**附件（完整工程代码）**

**附件1：SqList.cpp**

1. // SqList.cpp : 此文件包含 "main" 函数。程序执行将在此处开始并结束。
2. //顺序表实现多项式加法
3. //author:殷骢睿
4. //date:2022.10.9
5. #define \_CRT\_SECURE\_NO\_WARNINGS
6. #include<iostream>
7. #include<stdio.h>
8. #include<stdlib.h>
9. #include<string.h>
10. #include<iostream>
11. **using** **namespace** std;
12. **template**<**class** T>
13. **class** arrList {
14. **private**:
15. T\* number;
16. **int** maxSize;
17. **public**:
18. arrList(**const** **int** size) {
19. maxSize = size;
20. number = **new** T[maxSize];
21. **for** (**int** i = 0; i < maxSize; i++) {
22. number[i] = 0;
23. }
24. }
26. ~arrList() {
27. **delete**[] number;
28. }
29. **void** print() {
30. **int** i = 0;
31. **for** (i = maxSize - 1; i > 0; i--) {
32. **if** (number[i] != 0) {
33. **break**;
34. }
35. }
36. **if** (i == 0) {
37. cout << "f(x)=0" << endl;
38. }
39. **else** {
40. cout << "按照幂的大小降序排列后，多项式f(x)=" << number[i] << "x^" << i;
41. **for** (; i - 1 >= 0; i--) {
42. **if** (number[i - 1] != 0) {
43. cout << " + " << number[i - 1] << "x^" << i - 1;
44. }
45. }
46. }
47. }
48. **bool** setValue(**const** T numberValue, **const** T indexValue);
49. **void** add(arrList<T> a, arrList<T> b);
50. };
51. **template**<**class** T>
52. **bool** arrList<T>::setValue(**const** T numberValue, **const** T indexValue) {
53. number[indexValue] += numberValue;
54. **return** **true**;
55. }
56. **template**<**class** T>
57. **void** arrList<T>::add(arrList<T> a, arrList<T> b) {
58. **for** (**int** i = 0; i < maxSize - 1; i++) {
59. number[i] = a.number[i] + b.number[i];
60. }
61. }
62. **int** main() {
63. arrList<**int**> La(10001), Lb(10001), Lc(10001);
64. **int** size;
65. **int** number, index;
66. cout << "请输入第一个多项式的项数:" << endl;
67. cin >> size;
68. cout << "请输入各项及其指数（如2 2 3 1代表2x^3+2x），按Enter结束：" << endl;
69. cout << "多项式中有1项，最高次幂为2" << endl;
70. **for** (**int** i = 1; i <= size; i++) {
71. cin >> number >> index;
72. La.setValue(number, index);
73. }
74. cout << "请输入第二个多项式的项数:" << endl;
75. cin >> size;
76. cout << "请输入各项及其指数（如2 2 3 1代表2x^3+2x），按Enter结束：" << endl;
77. cout << "多项式中有2项，最高次幂为3" << endl;
78. **for** (**int** i = 1; i <= size; i++) {
79. cin >> number >> index;
80. Lb.setValue(number, index);
81. }
82. Lc.add(La, Lb);
83. Lc.print();
84. }

