```
/* Program to add two polynomials using a linked list */
#include <stdio.h>
#include <stdlib.h>
#define NULL 0

typedef struct poly{
int coef;
int exp;
struct poly *next;
}p;

p *p1, *p2, *ptr3, *p3, *new_node;
char ans;
int sum_coef=0;
void create_poly(p *);
void add_poly();
void display_poly(p *);
p *add_node(p *,int, int);

int main()
{
   p1 = (p *)malloc(sizeof(p));
   p2 = (p *)malloc(sizeof(p));

   printf("\nCreate the first polynomial:");
   create_poly(p1);

   printf("\nCreate the second polynomial:");
   create_poly(p2);

   printf("\nDisplay the first polynomial:");
   display_poly(p1);

   printf("\nDisplay the second polynomial:");
   display_poly(p2);

   printf("\nAdd the polynomial:");
   add_poly(p1,p2);

   printf("\nDisplay the third polynomial:");
   display_poly(p3);

   return 0;
```

```
}

void create_poly(p *ptr)
{
   printf("\nEnter the coeff. & expo. of the node:");
   scanf("%d %d",&ptr->coef,&ptr->exp);
   ptr->next=NULL;

   printf("\n Do you want to create more symbols?:");
   scanf(" %c",&ans);

   if(ans=='y' || ans=='Y')
    {
       ptr->next=(p *)malloc(sizeof(p));
       ptr=ptr->next;
       create_poly(ptr);
    }

}

void display_poly(p *ptr)
{
   if(ptr != NULL)
   {
      printf("%dx^%d +",ptr->coef,ptr->exp);
      ptr=ptr->next;
      display_poly(ptr);
   }
}

void add_poly(p *ptr1, p *ptr2)
{
   if(ptr1 != NULL && ptr2 == NULL)
   {
      p3 = add_node(p3, ptr1->coef, ptr1->exp);
      ptr1 = ptr1->next;
      add_poly(ptr1,ptr2);
   }

   else if(ptr1 == NULL && ptr2 != NULL)
   {
      p3 = add_node(p3, ptr2->coef,ptr2->exp);
      ptr2=ptr2->next;
```

```
      add_poly(ptr1,ptr2);
   }

   else if(ptr1 != NULL && ptr2 != NULL)
   {
      if(ptr1->exp > ptr2->exp)
      {
         p3 = add_node(p3, ptr1->coef, ptr1->exp);
         ptr1 = ptr1->next;
      }
      else if(ptr1->exp < ptr2->exp)
      {
         p3 = add_node(p3, ptr2->coef, ptr2->exp);
         ptr2 = ptr2->next;
      }
      else if(ptr1->exp == ptr2->exp)
      {
         sum_coef = ptr1->coef + ptr2->coef;
         p3 = add_node(p3, sum_coef, ptr1->exp);
         ptr1 = ptr1->next;
         ptr2 = ptr2->next;
      }

      add_poly(ptr1,ptr2);
   }
}

p *add_node(p *ptr,int c, int e)
{
   if(ptr == NULL)
   {
      new_node = (p *)malloc(sizeof(p));
      new_node->coef = c;
      new_node->exp = e;
      new_node->next = NULL;
      ptr=new_node;
   }
   else
   {
    p *ptr3=ptr;
    while(ptr3->next != NULL)
      ptr3=ptr3->next;
```

```
      new_node = (p *)malloc(sizeof(p));
      new_node->coef = c;
      new_node->exp = e;
      new_node->next = NULL;
      ptr3->next=new_node;
    }
   return(ptr);

}
```