

**exp\_7/a.c**

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Term {
5     int coeff;
6     int exp;
7     struct Term* next;
8 };
9
10 struct Term* createTerm(int c, int e) {
11     struct Term* newTerm = (struct Term*) malloc(sizeof(struct Term));
12     newTerm->coeff = c;
13     newTerm->exp = e;
14     newTerm->next = NULL;
15     return newTerm;
16 }
17
18 void insertTerm(struct Term** poly, int coeff, int exp) {
19     struct Term* newTerm = createTerm(coeff, exp);
20     if (*poly == NULL || (*poly)->exp < exp) {
21         newTerm->next = *poly;
22         *poly = newTerm;
23     } else {
24         struct Term* temp = *poly;
25         while (temp->next != NULL && temp->next->exp > exp)
26             temp = temp->next;
27         if (temp->next != NULL && temp->next->exp == exp) {
28             temp->next->coeff += coeff;
29             free(newTerm);
30             if (temp->next->coeff == 0) {
31                 struct Term* toDelete = temp->next;
32                 temp->next = toDelete->next;
33                 free(toDelete);
34             }
35         } else {
36             newTerm->next = temp->next;
37             temp->next = newTerm;
38         }
39     }
40 }
41
42 void printPoly(struct Term* poly) {
43     if (poly == NULL) {
44         printf("0\n");
45         return;
46     }
47     struct Term* temp = poly;
48     while (temp != NULL) {
```

```

49     if (temp->coeff != 0) {
50         if (temp != poly && temp->coeff > 0)
51             printf(" + ");
52         else if (temp->coeff < 0)
53             printf(" - ");
54         int c = temp->coeff < 0 ? -temp->coeff : temp->coeff;
55         if (temp->exp == 0)
56             printf("%d", c);
57         else if (temp->exp == 1)
58             printf("%dx", c);
59         else
60             printf("%dx^%d", c, temp->exp);
61     }
62     temp = temp->next;
63 }
64 printf("\n");
65 }

66

67 struct Term* addPoly(struct Term* poly1, struct Term* poly2) {
68     struct Term* result = NULL;
69     struct Term* t1 = poly1, *t2 = poly2;
70     while (t1 != NULL && t2 != NULL) {
71         if (t1->exp > t2->exp) {
72             insertTerm(&result, t1->coeff, t1->exp);
73             t1 = t1->next;
74         } else if (t1->exp < t2->exp) {
75             insertTerm(&result, t2->coeff, t2->exp);
76             t2 = t2->next;
77         } else {
78             insertTerm(&result, t1->coeff + t2->coeff, t1->exp);
79             t1 = t1->next;
80             t2 = t2->next;
81         }
82     }
83     while (t1 != NULL) {
84         insertTerm(&result, t1->coeff, t1->exp);
85         t1 = t1->next;
86     }
87     while (t2 != NULL) {
88         insertTerm(&result, t2->coeff, t2->exp);
89         t2 = t2->next;
90     }
91     return result;
92 }

93

94 struct Term* multiplyPoly(struct Term* poly1, struct Term* poly2) {
95     struct Term* result = NULL;
96     for (struct Term* t1 = poly1; t1 != NULL; t1 = t1->next) {
97         for (struct Term* t2 = poly2; t2 != NULL; t2 = t2->next) {
98             insertTerm(&result, t1->coeff * t2->coeff, t1->exp + t2->exp);

```

```
99         }
100     }
101     return result;
102 }
103
104 void freePoly(struct Term* poly) {
105     while (poly != NULL) {
106         struct Term* temp = poly;
107         poly = poly->next;
108         free(temp);
109     }
110 }
111
112 int main() {
113     struct Term *poly1 = NULL, *poly2 = NULL, *sum, *product;
114     int n, coeff, exp;
115
116     printf("Enter number of terms in first polynomial: ");
117     scanf("%d", &n);
118     printf("Enter coefficients and exponents:\n");
119     for (int i = 0; i < n; i++) {
120         scanf("%d %d", &coeff, &exp);
121         insertTerm(&poly1, coeff, exp);
122     }
123
124     printf("Enter number of terms in second polynomial: ");
125     scanf("%d", &n);
126     printf("Enter coefficients and exponents:\n");
127     for (int i = 0; i < n; i++) {
128         scanf("%d %d", &coeff, &exp);
129         insertTerm(&poly2, coeff, exp);
130     }
131
132     printf("First Polynomial: ");
133     printPoly(poly1);
134     printf("Second Polynomial: ");
135     printPoly(poly2);
136
137     sum = addPoly(poly1, poly2);
138     printf("Sum: ");
139     printPoly(sum);
140
141     product = multiplyPoly(poly1, poly2);
142     printf("Product: ");
143     printPoly(product);
144
145     freePoly(poly1);
146     freePoly(poly2);
147     freePoly(sum);
148     freePoly(product);
```

```
149 |
150     return 0;
151 }
152 |
```