

一元稀疏多项式计算器 实验报告

howlinggggg

2022 年 10 月 16 日

1 实验任务

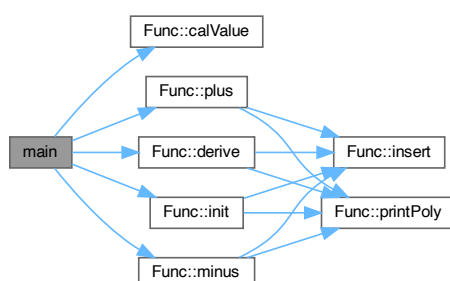
1.1 基本要求

1. 输入并建立多项式;
2. 输出多项式, 输出形式为整数序列; $n, c_1, e_1, c_2, e_2, \dots, c_n, e_n$, 其中 n 是多项式的项数, c_i 和 e_i 分别是第 i 项的系数和指数, 序列按指数降序排列;
3. 多项式 a 和 b 相加, 建立多项式 $a + b$;
4. 多项式 a 和 b 相减, 建立多项式 $a - b$.

1.2 选做内容

1. 计算多项式在 x 处的值。
2. 求多项式 a 的导函数 a'
4. 多项式的输出形式为类数学表达式.

2 概要设计



3 详细设计

3.1 class Func

```
1      在class定义了此次实验需要用到的函数
2 class Func
3 {
4 private:
5     vector<LNode *> head;
6
7 public:
8     Func()
9     {
10         for (int i = 0; i < 6; i++)
11         {
12             LNode *headx = new (LNode);
13             headx->coef = 0;
14             headx->next = NULL;
15             head.push_back(headx);
16         }
17     }
18     bool init();
19     void insert(LNode *node, int n);
20     void printPoly(int n);
21     void add();
22     void minus();
23     float calValue(float x);
24     void derive();
25 };
```

3.2 bool init()

初始化多项式 a 和多项式 b，将输入使用 insert() 插入 linklist 中

```
1 bool Func::init()
2 {
3     cout << "input poly a:";
4     float t;
5     vector<float> temp_a;
6     while (cin >> t)
7     {
8         temp_a.push_back(t);
```

```
9     }
10     if (temp_a.size() % 2 != 0)
11     {
12         cout << "error" << endl;
13         return false;
14     }
15     for (size_t i = 0; i < temp_a.size(); i += 2)
16     {
17         LNode *node = new (LNode);
18         node->coef = temp_a[i];
19         node->expn = temp_a[i + 1];
20         insert(node, 1);
21     }
22     cin.clear();
23     cin.ignore();
24
25     cout << "input poly b:";
26     vector<float> temp_b;
27     while (cin >> t)
28     {
29         temp_b.push_back(t);
30     }
31     if (temp_b.size() % 2 != 0)
32     {
33         cout << "input error" << endl;
34         return false;
35     }
36     for (size_t i = 0; i < temp_b.size(); i += 2)
37     {
38         LNode *node = new (LNode);
39         node->coef = temp_b[i];
40         node->expn = temp_b[i + 1];
41         insert(node, 2);
42     }
43     printPoly(1);
44     printPoly(2);
45     return true;
46 }
```

3.3 void insert(LNode *node, int n)

将 LNode 插入第 n 个 linklist 中，其中需要判断是否存在和插入的 LNode 相同的 expn，若存在则将 coef 相加，否则插入到 linklist 中并将后面的节点后移。

```
1 void Func::insert(LNode *node, int n)
2 {
3     LNode *t = head[n - 1];
4     while (t->next)
5     {
6         if (node->expn > t->next->expn)
7         {
8             node->next = t->next;
9             t->next = node;
10            ++head[n - 1]->coef;
11            return;
12        }
13        else if (node->expn == t->next->expn)
14        {
15            t->next->coef += node->coef;
16            return;
17        }
18        t = t->next;
19    }
20    if (!t->next)
21    {
22        node->next = t->next;
23        t->next = node;
24    }
25    ++head[n - 1]->coef;
26 }
```

3.4 void printPoly(int n)

实现输出多项式和选做内容 4，需要特别处理的是若 expn 为 0，则只输出 coef；若 coef 为 ± 1 则只输出 expn。在选做 4 中还必须特别考虑第一项，若第一项为 + 则只输出系数。

```
1 void printPoly(int n)
2 {
3     LNode *t = head[n - 1];
4     LNode *t1 = head[n - 1];
```

```

5      cout << head[n - 1]->coef;
6      while (t->next)
7      {
8          cout << ", " << t->next->coef << ", " << t->next->
          expn;
9          t = t->next;
10     }
11     cout << endl;
12
13     int temp = 0;
14
15     cout << head[n - 1]->coef << ":";
16     while (t1->next)
17     {
18         if (temp != 0 && t1->next->coef > 0)
19             cout << "+";
20         if (t1->next->coef == -1)
21             cout << "-";
22         if (t1->next->coef != 1 && t1->next->coef != -1)
23             cout << t1->next->coef;
24         else if ((t1->next->coef == 1 || t1->next->coef
                == -1) && t1->next->expn == 0)
25             cout << 1;
26         if (t1->next->expn != 0 && t1->next->expn != 1)
27             cout << "x^" << t1->next->expn;
28         else if (t1->next->expn == 1)
29             cout << "x";
30         t1 = t1->next;
31         temp++;
32     }
33     cout << endl;
34 }

