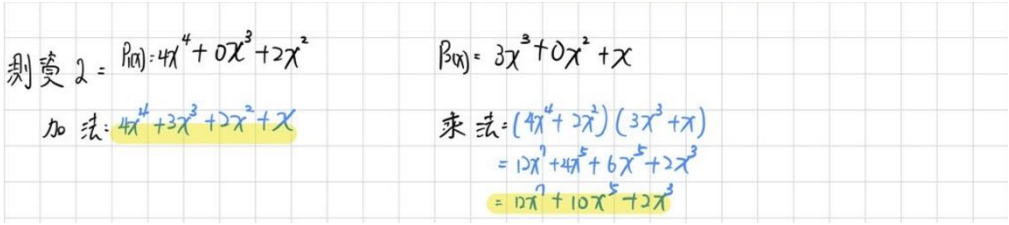
測資 2：包含零係數的多項式

輸入：

- 多項式 1：$4\boldsymbol{x^4}+0\boldsymbol{x^3}+2\boldsymbol{x^2}$
- 多項式 2：$3\boldsymbol{x^3}+0\boldsymbol{x^2}+\boldsymbol{x}$

預期結果：

- 加法結果：$4\boldsymbol{x^4}+3\boldsymbol{x^3}+2\boldsymbol{x^2}+\boldsymbol{x}$
- 乘法結果：$12\boldsymbol{x^7}+10\boldsymbol{x^5}+2\boldsymbol{x^3}$



實際結果：

```
Enter the first polynomial:
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 4 4
Enter the coefficient and exponent for term 2: 0 3
Enter the coefficient and exponent for term 3: 2 2

Enter the second polynomial
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 3 3
Enter the coefficient and exponent for term 2: 0 2
Enter the coefficient and exponent for term 3: 1 1

P1(x) = 4x^4 + 2x^2
P2(x) = 3x^3 + x

P1(x) + P2(x) = 4x^4 + 3x^3 + 2x^2 + x
P1(x) * P2(x) = 12x^7 + 10x^5 + 2x^3

Test variables x={ 0, 1, 2, -1 }
Add P(0) =  0           Mult P(0) =  0
Add P(1) =  10          Mult P(1) =  24
Add P(2) =  98          Mult P(2) =  1872
Add P(-1) =  2          Mult P(-1) =  -24
```

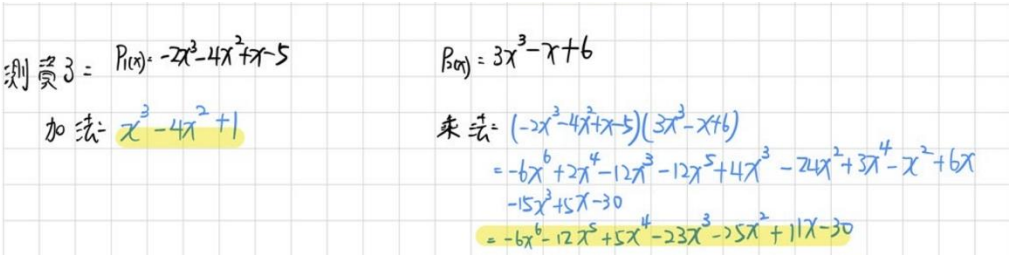
測資 3：包含負係數的多項式

輸入：

- 多項式 1：$-2x^3-4x^2+x-5$
- 多項式 2：$3x^3-x+6$

預期結果：

- 加法結果：x^3-4x^2+1
- 乘法結果：$-6x^6-12x^5+5x^4-23x^3-25x^2+11x-30$



實際結果：

```
Enter the first polynomial:
Enter the number of terms in the polynomial: 4
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: -2 3
Enter the coefficient and exponent for term 2: -4 2
Enter the coefficient and exponent for term 3: 1 1
Enter the coefficient and exponent for term 4: -5 0

Enter the second polynomial
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 3 3
Enter the coefficient and exponent for term 2: -1 1
Enter the coefficient and exponent for term 3: 6 0

P1(x) =  - 2x^3 - 4x^2 + x - 5
P2(x) = 3x^3 - x + 6

P1(x) + P2(x) = x^3 - 4x^2 + 1
P1(x) * P2(x) =  - 6x^6 - 12x^5 + 5x^4 - 23x^3 - 25x^2 + 11x - 30

Test variables x={ 0, 1, 2, -1 }
Add P(0) =  1           Mult P(0) =  -30
Add P(1) =  -2          Mult P(1) =  -80
Add P(2) =  -7          Mult P(2) =  -980
Add P(-1) =  -4         Mult P(-1) =  -32
```

測資 4：僅有常數項的多項式

輸入：

- 多項式 1：7
- 多項式 2：-3

預期結果：

- 加法結果：4
- 乘法結果：-21

實際結果：

```
Enter the first polynomial:
Enter the number of terms in the polynomial: 1
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 7 0

Enter the second polynomial
Enter the number of terms in the polynomial: 1
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: -3 0

P1(x) = 7
P2(x) =  - 3

P1(x) + P2(x) = 4
P1(x) * P2(x) =  - 21

Test variables x={ 0, 1, 2, -1 }
Add P(0) =  4           Mult P(0) =  -21
Add P(1) =  4           Mult P(1) =  -21
Add P(2) =  4           Mult P(2) =  -21
Add P(-1) =  4          Mult P(-1) =  -21
```

