資料結構

作業三

41243152 劉康申

解題說明

題目背景

多項式運算在科學計算和工程應用中十分常見。為高效處理多項式的加法、乘法和帶值運算，這次作業採用循環鏈結串列實現多項式的存儲與操作。此資料結構能有效節省空間並提高操作靈活性，適合動態更新多項式結構的應用場景。

題目解析

- 每個多項式由數個項組成，項由係數(coef)與指數(exp)描述。
- 採用循環鏈結串列，其中每個節點表示一個多項式項，包含：
    1. `coef`：整數係數。
    2. `exp`：指數，按照降序排列。
    3. `link`：指向下一個節點的指標。
- 多項式的外部表示形式為整數序列，便於輸入與輸出。
- 項目的操作需求包括：
    1. 新增節點到鏈結串列。
    2. 加法與乘法操作。
    3. 計算特定值的多項式值。

程式實作

程式實現包括以下主要部分:

節點結構

每個多項式的項目用 Term 節點表示, 包含:

- **coef:** 整數係數
- **exp:** 指數
- **link:** 指向下一節點的指標

多項式類別

主要實現以下功能:

1. 建構與解構:
    - 初始化一個帶有標頭節點的循環連結串列。
    - 確保正確釋放資源避免記憶體洩漏。
2. 輸入與輸出運算子重載:
    - 提供友好的多項式輸入與輸出方式。
3. 加法與乘法運算:
    - 加法:遍歷兩個多項式, 依據指數合併係數。
    - 乘法:逐項相乘並插入結果多項式。
4. 帶值運算:
    - 計算多項式在某點的值。
5. 賦值運算與拷貝建構:
    - 支持多項式的深層拷貝與自我賦值檢測。

效能分析

1. 空間複雜度：
    - 每個多項式的節點使用額外的指標儲存，空間複雜度為 O(n)，其中 n 為多項式的項數。
2. 時間複雜度：
    - 加法：O(n+m)，其中 n 和 m 為兩個多項式的項數。
    - 乘法：O(n \times m)，需要對兩多項式的所有項逐一相乘。
    - 輸入/輸出：O(n)，每個節點需被訪問一次。
    - 帶值運算：O(n)，需計算每項的值並累加。

測試與驗證

測試案例

以下是針對多項式操作的測試:

```
輸入第一個多項式:
輸入多項式的項數: 3
輸入係數和指數 (coef exp): 3 3
輸入係數和指數 (coef exp): 2 2
輸入係數和指數 (coef exp): 1 1
輸入第二個多項式:
輸入多項式的項數: 3
輸入係數和指數 (coef exp): 4 2
輸入係數和指數 (coef exp): 3 1
輸入係數和指數 (coef exp): 5 0
p1: 3x^3+2x^2+x
p2: 4x^2+3x+5
Sum: 3x^3+6x^2+4x+5
Product: 12x^5+17x^4+25x^3+13x^2+5x
輸入代入的值 x: 2
p1(2) = 34
p2(2) = 27
```

```
輸入第一個多項式:
輸入多項式的項數: 4
輸入係數和指數 (coef exp): 6 3
輸入係數和指數 (coef exp): 4 2
輸入係數和指數 (coef exp): 7 1
輸入係數和指數 (coef exp): 5 0
輸入第二個多項式:
輸入多項式的項數: 2
輸入係數和指數 (coef exp): 6 2
輸入係數和指數 (coef exp): 6 0
p1: 6x^3+4x^2+7x+5
p2: 6x^2+6
Sum: 6x^3+10x^2+7x+11
Product: 36x^5+24x^4+78x^3+54x^2+42x+30
輸入代入的值 x: 4
p1(4) = 481
p2(4) = 102
```

驗證

測試結果與理論計算結果一致，證明實作正確無誤。

申論及開發報告

問題與挑戰

在開發過程中，主要挑戰為：

1. 多項式項目的插入與排序：需要確保指數降序排列且合併同類項。
2. 記憶體管理：需避免記憶體洩漏並確保拷貝操作的正確性。

改進空間效率

1. 使用靜態記憶體池分配節點以減少頻繁的記憶體分配操作。
2. 優化節點的刪除操作，避免重複遍歷。

未來展望

1. 擴展功能：
    - 支援多項式的導數與積分運算。
    - 支援更多的數學運算如除法與模運算。
2. 改進效能：
    - 使用跳表結構優化查詢與插入性能。
3. 友好介面：
    - 增加圖形化界面，方便用戶輸入與觀察多項式結果。

心得

本專案充分運用了循環鏈結串列的靈活性，實現了多項式的高效操作。然而，串列的缺點在於訪問特定元素時效率較低。在未來的優化中，可以結合平衡樹或跳表等資料結構，提高對特定項操作的效率。此外，可以拓展類別功能，例如實現多項式微分與積分等高階運算功能，以滿足更廣泛的應用需求。