```

3.5 void plus()

将多项式相加并插入 linklist[2], 若相加后系数为 0 则不插入

```

1 void Func::plus()
2 {
3     LNode *t1 = head[0];
4     LNode *t2 = head[1];
5     while (t1->next && t2->next)

```

```
6      {
7          if (t1->next->expn == t2->next->expn)
8          {
9              LNode *node = new (LNode);
10             node->coef = t1->next->coef + t2->next->coef;
11             node->expn = t1->next->expn;
12             if (node->coef != 0)
13                 insert(node, 3);
14             t1 = t1->next;
15             t2 = t2->next;
16         }
17         else if (t1->next->expn > t2->next->expn)
18         {
19             LNode *node = new (LNode);
20             node->coef = t1->next->coef;
21             node->expn = t1->next->expn;
22             insert(node, 3);
23             t1 = t1->next;
24         }
25         else
26         {
27             LNode *node = new (LNode);
28             node->coef = t2->next->coef;
29             node->expn = t2->next->expn;
30             insert(node, 3);
31             t2 = t2->next;
32         }
33     }
34     if (!t1->next)
35     {
36         while (t2->next)
37         {
38             LNode *node = new (LNode);
39             node->coef = t2->next->coef;
40             node->expn = t2->next->expn;
41             insert(node, 3);
42             t2 = t2->next;
43         }
44     }
45     else if (!t2->next)
46     {
```

```
47     while (t1->next)
48     {
49         LNode *node = new (LNode);
50         node->coef = t1->next->coef;
51         node->expn = t1->next->expn;
52         insert(node, 3);
53         t1 = t1->next;
54     }
55 }
56 printPoly(3);
57 }
```

3.6 void minus()

将多项式相减并插入 linklist[3], 若相减后系数为 0 则不插入

```
1 void Func::minus()
2 {
3     LNode *t1 = head[0];
4     LNode *t2 = head[1];
5     while (t1->next && t2->next)
6     {
7         if (t1->next->expn == t2->next->expn)
8         {
9             LNode *node = new (LNode);
10            node->coef = t1->next->coef - t2->next->coef;
11            node->expn = t1->next->expn;
12            if (node->coef != 0)
13                insert(node, 4);
14            t1 = t1->next;
15            t2 = t2->next;
16        }
17        else if (t1->next->expn > t2->next->expn)
18        {
19            LNode *node = new (LNode);
20            node->coef = t1->next->coef;
21            node->expn = t1->next->expn;
22            insert(node, 4);
23            t1 = t1->next;
24        }
25        else
26        {
```

```

27         LNode *node = new (LNode);
28         node->coef = -1 * t2->next->coef;
29         node->expn = t2->next->expn;
30         insert(node, 4);
31         t2 = t2->next;
32     }
33 }
34 if (!t1->next)
35 {
36     while (t2->next)
37     {
38         LNode *node = new (LNode);
39         node->coef = -1 * t2->next->coef;
40         node->expn = t2->next->expn;
41         insert(node, 4);
42         t2 = t2->next;
43     }
44 }
45 else if (!t2->next)
46 {
47     while (t1->next)
48     {
49         LNode *node = new (LNode);
50         node->coef = t1->next->coef;
51         node->expn = t1->next->expn;
52         insert(node, 4);
53         t1 = t1->next;
54     }
55 }
56 printPoly(4);
57 }

```

3.7 float calValue(float x)

```

1 float Func::calValue(float x)
2 {
3     float sum = 0;
4     LNode *t = head[0];
5     while (t->next)
6     {
7         sum += t->next->coef * pow(x, t->next->expn);

```



```
8         t = t->next;
9     }
10     return sum;
11 }
```

3.8 void derive()

求得 a' 并存入 linklist[4]

```
1 void Func::derive()
2 {
3     LNode *t = head[0];
4
5     while (t->next)
6     {
7         LNode *t2 = new (LNode);
8         if (t->next->expn != 0)
9         {
10             t2->coef = t->next->coef * t->next->expn;
11             t2->expn = t->next->expn - 1;
12             insert(t2, 5);
13         }
14         t = t->next;
15     }
16     printPoly(5);
17 }
```

4 测试结果

4.1 输入数据

多项式 a: $(2x + 5x^8 - 3.1x^{11})$, 多项式 b: $(7 - 5x^8 + 11x^9)$

5 实验总结

在此次实验中熟悉了对单链表的操作

```
howlin0523@Haolins-MacBook-Pro:~/Documents/GitHub/22F_da... ￼3
(base) → lab1 git:(main) x ./main
input poly a:2 1 5 8 -3.1 11 c
input poly b:7 0 -5 8 11 9 c
3,-3.1,11,5,8,2,1
3:-3.1x^11+5x^8+2x
3,11,9,-5,8,7,0
3:11x^9-5x^8+7
a+b=4,-3.1,11,11,9,2,1,7,0
4:-3.1x^11+11x^9+2x+7
a-b=5,-3.1,11,-11,9,10,8,2,1,-7,0
5:-3.1x^11-11x^9+10x^8+2x-7
a'(x)=3,-34.1,10,40,7,2,0
3:-34.1x^10+40x^7+2
input x:1
a(1)=3.9
(base) → lab1 git:(main) x
```