Work Two

劉向榮

2024 10 23

Problems

1. **Implement the Polynomial class its ADT and private data members are shown in Figure 1 and 2, respectively.**
2. **Write C++ functions to input and output polynomials represented as Figure 2. Your functions should overload the << and >> operators.**

解題步驟如下:

1. 定義結構體 (Structure) 和節點 (Node)

使用 ChainNode 來定義每一個多項式的項，每個項包含：

coefficient：多項式係數（如 2, -4, 3 等）。

exponent：指數（如 3, 2, 1 等）。

next：指向下一個項的指標，用來連接鏈表。

2. 定義多項式鏈表 (Polynomial Chain) 類

Chain 類型表示多項式，包含：

first：指向多項式第一個項的指標。

方法：包括 插入、刪除、搜尋、加法、減法、乘法等操作。

3. 插入多項式項 (Insert Method)

插入項是按照 指數大小排序 插入到鏈表中，一定確保多項式項是有序的。

插入過程是遍歷鏈表，找到合適位置後進行插入。

4. 多項式加法 (Polynomial Addition)

為了進行多項式加法，需要遍歷兩個多項式的鏈表。

如果兩個多項式有相同的指數，就將係數相加，並合併該項。

如果指數不同，則將較小指數的項直接插入結果多項式中。

5. 多項式減法 (Polynomial Subtraction)

減法的操作與加法類似，但需要對另一個多項式的項進行取反處理。

每個項的係數需要乘以 -1，然後進行加法操作。

6. 多項式乘法 (Polynomial Multiplication)

乘法需要將每一項進行相乘。

當兩個多項式的每一項相乘時，兩項的 指數相加，係數 相乘。

然後需要將結果項按指數合併（如果有重複的指數，則合併它們的係數）。

7. 顯示結果 (Print Result)

遍歷結果鏈表，將每一項按適當格式顯示出來。

效能分析

1. 時間複雜度 (Time Complexity)

input()：O(m^2)，其中 m 是多項式中項的數量。

insertTerm()：O(n)，每次插入新項時可能需要遍歷所有項。

operator+ 和 operator-：O(n + m)，遍歷兩個多項式。

operator*：O(n * m)，對於兩個多項式進行乘法運算。

2. 空間複雜度 (Space Complexity)

空間複雜度主要取決於多項式中項的數量，因此對於 n 項的多項式，空間複雜度是 **O(n)**。

運算過程中生成的新多項式會額外佔用空間，總空間複雜度為 **O(n + m)**。

測試與驗證

舉例測資:

$2X^3 + 3X^4 - 4X + 9$ 、$9X^4 - 2X^3 + 1$

正確結果為

加法=$12X^4 - 4X^1 + 10$

減法=$-6X^4 + 4X^3 - 4X^1 + 8$

乘法=27X^8 + 12X^7 − 4X^6 − 36X^5 + 92X^4 − 16X^3 − 4X^1 + 9　(太長)

其實際運算結果為

```
Enter first polynomial (coefficient and exponent, end with 0 0):
2 3 3 4 -4 1 9 0
0 0
Enter second polynomial (coefficient and exponent, end with 0 0):
9 4 -2 3 1 0
0 0
First Polynomial: 3X^4+2X^3-4X^1+9
Second Polynomial: 9X^4-2X^3+1
Sum of polynomials: 12X^4-4X^1+10
Difference of polynomials: -6X^4+4X^3-4X^1+8
Product of polynomials: 27X^8+12X^7-4X^6-36X^5+92X^4-16X^3-4X^1+9
```

申論及心得

我自己認為這項作業最難的部分，是想像連結串列(鏈表)中的資料分布，每一筆指數與係數在指向下一個目標得時候，往往都會使腦袋打結，但是反而在運用的過程上面，相較簡單，不管是加減乘都是，還有是在運算的過程上每一筆資料的排序大小，一旦排序沒做好後面完全不知道在幹嘛，至於解題想法只是在之前的基礎上拓展開來與連結串列配合，不過我自己覺得在連結串列上，來實作多項式這種題目，其實還不如使用動態記憶體配合矩陣來得輕鬆，串來串去得十分抽象，不過也是個讓我更加理解連結串列的用法。