PROGRAM 22

Algorithm

Step 1:-Create a node using malloc function.

Step 2:-[insertion]Insertion can be performed at both the location i.e at the beginning and the end

Step 3:-[Deletion]The deletion application can be performed at the beginning as well as the end of the node.

Step 4:-if the End of the Node does not exist then underflow condition can be applied.

IF THE POLYNOMIAL HAVE SAME DEGREE THEY WILL BE ADDED

Step 5:-End

Source code

#include<stdio.h>

void create();

void display();

void polyadd();

struct list{

int coeff;

int pow;

struct list \*next;

};

struct list \*poly1,\*poly2,\*poly3;

void main()

{

poly1=poly2=poly3=NULL;

system("color F0");

poly1=(struct list\*)malloc(sizeof(struct list));

poly2=(struct list\*)malloc(sizeof(struct list));

poly3=(struct list\*)malloc(sizeof(struct list));

printf("Enter the first polynomial:\n");

create(poly1);

printf("Enter the second polynomial:\n");

create(poly2);

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

polyadd(poly1,poly2,poly3);

printf("first polynomial\n");

display(poly1);

printf("\nsecond polynomial\n");

display(poly2);

printf("\nprintf the polynomial:\n");

display(poly3);

}

void create(struct list \*n\_node)

{

char ch;

int c,p;

do{

printf("Enter the coeffiecient and power:\t");

scanf("%d%d",&c,&p);

n\_node->coeff=c;

n\_node->pow=p;

n\_node->next=(struct list\*)malloc(sizeof(struct list));

n\_node=n\_node->next;

n\_node->next=NULL;

printf("Continue\n");

ch=getch();

}while(ch=='y' || ch=='Y');

}

void display(struct list \*node)

{

while(node->next!=NULL)

{

printf("%dx^%d",node->coeff,node->pow);

if(node->coeff>0)

printf("+");

else

printf("-");

node=node->next;

}

}

void polyadd(struct list \*poly1,struct list \*poly2,struct list \*poly3)

{

while(poly1->next && poly2->next)

{

if(poly1->pow>poly2->pow)

{

poly3->coeff=poly1->coeff;

poly3->pow=poly1->pow;

poly1=poly1->next;

}

else if(poly2->pow>poly1->pow)

{

poly3->coeff=poly2->coeff;

poly3->pow=poly2->pow;

poly2=poly2->next;

}

else

{

poly3->coeff=poly1->coeff+poly2->coeff;

poly3->pow=poly1->pow;

poly1=poly1->next;

poly2=poly2->next;

}

poly3->next=(struct list\*)malloc(sizeof(struct list));

poly3=poly3->next;

poly3->next=NULL;

}

while(poly1->next ||poly2->next)

{

if(poly1->next)

{

poly3->coeff=poly1->coeff;

poly3->pow=poly1->pow;

poly1=poly1->next;

}

if(poly2->next)

{

poly3->coeff=poly2->coeff;

poly3->pow=poly2->pow;

poly2=poly2->next;

}

poly3->next=(struct list\*)malloc(sizeof(struct list));

poly3=poly3->next;

poly3->next=NULL;

}

}