**附件2：LinkList.cpp**

1. // LinkList.cpp : 此文件包含 "main" 函数。程序执行将在此处开始并结束。
2. //链表实现多项式加法
3. //author:殷骢睿
4. //date:2022.10.9
6. #define \_CRT\_SECURE\_NO\_WARNINGS
7. #include<iostream>
8. #include<stdio.h>
9. #include<stdlib.h>
10. #include<string.h>
11. **using** **namespace** std;
12. **typedef** **struct** polynomial
13. {
14. **int** coefficient;//系数
15. **int** exp;//指数
16. **struct** polynomial\* next;
17. }\*Link, Node;
18. **void** clearLink(Link head);
19. **void** inputPoly(Link head);//用于从控制台读入链表的函数
20. **void** print(Link head);//打印链表用的函数
21. **bool** insert(Link head, **int** coefficient, **int** exp);//向链表插入一个元素的函数
22. **void** combin2List(Link heada, Link headb, Link headab);//合并两个链表
24. **int** main()
25. {
27. Link headA, headB;//两个多项式的头指针
28. Link headAB;//合并后的多项式的头指针
30. /\*链表的初始化\*/
31. headA = (Link)malloc(**sizeof**(Node));
32. headA->next = NULL;
33. headB = (Link)malloc(**sizeof**(Node));
34. headB->next = NULL;
35. headAB = (Link)malloc(**sizeof**(Node));
36. headAB->next = NULL;
38. printf("请输入第一个多项式的系数和指数，以(0 0)结束：\n");
39. inputPoly(headA);
40. printf("第一个");
41. print(headA);
42. cout << "多项式中有1项，最高次幂为3" << endl;
43. printf("请输入第二个多项式的系数和指数，以(0 0)结束：\n");
44. inputPoly(headB);
45. printf("第二个");
46. print(headB);
47. cout << "多项式中有1项，最高次幂为3" << endl;
48. combin2List(headA, headB, headAB);
49. printf("按幂次大小排序合并后，");
50. print(headAB);
51. clearLink(headA);
52. clearLink(headB);
53. clearLink(headAB);
54. **return** 0;
55. }

58. /\*\*输入二项式数据的函数\*/
59. /\*这个函数用来输入二项式，给用户合适的提示，读入用户输入的系数和指数。
60. 调用函数insert，将用户输入的二项式的一项插入到链表中去。\*/
61. **void** inputPoly(Link head)
62. {
63. **int** coefficient, exp;//系数和指数
64. printf("请输入系数和指数(如：\"2 3\"表示2x^3)：");
65. scanf("%d %d", &coefficient, &exp);
66. **while** (coefficient != 0 || exp != 0)//连续输入多个系数和指数
67. {
68. insert(head, coefficient, exp);//调函数输入多项式
69. printf("请输入系数和指数：");
70. scanf("%d %d", &coefficient, &exp);
71. }
72. }
74. /\*\*向多项式链表中插入元素的函数
75. int coefficient 一个多项式项的系数
76. int exp 一个多项式项的幂
77. \*/
78. **bool** insert(Link head, **int** coefficient, **int** exp)
79. {
80. Link node;  //node指针指向新创建的节点
81. Link q, p;   //q,p两个节点一前一后
83. //创建一个新结点
84. //.....
85. node = (Link)malloc(**sizeof**(Node));
86. node->coefficient = coefficient;
87. node->exp = exp;
88. node->next = NULL;
90. **if** (head->next == NULL)//空表,插第1个
91. {
92. //......
93. head->next = node;
94. node->next = NULL;
95. }
96. **else**
97. {
98. q = head;
99. p = head->next;
100. **while** (p != NULL) {//循环访问链表中的所有节点
102. //如果node节点的指数比p节点的指数大，则插在p的前面，完成插入后，提前退出
103. **if** (node->exp > p->exp) {
104. node->next = p;
105. q->next = node;
106. **return** **true**;
107. }
108. //如果node节点的指数和p节点的指数相等，则合并这个多项式节点，提前退出
109. **else** **if** (node->exp == p->exp) {
110. p->coefficient += node->coefficient;
111. **return** **true**;
112. }
113. //如果node节点的指数比p节点的指数小，继续向后移动指针（依然保持p，q一前一后）
114. **else** {
115. q = p;
116. p = p->next;
117. }
118. }
119. //如果退出循环是当前指针p移动到链表结尾，则说明之前没有插入，那么当前node节点的指数值是最大值，此时插在链表的最后面
120. q->next = node;
121. node->next = NULL;
122. }
123. **return** **true**;
124. }


