ACADEMY OF TECHNOLOGY

NAME : PRIYADARSHAN GHOSH

COLLEGE ROLL NO: 72

UNIVERSITY ROLL NO: 16900319072

DEPARTMENT: ECE-1(Y)

SEMESTER: 3rd

PAPER CODE : ES-CS391

➢ Laboratory *Assignment #12*

Q1. REPRESENTATION OF POLYNOMIAL USING SINGLE LINKED LIST AND DISPLAY IT.

ADDITION OF TWO POLYNOMIALS USING LINKED LIST.

MULTIPLICATION OF TWO POLYNOMIALS USING LINKED LIST.

Ans:

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

struct node

{
int coef;

 int exp;

 struct node *link;  };

       void insert_term(struct node **,int,int);

       void traverse(struct node *);

void poly_add(struct node **,struct node **,struct node **);

void poly_pdt(struct node **,struct node **,struct node **);
```

```
int main()

{

struct node *start_p=NULL,*start_q=NULL,*start_r=NULL;

        int i,n,c,e;

                printf("Enter first polynomial!\n");

                printf("Enter no of terms:");

                scanf("%d",&n);

for(i=0;i<n;i++)

{
printf("Enter coefficient: ");

scanf("%d",&c);

printf("Enter exponent: ");

scanf("%d",&e);

insert_term(&start_p,c,e);
```

```
}
        printf("Enter second polynomial!\n");
            printf("Enter no of terms:");
scanf("%d",&n);
for(i=0;i<n;i++)
{
        printf("Enter coefficient: ");
        scanf("%d",&c);
        printf("Enter exponent: ");
        scanf("%d",&e);
    insert_term(&start_q,c,e);
}
        printf("First polynomial: ");
        traverse(start_p);
```

```
            printf("Second polynomial: ");

            traverse(start_q);

            poly_add(&start_p,&start_q,&start_r);

            printf("Sum of two polynomial:\n");

            traverse(start_r);

start_r=NULL;

poly_pdt(&start_p,&start_q,&start_r);

printf("Multiplication of the two polynomial:\n");

traverse(start_r);

getch();

}

void insert_term(struct node **start,int c,int e)

{  struct node *temp,*temp1,*prev;

 if (*start==NULL)
```

```
 {
temp=(struct node*)malloc(sizeof(struct node));
if (temp==NULL)
printf("Node is not created, Term cannot be inserted\n");
else
{ temp->coef=c;
   temp->exp=e;
   temp->link=NULL;
   *start=temp;
  }
  }
else
{ temp1=*start;
  while (temp1!=NULL && temp1->exp>e)
```

```
{
 prev=temp1;
 temp1=temp1->link;
}
if(temp1==NULL)
{
 temp=(struct node *)malloc(sizeof(struct node));
 if (temp==NULL)
  printf("Node is not created\n");
 else
 { temp->coef=c;
  temp->exp=e;
  temp->link=NULL;
```

```
  prev->link=temp;

 }

}

else

{ if(temp1->exp==e)

  temp1->coef=temp1->coef+c;

 else

 { if(temp1==*start)

  {  temp=(struct node *)malloc (sizeof (struct node));

   if(temp==NULL)

    printf("Node is not created\n");

   else

   { temp->coef=c;

    temp->exp=e;
```

```
 temp->link=*start;

 *start=temp;

}
    }

else

{ temp=(struct node *)malloc(sizeof(struct node));

 if (temp==NULL)

  printf("Node is not created\n");

 else

 { temp->coef=c;

  temp->exp=e;

  temp->link=temp1;

  prev->link=temp;

 }
```

```
            }
                  }


 }
      }
            }
void traverse(struct node *start)

{ struct node *temp;

 temp=start;

 if (temp==NULL)

  printf("Empty polynomial\n");

 else

 { while(temp!=NULL)

  { printf("%d x^%d",temp->coef,temp->exp);

   temp=temp->link;

   if(temp!=NULL)
```

```
   printf(" + ");

   else

   printf("\n");

  }
       }
         }

void poly_add(struct node** start_p,struct node **start_q,struct
node** start_r)

{ int c,e;

 struct node *pptr,*qptr;

 *start_r=NULL;

 pptr=*start_p;

 qptr=*start_q;

 while(pptr!=NULL && qptr!=NULL)

 {  if (pptr->exp==qptr->exp)

  {
```

```
 c=pptr->coef+qptr->coef;

 e=pptr->exp;

 insert_term(start_r,c,e);

 pptr=pptr->link;

 qptr=qptr->link;

}

else

{ if (pptr->exp>qptr->exp)

 {  c=pptr->coef;

  e=pptr->exp;

  insert_term(start_r,c,e);

  pptr=pptr->link;

 }
```

```
else
  { c=qptr->coef;
    e=qptr->exp;
    insert_term(start_r,c,e);
    qptr=qptr->link;
  }
      }
                  }
while(pptr!=NULL)
{
 c=pptr->coef;
 e=pptr->exp;
 insert_term(start_r,c,e);
 pptr=pptr->link;
}
```

```
while (qptr!=NULL)

{

 c=qptr->coef;

 e=qptr->exp;

 insert_term(start_r,c,e);

 qptr=qptr->link;

}
      }

void poly_pdt(struct node ** start_p,struct node **start_q,struct
node** start_r)

{

 int c,e;

 struct node *pptr,*qptr;

 *start_r=NULL;

 pptr=*start_p;
```

```
 qptr=*start_q;
 if (*start_p==NULL && *start_q==NULL)
  printf("\nMultiplication of polynomial is not possible!\n");
else
 { while(pptr!=NULL)
  {
   qptr=*start_q;
   while(qptr!=NULL)
   {
    c=pptr->coef*qptr->coef;
    e=pptr->exp+qptr->exp;
    insert_term(start_r,c,e);
    qptr=qptr->link;
```

```
    }

    pptr=pptr->link;

    }

    }
}
```

OUTPUT =>

```
Enter first polynomial!
Enter no of terms:3
Enter coefficient: 2
Enter exponent: 3
Enter coefficient: 4
Enter exponent: 2
Enter coefficient: 6
Enter exponent: 3
Enter second polynomial!
Enter no of terms:3
Enter coefficient: 2
Enter exponent: 6
Enter coefficient: 9
Enter exponent: 4
Enter coefficient: 3
Enter exponent: 2
First polynomial: 8 x^3 + 4 x^2
Second polynomial: 2 x^6 + 9 x^4 + 3 x^2
```

Sum of two polynomial:

2 x^6 + 9 x^4 + 8 x^3 + 7 x^2

Multiplication of the two polynomial:

16 x^9 + 8 x^8 + 72 x^7 + 36 x^6 + 24 x^5 + 12 x^4

Process exited after 28.86 seconds with return value 0

Press any key to continue . . .