資料結構報告

林聖紘

Nov. 27 / 2024

Polynomial

解題說明

多項式由 Term 組成，每項包含：

- 係數（coef）：多項式中每項的倍數。
- 指數（exp）：變數的次方。

EX： $3x^2-4x+5$ 包含三項：

- $3x^2$：係數 3，指數 2；
- $-4x$：係數 -4，指數 1；
- +5：係數 5，指數 0。

多項式運算：

- 輸入：輸入建立多項式。
- 輸出：用數學式顯示多項式。
- 加法：對兩個多項式進行加法運算，結果應合併同指數的項，並按指數降冪排列。

演算法設計與設計

AddTerm

當新增一個項目時，需處理以下情況：

1. 將相同指數項的係數相加。
2. 動態擴展容量，確保能加入新項。
3. 維持降冪排序，以利後續操作。

```
50  void AddTerm(float coef, int exp) {
51      for (int i = 0; i < terms; ++i) {
52          if (termArray[i].getExp() == exp) {
53              termArray[i].setCoef(termArray[i].getCoef() + coef);
54              if (fabs(termArray[i].getCoef()) < 1e-6) {
55                  for (int j = i; j < terms - 1; ++j) {
56                      termArray[j] = termArray[j + 1];
57                  }
58                  --terms;
59              }
60              SortTerms();
61              return;
62          }
63      }
64      if (terms == capacity) {
65          Resize(2 * capacity);
66      }
67      termArray[terms].setCoef(coef);
68      termArray[terms].setExp(exp);
69      ++terms;
70      SortTerms();
71  }
```

演算法設計與設計

Operator(輸出格式化)

以降冪格式輸出多項式，需處理：

1. 第一項直接顯示符號，其後項以 + 或 - 分隔。
2. 特殊情況：
    - 係數為 1 或 -1 且指數非零，僅顯示變數。
    - 指數為 0，只顯示係數。

```
116 ostream& operator<<(ostream& out, const Polynomial& poly) {
117     if (poly.terms == 0) {
118         out << "0";
119         return out;
120     }
121
122     for (int i = 0; i < poly.terms; ++i) {
123         float coef = poly.termArray[i].getCoef();
124         int exp = poly.termArray[i].getExp();
125
126         if (i == 0) {
127             if (coef < 0) out << "-";
128             if (fabs(coef) != 1 || exp == 0) out << fabs(coef);
129         }
130         else {
131             if (coef > 0) out << " + ";
132             else out << " - ";
133             if (fabs(coef) != 1 || exp == 0) out << fabs(coef);
134         }
135
136         if (exp > 0) {
137             out << "x";
138             if (exp > 1) out << "^" << exp;
139         }
140     }
141
142     return out;
143 }
```

效能分析

主要影響因素

1. 排序開銷：
    - 排序是效能的主要瓶頸，特別是在新增項目或進行多項式相加時。
2. 記憶體管理：
    - 當多項式非常大時，頻繁的動態記憶體分配可能會造成額外的效能損失。

最差情況總時間複雜度

假設多項式的項目數為 nnn，則：

1. 新增所有項目：
   $O(n^2)$。
2. 多項式相加：
   $O((m+n)^2)$。

測試與過程

測試案例包含三種情境：簡單多項式、不同指數項目和相同多項式的加法。
第一個測試檢查項目合併及零係數項目移除；第二個測試驗證合併過程中項目排序的正確性，以及不同指數項的正確處理；第三個測試確保相同多項式加法後結果倍增。
過程中觀察程式對多項式操作的正確性，包括排序是否按降冪排列，動態記憶體管理是否有效，以及執行效率是否滿足預期，確保程式穩定處理各種多項式運算。