AddPolynomials.c

Saved

```c
#include<stdio.h>
struct Node
{
  int coeff;
  int pow;
  struct Node* next;
};

void readPolynomial(struct Node** poly)
{
  int coeff, exp, cont;
  struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
  *poly = temp;
  do{
    printf("Coeffecient: ");
    scanf("%d", &coeff);
    printf("Exponent: ");
    scanf("%d", &exp);
    temp->coeff = coeff;
    temp->pow = exp;
    temp-> next = NULL;
    printf("Have more terms? 1 for yes and 0 for no : ");
    scanf("%d", &cont);
    if(cont)
    {
      temp->next = (struct Node*)malloc(sizeof(struct Node));
      temp = temp->next;
      temp->next = NULL;
    }
  }while(cont);
}

void displayPolynomial(struct Node* poly)
{
  printf("\nPolynomial Expression  is: ");
  while(poly != NULL)
  {
    printf("%dX^%d", poly->coeff, poly->pow);
    poly = poly->next;
    if(poly != NULL)
      printf("+");
  }
}
void displayResult(struct Node* poly)
{
  printf("\nPolynomial after Addition is: ");
  while(poly != NULL)
  {
    printf("%dX^%d", poly->coeff, poly->pow);
    poly = poly->next;
    if(poly != NULL)
      printf("+");
  }
}
void addPolynomials(struct Node** result, struct Node* first, struct Node* second)
{
   struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
   temp->next = NULL;
   *result = temp;
   while(first && second)
   {
     if(first->pow > second->pow)
     {
       temp->coeff = first->coeff;
       temp->pow = first->pow;
       first = first->next;
     }
     else if(first->pow < second->pow)
     {
       temp->coeff = second->coeff;
       temp->pow = second->pow;
       second = second->next;
     }
     else
     {
       temp->coeff = first->coeff + second->coeff;
       temp->pow = first->pow;
       first = first->next;
       second = second->next;
     }

     if(first && second)
     {
       temp->next = (struct Node*)malloc(sizeof(struct Node));
       temp = temp->next;
       temp->next = NULL;
     }
   }
   while(first || second)
   {
     temp->next = (struct Node*)malloc(sizeof(struct Node));
     temp = temp->next;
     temp->next = NULL;

     if(first)
     {
       temp->coeff = first->coeff;
       temp->pow = first->pow;
       first = first->next;
     }

     else if(second)
     {
       temp->coeff = second->coeff;
       temp->pow = second->pow;
       second = second->next;
     }
   }
}

int main()
{
  struct Node* first = NULL;
  struct Node* second = NULL;
  struct Node* result = NULL;
  printf("First polynomial:\n\n");
  readPolynomial(&first);
  displayPolynomial(first);
  printf("\nSecond polynomial:\n\n");
  readPolynomial(&second);
  displayPolynomial(second);
  addPolynomials(&result, first, second);
  displayResult(result);
  return 0;
}
```

```
Terminal
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

```c
#include<stdio.h>
struct NODE
{
    int info;
    struct NODE*link;
};
typedef struct NODE*node;
node getnode()
{
    node x;
    x=(node)malloc(sizeof(struct NODE));
    if(x==NULL)
    {
        printf("out of mamory\n");
        exit(0);
    }
    return x;
}
node ins_front(node first,int item)
{
    node temp;
    temp=getnode();
    temp->info=item;
    temp->link=first;
    return temp;
}
node extract(char *s,node head)
{
  int i,n;
  for(i=0;i<strlen(s);i++)
  {
  n=s[i]-'0';
  head=ins_front(head,n);
  }
  return head;
}
node addlong(node head1,node head2,node head3)
{
  int temp,sum,carry=0;
  node cur1,cur2;
  cur1=head1;
  cur2=head2;
  while(cur1!=NULL&&cur2!=NULL)
  {
    temp=cur1->info+cur2->info+carry;
    if(temp>9)
    {
        sum=temp%10;
        carry=temp/10;
    }
    else
    {
        sum=temp;
        carry=0;
    }
    head3=ins_front(head3,sum);
    cur1=cur1->link;
    cur2=cur2->link;
  }
  while(cur1!=NULL)
  {
      temp=cur1->info+carry;
      if(temp>9)
      {
          sum=temp%10;
          carry=temp/10;
      }
      else
      {
          sum=temp;
          carry=0;
      }
      head3=ins_front(head3,sum);
      cur1=cur1->link;
  }
    while(cur2!=NULL)
  {
      temp=cur2->info+carry;
      if(temp>9)
      {
          sum=temp%10;
          carry=temp/10;
      }
      else
      {
          sum=temp;
          carry=0;
      }
      head3=ins_front(head3,sum);
      cur2=cur2->link;
  }
  if(cur1==NULL&&cur2==NULL)
  {
    if(carry==1)
        head3=ins_front(head3,carry);
  }
  return head3;
}
void printReverse(node head)
{
    if (head == NULL)
       return;
    printReverse(head->link);
    printf("%d  ", head->info);
}
void display(node first)
{
    node cur;
    if(first==NULL)
    {
        printf("Empty\n");
        return;
    }
    cur=first;
    while(cur!=NULL)
    {
        printf("%d ",cur->info);
        cur=cur->link;
    }
}
void main()
{
  int ch;
  node head1=NULL;
  node head2=NULL;
  node head3=NULL;
  char s1[30],s2[30];
  printf("\nEnter First Integer :");
  scanf("%s",s1);
  head1=extract(s1,head1);
  printReverse(head1);
  printf("\nEnter Second Integer :");
  scanf("%s",s2);
  head2=extract(s2,head2);
  printReverse(head2);
  head3=addlong(head1,head2,head3);
  printf("\nThe result is\n");
  display(head3);
}
```