


Full Name: Micah Nicole Chu Im
Email: micah_chuim@yahoo.com
Test Name: **MP8 B version 2**
Taken On: 4 May 2016 11:26:01 PHT
Time Taken: 31 min/ 90 min
Invited by: Ryan
Invited on: 3 May 2016 18:29:10 PHT
Tags Score:



Candidate Feedback: (_____) (; ;) (; ;) T.T.T.T

**Recruiter/Team
Comments:**

No Comments.

Question Description	Time Taken	Score	Status
Q1 The Polynomials Have Many Faces > Coding	19 min 19 sec	10/ 10	

QUESTION 1



Correct Answer

Score 10

The Polynomials Have Many Faces > Coding

QUESTION DESCRIPTION

The first line of input is the number of test cases, say t . t test cases will follow. Each test case is composed of 2 sets of numbers representing two polynomial expressions. Each polynomial expression is represented by a number n representing the number of terms present in the expression followed by n pairs of numbers representing the coefficient and exponent of each of the terms.

Represent the polynomial expressions using a list of terms.

Sample Input:

```
1
3
2 3 5 2 1 0
3
7 3 4 2 8 0
```

Sample output:

```
9x^3+9x^2+9
-5x^3+x^2-7
```

CANDIDATE ANSWER

Language used: **C**

```
1 #include <math.h>
```

```

2  #include <stdio.h>
3  #include <string.h>
4  #include <stdlib.h>
5  #include <assert.h>
6  #include <limits.h>
7  #include <stdbool.h>
8
9
10 typedef struct elem{
11     int coef,exp; //data
12     struct elem *next;
13 }node;
14
15 typedef struct{
16     node *head, *tail;
17     int size;
18 }list;
19
20
21 void init(list*);
22 void append(list*,int,int);
23 void display(list);
24 void display2(node*);
25 int deleteItem(list*,int);
26 node* itemAt(list,int);
27 int setItem(list*,int,int);
28 void simplify(list *l);
29 list add(list, list);
30 list sub(list,list);
31 void arrange(list *l);
32
33 int main() {
34
35     list l1,l2,sum,diff;
36     int t,n,x,y;
37     int n2;
38
39     init(&l1);
40     init(&l2);
41
42     scanf("%d",&t);
43
44     do{
45
46         scanf("%d",&n);
47
48         do{
49
50             scanf("%d %d",&x,&y);
51             append(&l1,x,y);
52             n--;
53         }while(n>0);
54
55         scanf("%d",&n2);
56
57         do{
58
59             scanf("%d %d",&x,&y);
60             append(&l2,x,y);
61
62             n2--;
63
64         }while(n2>0);
65
66
67         //display(l1);
68         //simplify(&l1);
69         //display(l1);
70         //display(l2);
71         //simplify(&l2);
72         //display(l2);
73         sum=add(l1,l2);
74         //simplify(&sum);
75         diff=sub(l1,l2);
76         //display(sum);

```

```

76 //display(sum);
77 //display(diff);
78 arrange(&sum);
79 display(sum);
80 arrange(&diff);
81 display(diff);
82
83 t--;
84
85 init(&l1);
86 init(&l2);
87
88 }while(t>0);
89
90 return 0;
91 }
92
93
94 void init(list* l){
95     l->size = 0;
96     l->head = l->tail = NULL;
97 }
98
99 void append(list* l,int x,int y){
100     node *n = malloc(sizeof(node));
101
102     n->coef = x;
103     n->exp = y;
104     n->next = NULL;
105     if(l->size==0){
106         l->head = l->tail = n;
107     }
108     else{
109         l->tail->next = n;
110         l->tail = n;
111     }
112     l->size++;
113 }
114
115
116 void display(list l){
117     node *tmp = l.head;
118     while(tmp!=NULL){
119
120         if(tmp->coef!=0){
121
122             if(tmp->exp==0)
123                 printf("%d",tmp->coef);
124             else if(tmp->exp==1)
125                 printf("%dx",tmp->coef);
126             else if(tmp->coef==1)
127                 printf("x^%d",tmp->exp);
128             else
129                 printf("%dx^%d",tmp->coef,tmp->exp);
130         }
131         tmp = tmp->next;
132
133         if(tmp!=NULL){
134             if(tmp->coef>0)
135                 printf("+");
136         }
137     }
138     printf("\n");
139 }
140
141
142
143 int deleteItem(list* l,int pos){
144     node *tmp = l->head, *del;
145     int i = 1;
146     if(pos<1 || pos>l->size)
147         return 0;
148     else{
149         if(pos==1){
150             del = l->head;

