```
/**PROGRAM TO ADD TWO POLYNOMIALS**/
#include<stdio.h>
#include<conio.h>
 struct barbie
   {
         int coff;
         int pow;
         struct barbie *link;
        }*ptr,*start1,*node,*start2,*start3,*ptr1,*ptr2;
        typedef struct barbie bar;
        int temp1,temp2;

   void main()
         {

          void create(void);
          void display(void);
          void polyaddtion(void);
          void sorting(void);
          clrscr();

          printf("\n\nEnrter the elements of the first poly");
          node = (bar *) malloc(sizeof (bar));
          start1=node;
          if (start1==NULL)
                 {
                   printf("\n\nUnable to create memory.");
                   getch();
                   exit();
                  }
          create();

          printf("\n\nEnrter the elements of the second poly :");
          node = (bar *) malloc(sizeof (bar));
          start2=node;
          if (start2==NULL)
                 {
                   printf("\n\nUnable to create memory.");
                   getch();
                   exit();
                  }
          create();
          clrscr();
          //printing the elements of the lists
          printf("\n\nThe elements of the poly first are :");
          ptr=start1;
          display();

          printf("\n\nThe elements of the poly second are :");
          ptr=start2;
          display();

          printf("\n\nThe first sorted list is :");
          ptr=start1;
          sorting();
          ptr=start1;
          display();

          printf("\n\nThe second sorted list is :");
          ptr=start2;
          sorting();
          ptr=start2;
          display();

          printf("\n\nThe sum of the two lists are :");
          polyaddtion();
          ptr=start3;
          display();

          getch();

         }

/*--------------------------------------------------------------------------*/
         void create()
          {
                 char ch;
                 while(1)
                  {
                     printf("\n\nEnter the coff and pow :");
                     scanf("%d%d",&node->coff,&node->pow);
                     if (node->pow==0 )
                           {
                              ptr=node;
                              node=(bar *)malloc(sizeof(bar));
```

```
                                node=NULL;
                                ptr->link=node;
                                break;
                           }

                   printf("\n\nDo u want enter more coff ?(y/n)");
                   fflush(stdin);
                   scanf("%c",&ch);
                          if (ch=='n' )
                           {
                                ptr=node;
                                node=(bar *)malloc(sizeof(bar));
                                node=NULL;
                                ptr->link=node;
                                break;
                           }
                     ptr=node;
                     node=(bar *)malloc(sizeof(bar));
                     ptr->link=node;
                   }
             }
/*------------------------------------------------------------------------*/
        void display()
             {  int i=1;

               while(ptr!=NULL )
                    {
                          if(i!=1)
                          printf("+ ");
                          printf(" %dx^%d ",ptr->coff,ptr->pow);
                          ptr=ptr->link;
                          i++;
                    }
                   //printf(" %d^%d",ptr->coff,ptr->pow);
              }
/*--------------------------------------------------------------------------*/
      void sorting()
       {
             for(;ptr->coff!=NULL;ptr=ptr->link)
                     for(ptr2=ptr->link;ptr2->coff!=NULL;ptr2=ptr2->link)
                          {
                           if(ptr->pow>ptr2->pow)
                                {
```

```
                             temp1=ptr->coff;
                             temp2=ptr->pow;
                             ptr->coff=ptr2->coff;
                             ptr->pow=ptr2->pow;
                             ptr2->coff=temp1;
                             ptr2->pow=temp2;

                            }
                       }
       }
/*--------------------------------------------------------------------------------------*/
        void polyaddtion()
            {
             node=(bar *)malloc (sizeof(bar));
             start3=node;

             ptr1=start1;
             ptr2=start2;

             while(ptr1!=NULL && ptr2!=NULL)
             {
                  ptr=node;
                  if  (ptr1->pow > ptr2->pow )
                      {
                       node->coff=ptr2->coff;
                       node->pow=ptr2->pow;
                       ptr2=ptr2->link;   //update ptr list B
                       }
                   else if ( ptr1->pow < ptr2->pow )
                      {
                       node->coff=ptr1->coff;
                       node->pow=ptr1->pow;
                       ptr1=ptr1->link;   //update ptr list A
                       }
                   else
                       {
                       node->coff=ptr2->coff+ptr1->coff;
                       node->pow=ptr2->pow;
                       ptr1=ptr1->link;   //update ptr list A
                       ptr2=ptr2->link;   //update ptr list B

                       }
```

```
                    node=(bar *)malloc (sizeof(bar));
                    ptr->link=node;   //update ptr list C
                }//end of while

            if (ptr1==NULL)     //end of list  A
                {
                    while(ptr2!=NULL)
                     {
                          node->coff=ptr2->coff;
                          node->pow=ptr2->pow;
                          ptr2=ptr2->link;   //update ptr list B
                          ptr=node;
                          node=(bar *)malloc (sizeof(bar));
                          ptr->link=node;   //update ptr list C
                     }
                }

               else if (ptr2==NULL)     //end of list  B
                {
                    while(ptr1!=NULL)
                     {
                          node->coff=ptr1->coff;
                          node->pow=ptr1->pow;
                          ptr1=ptr1->link;   //update ptr list B
                          ptr=node;
                          node=(bar *)malloc (sizeof(bar));
                          ptr->link=node;   //update ptr list C
                     }
                }
           node=NULL;
           ptr->link=node;
          }


-->>SAMPLE INPUT AND OUTPUT:

 Enter the element of the first poly

Enter the coff and pow : 3  2

Do u want enter more coff ?(y/n)y

Enter the coff and pow :5  1
```

```
Do u want enter more coff ?(y/n)n

Enter the elements of the second poly:

Enter the coff and pow :4  2

Do u want enter more coff ?(y/n)y

Enter the coff and pow :4  2

The elements of the poly first are :  3x^2 + 5x^1
The elements of the poly first are :  4x^2 + 4x^1
The first sorted list is  : 5X^1 + 3X^2
The sacond sorted list is : 4X^1 + 4x^2
The sum of the two lists are : 9X^1 + 7X^2
```