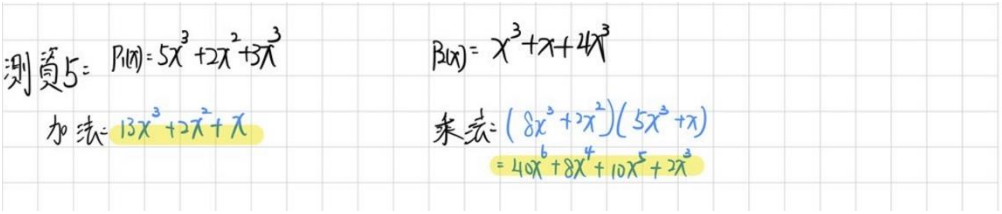
測資 5：包含相同指數的多項式項

輸入：

- 多項式 1：$5x^3 + 2x^2 + 3x^3$
- 多項式 2：$x^3 + x + 4x^3$

預期結果：

- 加法結果：$13x^3 + 2x^2 + x$
- 乘法結果：$40x^6 + 10x^5 + 8x^4 + 2x^3$



實際結果：

```
Enter the first polynomial
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 5 3
Enter the coefficient and exponent for term 2: 2 2
Enter the coefficient and exponent for term 3: 3 3

Enter the second polynomial
Enter the number of terms in the polynomial: 3
(Input example: 3 2 means 3x^2)
Enter the coefficient and exponent for term 1: 1 3
Enter the coefficient and exponent for term 2: 1 1
Enter the coefficient and exponent for term 3: 4 3

P1(x) = 8x^3 + 2x^2
P2(x) = 5x^3 + x

P1(x) + P2(x) = 13x^3 + 2x^2 + x
P1(x) * P2(x) = 40x^6 + 10x^5 + 8x^4 + 2x^3

Test variables x={ 0, 1, 2, -1 }
Add P(0) =  0           Mult P(0) =  0
Add P(1) =  16          Mult P(1) =  60
Add P(2) =  114         Mult P(2) =  3024
Add P(-1) =  -12                Mult P(-1) =  36
```

測試結論

1. 程式在加法與乘法操作中能正確處理零係數、負係數及相同指數項的合併。
2. 多項式的輸入輸出格式清晰，符合題目需求。
3. 排序功能正確，結果按照指數從大到小排列。
4. 求值功能準確，能正確計算指定 x 值下的多項式值。

申論及開發報告

挑戰

在處理多項式時，主要的挑戰在於如何應對多項式項數不同與指數不一致的情況。由於多項式的項數和每個項的指數並非固定，這需要設計靈活的資料結構來動態處理。此外，動態記憶體管理也是一個挑戰，特別是在需要根據運算需求擴充記憶體時，如何有效釋放和管理記憶體是一個需要謹慎處理的問題。另外，運算子重載的實現也需要確保正確性，特別是在進行多項式相加、相乘等操作時，重載的運算符需要維持正確的邏輯與效能。

收穫

通過這次的學習，我對抽象資料型別（ADT）的設計有了更深入的理解。如何設計一個能夠靈活處理各種情況的資料結構，並且通過處理多項式加法、乘法等操作，對我理解資料結構的應用有很大的幫助。同時，在 C++ 中動態記憶體分配與運算子重載的技術也進一步的學習，尤其是在面對需要靈活擴展儲存空間的情況下，能夠有效地運用動態記憶體來提升程式的彈性與效能。

改進方向

1. **排序效能的提升**
    - 目前多項式項次的排序使用泡沫排序（BubbleSort），該方法的時間複雜度為 $O(n^2)$，對於項數較多的多項式，排序效率偏低。
    - 未來可以考慮改用更高效的排序演算法，例如快速排序（QuickSort）或合併排序（MergeSort），其時間複雜度為 $O(n \log n)$，對於大項數多項式的處理，能有效提升排序效能。
2. **合併操作的優化**
    - 在多項式的加法和乘法中，對於指數相同的項次需要進行合併。目前的合併操作採用了雙層迴圈，時間複雜度為 $O(n^2)$。
    - 可以嘗試通過排序後的單次遍歷來進行合併，將時間複雜度降低至 $O(n)$。此外，使用哈希表來記錄項次的指數，進一步優化查找與合併的效率。

3. **功能的擴展**

- 現有操作僅實現了加法、乘法、排序與求值功能，未來可以考慮加入多項式的減法、乘法操作，甚至是多項式分解功能，以進一步提升系統的應用價值。

結論

本次作業從資料結構的設計到程式的實現，讓我充分體會到設計靈活、高效系統的重要性。

雖然在排序與合併操作上仍存在效率瓶頸，但通過作業實作，我學會了如何應用 C++ 的動態記憶體分配技術，並深入掌握了運算子重載的實現。

我將繼續探索更高效的演算法與資料結構設計方法，並嘗試為多項式運算提供更完整的解決方案。也將此次寶貴的實作經驗運用在日後的學習上，提升自我實力。