ADDITION AND MULTIPLICATION OF TWO POLYNOMIALS:-

Name:- Brendan Lucas
Roll No:- 8953
Div:- SE Comps B
Source Code:-

```
#include<stdio.h>
#include<stdlib.h>
#include <conio.h>
typedef struct node
{
	int pow;
	int coeff;
	struct node *next;
}node;


typedef struct
{
	node *start;
}LL;

void display(LL *ll)
{
	node *p;
	int i=0;
	if(ll->start==NULL)
	{
		printf("Queue is empty\n");
		return;
	}
	p=ll->start;
	if(p->coeff!=0)
	{
		printf("%dx^(%d)",p->coeff,p->pow);
	}
	 p=p->next;
	while(p!=NULL)
	{
	if(p->coeff==0)
	{
		p=p->next;
	}
	else
		{
		if(p->coeff>0)
```

```
	{
		printf("+");
	}
	if(p->pow==0)
	{
		printf("%d",p->coeff);
	 p=p->next;
	}
	else
	{
		printf("%dx^(%d)",p->coeff,p->pow);
		p=p->next;
	}
		}
	}
	printf("\n");
	return;
}


void createlist(LL *ll)
{
	node *p,*q;
	int i,k,c;
	printf("Enter the degree of equation:- ");
	scanf("%d",&k);
	for(i=k;i>=0;i--)
	{
		p=(node *)malloc(sizeof(node));
		if(i!=0)
		{
			printf("Enter the coefficient of power %d term:- ",i);
		}
		else
		{
			printf("Enter the constant term:-		");
		}

		scanf("%d",&c);
		p->pow=i;
		p->coeff=c;
		p->next=NULL;
		if(ll->start==NULL)
		{
			 ll->start=p;
		}
		else
		{
```

```
                q=ll->start;
                        while(q ->next!=NULL)
                        {
                                q=q->next;
                        }
                q->next=p;
                }
        }
        printf("\n\n");
}

void removeduplicate(LL *prodll)
{
        node *p1,*p2,*dup;
        p1=prodll->start;
        p2=p1;
        while(p1!=NULL&&p1->next!=NULL)
        {
                p2=p1;
                while(p2->next!=NULL)
                {
                        if(p1->pow==(p2->next)->pow)
                        {
                                p1->coeff=p1->coeff+(p2->next)->coeff;
                                dup=p2->next;
                                p2->next=(p2->next)->next;
                                free(dup);
                                p2=p2->next;
                        }
                        else
                        {
                                p2=p2->next;
                        }
                }
                p1=p1->next;
        }
}

void add(LL *ll1, LL *ll2,LL *sumll)
{
        node *p1,*p2,*sp,*p,*q;
        p1=ll1->start;
        p2=ll2->start;
        sp=sumll->start;
        while(p1!=NULL&&p2!=NULL)
        {
                p=(node*)malloc(sizeof(node ));
                p->next=NULL;
```

```
                if(p1->pow==p2->pow)
                {
                        p->pow=p1->pow;
                        p->coeff=p1->coeff+p2->coeff;
                        p1=p1->next;
                        p2=p2->next;
                }
                else if(p1->pow>p2->pow)
                {
                        p->pow=p1->pow;
                        p->coeff=p1->coeff;
                        p1=p1->next;
                }
                else if(p2->pow>p1->pow)
                {
                        p->pow=p2->pow;
                        p->coeff=p2->coeff;
                        p2=p2->next;
                }

                if(sumll->start==NULL)
                {
                         sumll->start=p;
                }
                else
                {
                  q=sumll->start;
                         while(q ->next!=NULL)
                        {
                                q=q->next;
                        }
                  q->next=p;
                }
        }
}

void multiply(LL *ll1, LL *ll2,LL *prodll)
{
        node *p1,*p2,*prop,*p,*q;
        p1=ll1->start;
        p2=ll2->start;
        prop=prodll->start;
        while(p1!=NULL)
        {
                while(p2!=NULL)
                {
                        p=(node*)malloc(sizeof(node));
                        p->next=NULL;
```

```
                        p->pow=(p1->pow)+(p2->pow);
                        p->coeff=(p1->coeff)*(p2->coeff);
                        if(prodll->start==NULL)
                        {
                                prodll->start=p;
                                prop=prodll->start;
                        }
                        else
                        {
                                prop->next=p;
                                prop=p;
                        }
                        p2=p2->next;
                }
                p2=ll2->start;
                p1=p1->next;
        }
}

int main()
{
LL arr[4];
int i;
for(i=0;i<4;i++)
{
        arr[i].start=NULL;
}
int choice,ele,c1;
createlist(&arr[0]);
createlist(&arr[1]);
add(&arr[0],&arr[1], &arr[2]);
multiply(&arr[0], &arr[1], &arr[3]);
removeduplicate(&arr[3]);
printf("\nThe First equation is:- ");
display(&arr[0]);
printf("\nThe Second equation is:- ");
display(&arr[1]);
printf("\nThe Sum of two equations is:- \n");
display(&arr[2]);
printf("\nThe Product of two equations is:- \n");
display(&arr[3]);
return 0;
}
```

OUTPUT:-

```
Enter the degree of equation:- 5
Enter the coefficient of power 5 term:-   1
Enter the coefficient of power 4 term:-   5
Enter the coefficient of power 3 term:-   10
Enter the coefficient of power 2 term:-   10
Enter the coefficient of power 1 term:-   5
Enter the constant term:-          1


Enter the degree of equation:- 3
Enter the coefficient of power 3 term:-   1
Enter the coefficient of power 2 term:-   3
Enter the coefficient of power 1 term:-   3
Enter the constant term:-          1



The First equation is:- 1x^(5)+5x^(4)+10x^(3)+10x^(2)+5x^(1)+1

The Second equation is:- 1x^(3)+3x^(2)+3x^(1)+1

The Sum of two equations is:-
1x^(5)+5x^(4)+11x^(3)+13x^(2)+8x^(1)+2

The Product of two equations is:-
1x^(8)+8x^(7)+28x^(6)+56x^(5)+70x^(4)+56x^(3)+28x^(2)+8x^(1)+1
```