資料結構報告

張凱茗

2024/11/23

目錄

解題說明

Problem 1: 根據題目要求實作 Polynomial 類別成員(圖 1)，透過 termArray 存放 Tern 成員

Problem 2:多載>>和<<(圖 2 及圖 3)

```
class Term {
    friend class Polynomial;
    friend ostream& operator<<(ostream& os, const Polynomial& poly);
private:
    float coef; //係數
    int exp;    //指數
};

class Polynomial {
private:
    Term* termArray; //Term 陣列
    int capacity;    //陣列大小
    int terms;       //非零項數量
```

圖 1

```
//重載輸入運算子>>
friend istream& operator>>(istream& is, Polynomial& poly) {
    int n;
    cout << "請輸入多項式的項數:";
    is >> n;
    for (int i = 0; i < n; ++i) {
        float coef;
        int exp;
        cout << "請輸入第" << i + 1 << "項的係數和指數:";
        is >> coef >> exp;
        poly.addTerm(coef, exp);
    }
    return is;
}
```

圖 2

```
//重載輸出運算子<<
friend ostream& operator<<(ostream& os, const Polynomial& poly) {
    if (poly.terms == 0) {
        os << "0"; //多項式為空輸出 0
        return os;
    }
    for (int i = 0; i < poly.terms; ++i) {
        if (i > 0 && poly.termArray[i].coef > 0) os << " + ";
        os << poly.termArray[i].coef << "x^" << poly.termArray[i].exp;
    }
    return os;
}
```

圖 3

演算法設計與實作

Problem 1:如圖 4、圖 5、圖 6

```
//新增單項式
void addTerm(float coef, int exp) {
    if (coef == 0) return; //忽略係數0

    //搜尋同指數
    for (int i = 0; i < terms; ++i) {
        if (termArray[i].exp == exp) {
            termArray[i].coef += coef; //合併同項
            if (termArray[i].coef == 0) { //係數變成0刪除
                for (int j = i; j < terms - 1; ++j) {
                    termArray[j] = termArray[j + 1];
                }
                --terms;
            }
            return;
        }
    }

    //無相同指數的項時新增
    if (terms == capacity) {
        resize(); //空間不足時加大
    }
    termArray[terms].coef = coef;
    termArray[terms].exp = exp;
    ++terms;
}
```

圖 4

```
//多項式相加
Polynomial Add(const Polynomial& poly) const {
    Polynomial result;
    for (int i = 0; i < terms; ++i) {
        result.addTerm(termArray[i].coef, termArray[i].exp);
    }
    for (int i = 0; i < poly.terms; ++i) {
        result.addTerm(poly.termArray[i].coef, poly.termArray[i].exp);
    }
    return result;
}

//多項式相乘
Polynomial Mult(const Polynomial& poly) const {
    Polynomial result;
    for (int i = 0; i < terms; ++i) {
        for (int j = 0; j < poly.terms; ++j) {
            float newCoef = termArray[i].coef * poly.termArray[j].coef;
            int newExp = termArray[i].exp + poly.termArray[j].exp;
            result.addTerm(newCoef, newExp); //合併或新增同類項
        }
    }
    return result;
}
```

圖 5

```
//多項式計算
float Eval(float x) const {
    float result = 0;
    for (int i = 0; i < terms; ++i) {
        result += termArray[i].coef * pow(x, termArray[i].exp);
    }
    return result;
}
```

圖 6

效能分析

Problem 1:

時間複雜度：

addTerm 函式最壞為 **O(n)**

add 函式最壞為 **O(n^2)**

Mult 函式最壞為 **O(n1·n2·n)**

Eval 函式度為 **O(n)**

operator>>為 **O(n^2)**

operator<<為 **O(n)**

空間複雜度

termArray 初始容量為 10，當項數超出容量時，每次容量翻倍，最終總空間需求為 O(n)

Add 函式在最壞情況下空間需求為 O(n1+n2)

Mult 函式:最壞情況下空間需求為 O(n1·n2)

測試與過程

Problem 1:

```
請輸入第一個多項式:
請輸入多項式的項數:3
請輸入第1項的係數和指數:2 2
請輸入第2項的係數和指數:2 1
請輸入第3項的係數和指數:2 0
請輸入第二個多項式:
請輸入多項式的項數:2
請輸入第1項的係數和指數:2 1
請輸入第2項的係數和指數:2 0
第一個多項式為:2x^2 + 2x^1 + 2x^0
第二個多項式為:2x^1 + 2x^0
兩個多項式相加的結果為:2x^2 + 4x^1 + 4x^0
兩個多項式相乘的結果為:4x^3 + 8x^2 + 8x^1 + 4x^0
請輸入一個值來計算第一個多項式的值:2
第一個多項式在x=2時的值為:14
```

驗證

$(2x^2 + 2x + 2) + (2x + 2) = (2x^2 + 4x + 4)$

$(2x^2 + 2x + 2)(2x + 2) = (4x^3 + 8x^2 + 8x + 4)$

$(2 * 2^2 + 2 * 2 + 2) = 14$

申論及開發報告

在寫這次的功課的問題二時忘記在 Term 裡面 friend 重載運算子，導致重載運算子無法存取 exp 和 coef，檢查之後解決才成功完成，在這次的實作中讓我對多項式的存放方式及運算更加熟悉，也對輸入、輸出運算子的重構更加的了解