```

```




151         l->head = del->next;
152         del->next = NULL;
153     }
154     else{
155         while(i<pos-1){
156             tmp = tmp->next;
157             i++;
158         }
159         del = tmp->next;
160         tmp->next = del->next;
161         del->next = NULL;
162         if(del==l->tail)
163             l->tail = tmp;
164     }
165     free(del);
166     l->size--;
167     return 1;
168 }
169 }
170
171
172 node* itemAt(list l, int pos){
173     node *tmp = l.head;
174     int i = 1;
175
176     while(i<pos){
177         tmp = tmp->next;
178         i++;
179     }
180     return tmp;
181 }
182
183 int setItem(list *l, int pos, int x){
184     if(pos<1 || pos > l->size)
185         return 0;
186
187     node *tmp = l->head;
188     int i = 1;
189
190     while(i<pos){
191         tmp = tmp->next;
192         i++;
193     }
194     tmp->coef = x;
195     return 1;
196 }
197
198 void simplify(list *l){
199     int i=1,j;
200
201     //display(*l);
202
203     for(;i<l->size;i++){
204         for(j=i+1;j<=l->size;j++){
205             if(itemAt(*l,i)->exp==itemAt(*l,j)->exp){
206                 setItem(l,i,itemAt(*l,i)->coef + itemAt(*l,j)->coef );
207                 deleteItem(l,j);
208                 j--;
209             }
210         }
211     }
212 }
213
214 list add(list l1, list l2){
215     list sum;
216     init(&sum);
217     int c=1;
218
219     while(c<=l1.size){
220         append(&sum,itemAt(l1,c)->coef,itemAt(l1,c)->exp);
221         c++;
222     }
223
224
225     c=1;

```

```

226     while(c<=l2.size){
227         append(&sum,itemAt(l2,c)->coef,itemAt(l2,c)->exp);
228         c++;
229     }
230
231     //display(sum);
232     simplify(&sum);
233     return sum;
234 }
235
236 list sub(list l1, list l2){
237     list sum;
238     init(&sum);
239     int c=1;
240
241     while(c<=l1.size){
242         append(&sum,itemAt(l1,c)->coef,itemAt(l1,c)->exp);
243         c++;
244     }
245
246     c=1;
247     while(c<=l2.size){
248         append(&sum,(itemAt(l2,c)->coef)*-1,itemAt(l2,c)->exp);
249         c++;
250     }
251
252     //display(sum);
253     simplify(&sum);
254     return sum;
255 }
256
257 void arrange(list *l){
258     int i=1,j;
259     int tmpcoef,tmpexp;
260     //display(*l);
261
262     for(;i<l->size;i++){
263         for(j=i+1;j<=l->size;j++){
264             if(itemAt(*l,i)->exp<=itemAt(*l,j)->exp){
265                 tmpcoef=itemAt(*l,j)->coef;
266                 tmpexp=itemAt(*l,j)->exp;
267                 itemAt(*l,j)->coef=itemAt(*l,i)->coef;
268                 itemAt(*l,j)->exp=itemAt(*l,i)->exp;
269                 itemAt(*l,i)->coef=tmpcoef;
270                 itemAt(*l,i)->exp=tmpexp;
271             }
272         }
273     }
274 }
275 }
276

```

TESTCASE	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	 Success	3	0.0 sec	2.54 MB
Testcase 1	Easy	 Success	5	0.0 sec	2.54 MB
Testcase 2	Easy	 Success	2	0.0 sec	2.3 MB

No Comments