NAME: Venkateswar L
BRANCH: Computer Science and Engineering SEC: F
ROLL NO.: 230701376

PROGRAM: Polynomial Manipulation

Write a C program to implement the following operations on Singly Linked List.

1. Polynomial Addition
2. Polynomial Subtraction
3. Polynomial Multiplication

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
   int coef;
   int power;
   struct node*link;
};
typedef struct node NODE;

void create_poly(NODE *list)
{
   int coef;
   int power;
   int choice;
   NODE *newnode;
   do{
   newnode=malloc(sizeof(NODE));
   printf("Enter the coefficient : ");
```

```
    scanf("%d", &coef);

    printf("Enter the power : ");
    scanf("%d", &power);

    newnode->coef=coef;
    newnode->power=power;
    newnode->link=NULL;

    if(list->link==NULL)
    {
       list->link=newnode;
    }
    else
    {
       while(list->link!=NULL)
       {
          list=list->link;
       }
       list->link=newnode;
    }
    printf("Enter 1 to continue : ");
    scanf("%d", &choice);
   }
   while(choice==1);
}
```

NAME: Venkateswar L
BRANCH: Computer Science and Engineering SEC: F
ROLL NO.: 230701376

```
void add(NODE *list1,NODE *list2,NODE *Result)
{
    NODE *newnode,*temp=Result;
   while(list1!=NULL && list2!=NULL)
 {

    newnode=malloc(sizeof(NODE));
    if(list1->power == list2->power)
    {
      newnode->coef = list1->coef+list2->coef;
      newnode->power =list1->power;
      newnode->link=NULL;
      list1=list1->link;
      list2=list2->link;

    }
    else if(list1->power > list2->power)
    {
      newnode->coef=list1->coef;
      newnode->power=list1->power;
      newnode->link=NULL;
      list1 = list1->link;
    }
    else if(list1->power<list2->power)
    {
       newnode->coef=list2->coef;
```

```
        newnode->power=list2->power;
        newnode->link=NULL;
        list2=list2->link;


    }
     temp->link=newnode;
     temp=temp->link;


}


 while(list2!=NULL || list2!=NULL)
 {
   newnode = malloc(sizeof(NODE));
   if(list1->link!=NULL)
   {
     newnode->coef=list1->coef;
     newnode->power=list1->power;
     newnode->link= NULL;
     list1=list1->link;
   }
  if(list2->link!= NULL)
  {
     newnode->coef=list2->coef;
     newnode->power=list2->power;
     newnode->link= NULL;
     list2 = list2->link;
   }
```

```
    temp->link=newnode;
    temp=temp->link;
 }
}
```

```
void sub(NODE *list1,NODE *list2,NODE *Result)
{
    NODE *newnode,*temp=Result;
    while(list1!=NULL && list2!=NULL)
    {
       newnode=malloc(sizeof(NODE));
       if(list1->power==list2->power)
       {
         newnode->coef=list1->coef-list2->coef;
          newnode->power=list1->power;
          list1=list1->link;
          list2=list2->link;
       }
      else if(list1->power>list2->power)
      {
        newnode->coef=list1->coef;
        newnode->power=list1->power;
        list1=list1->link;
      }
      else if(list1->power<list2->power)
```

```
        {
          newnode->coef= -(list2->coef);
          newnode->power=list2->power;
          list2=list2->link;
        }
        newnode->link= NULL;
        temp->link=newnode;
        temp=temp->link;
      }

      while(list1!=NULL || list2!= NULL)
      {
      newnode = malloc(sizeof(NODE));
      if(list1!= NULL)
      {
       newnode->coef=list1->coef;
       newnode->power=list1->power;
       list1 = list1->link;
      }
      if(list2 != NULL)
      {
       newnode->coef= -(list2->coef);
       newnode->power=list2->power;
       list2 = list2->link;
      }
      newnode->link= NULL;
      temp->link=newnode;
```

```
    temp=temp->link;
  }
}

void multi(NODE *list1, NODE *list2, NODE *Result)
{
   NODE *newnode;
   NODE *t1=list1->link;
   NODE *t2=list2->link;
   NODE *t3=Result;

   while(t1!=NULL)
   {
      t2=list2->link;
      while(t2!=NULL)
      {
         newnode=(NODE*)malloc(sizeof(NODE));
         t3->link=newnode;
         newnode->coef=t1->coef*t2->coef;
         newnode->power=t1->power+t2->power;
         t2=t2->link;
         newnode->link=NULL;
         t3=t3->link;
      }
      t1=t1->link;
   }
}
```

NAME: Venkateswar L
BRANCH: Computer Science and Engineering SEC: F
ROLL NO.: 230701376

```
void display(NODE *list)
{
   NODE *temp=list->link;
   while(temp!=NULL)
   {
      printf("%dX^%d",temp->coef,temp->power);
       temp=temp->link;
       if(temp != NULL && temp->coef >= 0)
        {
          printf("+");
         }
   }
}

int main(){
   int t=1,choice;
   NODE *Poly1 = malloc(sizeof(NODE));
    NODE *Poly2 = malloc(sizeof(NODE));
    NODE *Result = malloc(sizeof(NODE));
   while (t==1){
       Poly1->link=NULL;
       Poly2->link=NULL;
       printf("\nMENU\n");
       printf("1.Add the polynomials\n2.Subtract the polynomials\n3.Multiply the
polynomials\n4.EXIT\n");
       printf("\nEnter your choice:");
```

```
    scanf("%d",&choice);
    if (choice!=4){
       printf("Enter the values for first polynomial :\n");
       create_poly(Poly1);
       printf("The polynomial equation is : ");
       display(Poly1);
       printf("\nEnter the values for second polynomial :\n");
       create_poly(Poly2);
       printf("The polynomial equation is : ");
       display(Poly2);
    }
    switch (choice)
    {
       case 1:
       add(Poly1, Poly2, Result);
       printf("\nThe polynomial equation addition result is : ");
       display(Result->link);
       break;
       case 2:
       sub(Poly1, Poly2, Result);
       printf("\nThe polynomial equation addition result is : ");
       display(Result->link);
       break;
       case 3:
       multi(Poly1, Poly2, Result);
       printf("\nThe polynomial equation addition result is : ");
       display(Result);
```

```
            break;
            case 4:
            t=0;
            break;
        }
    }
}
```