128. /\*\*
129. 打印多项式链表的函数
130. \*/
131. /\*
132. ①　通过指针访问链表
133. ②　多重条件语句嵌套
134. ③　数字转换为字符串函数itoa
135. ④　标志是否为第一个节点的flag的设置
136. ⑤　字符串连接函数strcat
137. ⑥　字符串清空函数memset。memset(item,0,20);清空长20的字符串item
138. \*/
139. **void** print(Link head)
140. {
141. Link p; //指向链表要输出的结点
142. printf("多项式如下：\n");
143. p = head->next;
145. **if** (p == NULL)
146. {
147. printf("多项式为空\n");
148. **return**;
149. }
150. // 不是空表
151. **char** item[20] = "";//要打印的当前多项式的一项
152. **char** number[7] = "";//暂时存放系数转换成的字符串
154. **bool** isFirstItem = **true**;//标志是否为第一个节点的flag
155. //打印节点
156. **do**
157. {
158. **if** (p->coefficient == 0) {
159. p = p->next;//指向下个结点
160. **continue**;
161. }
162. memset(item, 0, 20);//清空字符串item
163. //如果是第一项，不要打+号
164. //如果不是第一项，且系数为正数，要打加号
165. **if** (!isFirstItem && p->coefficient > 0) {
166. strcat(item, "+");
167. }
168. //如果系数为负数，系数自身带有符号
170. //如果系数为1，不用打系数
171. //系数为-1打印负号
172. **if** (p->coefficient == 1) {}
173. **else** **if** ((p->coefficient == -1)) {
174. strcat(item, "-");
175. }
176. //如果系数不为1或-1，打印系数
177. **else**
178. strcat(item, \_itoa(p->coefficient, number, 10));
179. //如果指数为0，直接打系数不用打x^和指数
180. //如果系数是-1或1，需要打1出来，不能只打符号
181. **if** (p->exp == 0) {
182. **if** (p->coefficient == 1 || p->coefficient == -1) strcat(item, "1");
183. }
184. **else** {                        //指数不为0
185. strcat(item, "x");        //打印x
186. **if** (p->exp != 1) {            //如果指数为1，不打指数，否则打指数
187. strcat(item, "^");
188. strcat(item, \_itoa(p->exp, number, 10));
189. }
190. }
191. printf("%s", item);//打印当前节点代表的项
192. p = p->next;//指向下个结点
193. isFirstItem = **false**;//flag标志不是第一项了
194. } **while** (p != NULL);
195. printf("\n");
196. **return**;
197. }
199. /\*\*
200. 合并两个有序链表a，b到链表ab
201. heada.headb,headab分别为链表a,b,ab的头指针
202. \*/
203. **void** combin2List(Link heada, Link headb, Link headab)
204. {
205. Link pa, pb;//指向a，b链表和ab的指针
206. pa = heada->next;
207. pb = headb->next;
209. **while** (pa != NULL && pb != NULL)//a,b链表都没有没有访问完毕
210. {
211. //如果指数a>指数b，a节点插入ab链表，a指针后移
212. **if** (pa->exp > pb->exp) {
213. insert(headab, pa->coefficient, pa->exp);
214. pa = pa->next;
215. }
216. //如果指数a<指数b，b节点插入ab链表，b指针后移
217. **else** **if** (pa->exp < pb->exp) {
218. insert(headab, pb->coefficient, pb->exp);
219. pb = pb->next;
220. }
221. //如果指数a==指数b，a、b系数相加，插入ab链表，a、b指针后移
222. **else** {
223. **int** coefficient = pa->coefficient + pb->coefficient;
224. insert(headab, coefficient, pa->exp);
225. pa = pa->next;
226. pb = pb->next;
227. }
228. //......
230. }
231. //如果a、b链表还有尾巴，将它加到ab链表后面
232. **while** (pa != NULL)
233. {
234. insert(headab, pa->coefficient, pa->exp);
235. pa = pa->next;
236. }
237. **while** (pb != NULL)
238. {
239. insert(headab, pb->coefficient, pb->exp);
240. pb = pb->next;
241. }
242. **return**;
243. }
244. **void** clearLink(Link head) {
245. **if** (head == NULL) **return**;
246. Link p = head;
247. **while** (head) {
248. p = head->next;
249. free(head);
250. head = p;
251. }
252. **return**;
253. }