

```

AddPolynomials.c
Saved

1 #include<stdio.h>
2 struct Node
3 {
4     int coeff;
5     int pow;
6     struct Node* next;
7 };
8
9 void readPolynomial(struct Node** poly)
10 {
11     int coeff, exp, cont;
12     struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
13     *poly = temp;
14     do{
15         printf("Coefficient: ");
16         scanf("%d", &coeff);
17         printf("Exponent: ");
18         scanf("%d", &exp);
19         temp->coeff = coeff;
20         temp->pow = exp;
21         temp->next = NULL;
22         printf("Have more terms? 1 for yes and 0 for no : ");
23         scanf("%d", &cont);
24         if(cont)
25         {
26             temp->next = (struct Node*)malloc(sizeof(struct Node));
27             temp = temp->next;
28             temp->next = NULL;
29         }
30     }while(cont);
31 }
32
33 void displayPolynomial(struct Node* poly)
34 {
35     printf("\nPolynomial Expression is: ");
36     while(poly != NULL)
37     {
38         printf("%dX^%d", poly->coeff, poly->pow);
39         poly = poly->next;
40         if(poly != NULL)
41             printf("+");
42     }
43 }
44 void displayResult(struct Node* poly)
45 {
46     printf("\nPolynomial after Addition is: ");
47     while(poly != NULL)
48     {
49         printf("%dX^%d", poly->coeff, poly->pow);
50         poly = poly->next;
51         if(poly != NULL)
52             printf("+");
53     }
54 }
55 void addPolynomials(struct Node** result, struct Node* first, struct Node* second)
56 {
57     struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
58     temp->next = NULL;
59     *result = temp;
60     while(first && second)
61     {
62         if(first->pow > second->pow)
63         {
64             temp->coeff = first->coeff;
65             temp->pow = first->pow;
66             first = first->next;
67         }
68         else if(first->pow < second->pow)
69         {
70             temp->coeff = second->coeff;
71             temp->pow = second->pow;
72             second = second->next;
73         }
74         else
75         {
76             temp->coeff = first->coeff + second->coeff;
77             temp->pow = first->pow;
78             first = first->next;
79             second = second->next;
80         }
81
82         if(first && second)
83         {
84             temp->next = (struct Node*)malloc(sizeof(struct Node));
85             temp = temp->next;
86             temp->next = NULL;
87         }
88     }
89     while(first || second)
90     {
91         temp->next = (struct Node*)malloc(sizeof(struct Node));
92         temp = temp->next;
93         temp->next = NULL;
94
95         if(first)
96         {
97             temp->coeff = first->coeff;
98             temp->pow = first->pow;
99             first = first->next;
100         }
101
102         else if(second)
103         {
104             temp->coeff = second->coeff;
105             temp->pow = second->pow;
106             second = second->next;
107         }
108     }
109 }
110
111 int main()
112 {
113     struct Node* first = NULL;
114     struct Node* second = NULL;
115     struct Node* result = NULL;
116     printf("First polynomial:\n\n");
117     readPolynomial(&first);
118     displayPolynomial(first);
119     printf("\nSecond polynomial:\n\n");
120     readPolynomial(&second);
121     displayPolynomial(second);
122     addPolynomials(&result, first, second);
123     displayResult(result);
124     return 0;
125 }

```



First polynomial:

Coeffecient: 2

Exponent: 2

Have more terms? 1 for yes and 0 for no : 1

Coeffecient: 4

Exponent: 1

Have more terms? 1 for yes and 0 for no : 1

Coeffecient: 5

Exponent: 0

Have more terms? 1 for yes and 0 for no : 0

Polynomial Expression is: $2X^2+4X^1+5X^0$

Second polynomial:

Coeffecient: 3

Exponent: 2

Have more terms? 1 for yes and 0 for no : 1

Coeffecient: 6

Exponent: 1

Have more terms? 1 for yes and 0 for no : 0

Polynomial Expression is: $3X^2+6X^1$

Polynomial after Addition is: $5X^2+10X^1+5X^0$

Process finished.

```

AddLongNumbers.c
Saved
1 #include<stdio.h>
2 struct NODE
3 {
4     int info;
5     struct NODE*link;
6 };
7 typedef struct NODE*node;
8 node getnode()
9 {
10     node x;
11     x=(node)malloc(sizeof(struct NODE));
12     if(x==NULL)
13     {
14         printf("out of mamory\n");
15         exit(0);
16     }
17     return x;
18 }
19 node ins_front(node first,int item)
20 {
21     node temp;
22     temp=getnode();
23     temp->info=item;
24     temp->link=first;
25     return temp;
26 }
27 node extract(char *s,node head)
28 {
29     int i,n;
30     for(i=0;i<strlen(s);i++)
31     {
32         n=s[i]-'0';
33         head=ins_front(head,n);
34     }
35     return head;
36 }
37 node addlong(node head1,node head2,node head3)
38 {
39     int temp,sum,carry=0;
40     node cur1,cur2;
41     cur1=head1;
42     cur2=head2;
43     while(cur1!=NULL&&cur2!=NULL)
44     {
45         temp=cur1->info+cur2->info+carry;
46         if(temp>9)
47         {
48             sum=temp%10;
49             carry=temp/10;
50         }
51         else
52         {
53             sum=temp;
54             carry=0;
55         }
56         head3=ins_front(head3,sum);
57         cur1=cur1->link;
58         cur2=cur2->link;
59     }
60     while(cur1!=NULL)
61     {
62         temp=cur1->info+carry;
63         if(temp>9)
64         {
65             sum=temp%10;
66             carry=temp/10;
67         }
68         else
69         {
70             sum=temp;
71             carry=0;
72         }
73         head3=ins_front(head3,sum);
74         cur1=cur1->link;
75     }
76     while(cur2!=NULL)
77     {
78         temp=cur2->info+carry;
79         if(temp>9)
80         {
81             sum=temp%10;
82             carry=temp/10;
83         }
84         else
85         {
86             sum=temp;
87             carry=0;
88         }
89         head3=ins_front(head3,sum);
90         cur2=cur2->link;
91     }
92     if(cur1==NULL&&cur2==NULL)
93     {
94         if(carry==1)
95             head3=ins_front(head3,carry);
96     }
97     return head3;
98 }
99 void printReverse(node head)
100 {
101     if (head == NULL)
102         return;
103     printReverse(head->link);
104     printf("%d ", head->info);
105 }
106 void display(node first)
107 {
108     node cur;
109     if(first==NULL)
110     {
111         printf("Empty\n");
112         return;
113     }
114     cur=first;
115     while(cur!=NULL)
116     {
117         printf("%d ",cur->info);
118         cur=cur->link;
119     }
120 }
121 void main()
122 {
123     int ch;
124     node head1=NULL;
125     node head2=NULL;
126     node head3=NULL;
127     char s1[30],s2[30];
128     printf("\nEnter First Integer :");
129     scanf("%s",s1);
130     head1=extract(s1,head1);
131     printReverse(head1);
132     printf("\nEnter Second Integer :");
133     scanf("%s",s2);
134     head2=extract(s2,head2);
135     printReverse(head2);
136     head3=addlong(head1,head2,head3);
137     printf("\nThe result is\n");
138     display(head3);
139 }

```

× Terminal



```
Enter First Integer :42579135789
4 2 5 7 9 1 3 5 7 8 9
Enter Second Integer :2568368025679
2 5 6 8 3 6 8 0 2 5 6 7 9
The result is
2 6 1 0 9 4 7 1 6 1 4 6 8
Process finished.
```