Experiment 12
Date 27-11-2022

POLYNOMIAL OPERATIONS USING LINKED LISTS

Aim

Write a C program to read two polynomials and store them using a linked list. Do the following operations on them.
a. Find the product of two polynomials and store the result using a linked list. Display the resultant polynomial.
b. Calculate the sum of the two polynomials and display the first polynomial, second polynomial and the resultant polynomial

1 Polynomial Operations

1.1 Algorithm

```
Step 1: Start
Step 2: Input number of terms of PolyA n1
Step 3: inputPoly(headA,n1)
Step 4: Input number of terms of PolyB n2
Step 5: inputPoly(headB,n2)
Step 6: displayPoly(headA)
Step 7: displayPoly(headB)
Step 8: sum(headA,headB,headS)
Step 9: product(headA,headB,headP)
Step 10: Stop
void inputPoly(struct node **head,int n)
Step 1: *head=createNode();
Step 2: struct node *ptr=createNode();
Step 3: If(n¿0) then
Step 4: (*head)→link=ptr
Step 5: Input ptr→ coeff,ptr→pow
Step 6: ptr→ link=NULL
Step 7: for(i=1;i¡n;i++)
Step 8: struct node *new =createNode();
Step 9: Input new→coeff,new→pow
Step 10: new→link=NULL
Step 11: ptr→link=new
Step 12: ptr=ptr→link
Step 13: EndFor
Step 14: EndIf
Step 15: return
void sum(struct node *headA,struct node *headB,struct node **headS)
Step 1: *headS=createNode()
Step 2: If(headA=NULL and headB=NULL)
Step 3: print sum is 0
Step 4: (*headS)→ link=NULL
Step 5: return
Step 6: EndIf
Step 7: ptrA=headA→ link,ptrB=headB→ link,ptrS=*headS
Step 8: while(ptrA!=NULL and ptrB!=NULL)
Step 9: ptrS→ link=createNode()
Step 10: ptrS=ptrS→ link
Step 11: If(ptrA→ pow equals ptrB→ pow)
Step 12: ptrS→ coeff=ptrA→ coeff+ptrB→ coeff;
Step 13: ptrS→ pow=ptrA→ pow;
Step 14: ptrA=ptrA→ link;
Step 15: ptrB=ptrB→ link;
Step 16: Else If (ptrA→ pow¿ptrB→ pow)
Step 17: *ptrS=*ptrA;
Step 18: ptrA=ptrA→ link;
Step 19: Else
Step 20: *ptrS=*ptrB;
Step 21: ptrB=ptrB→ link;
Step 22: EndIf
Step 23: EndWhile
Step 24: If there are remaining terms in polyA or polyB
Step 25: copy remaining terms to ptrS
Step 26: displayPOly(*headS)
Step 27: return
void product(struct node *headA,struct node *headB,struct node **headP)
Step 1: *headP=createNode();
Step 2: ptrA=headA→ link,ptrP=*headP;
Step 3: while(ptrA!=NULL)
Step 4: ptrB=headB→ link;
Step 5: while(ptrB!=NULL)
Step 6: ptrP→ link=createNode();
Step 7: ptrP=ptrP→ link;
Step 8: ptrP→ coeff=(ptrA→ coeff)*(ptrB→ coeff);
Step 9: ptrP→ pow=ptrA→ pow+ptrB→ pow;
Step 10: ptrB=ptrB→ link;
Step 11: EndWhile
Step 12: ptrA=ptrA→ link;
Step 13: EndWhile
Step 14: ptrP→ link=NULL;
Step 15: removeDup(*headP);
Step 16: displayPoly(*headP);
Step 17: return
void removeDup(struct node *head)
Step 1: i=head;
Step 2: while(i→ link!=NULL)
Step 3: j=i;
Step 4: while(j→ link!=NULL)
Step 5: if(i→ pow==j→ link→ pow)
Step 6: i→ coeff=i→ coeff+j→ link→ coeff;
Step 7: dup=j→ link;
Step 8: j→ link=j→ link→ link;
Step 9: returnNode(dup);
Step 10: Else
Step 11: j=j→ link;
Step 12: EndIf
Step 13: EndWhile
Step 14: i=i→ link
Step 15: return
```

1.2 Program

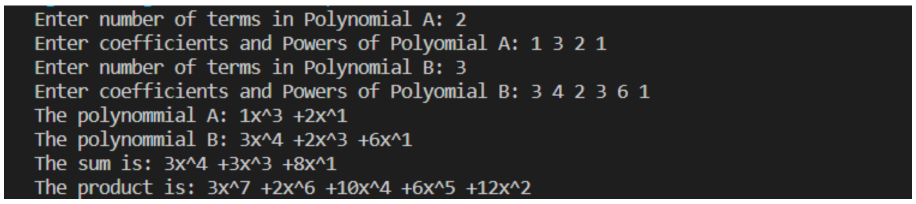
```
#include <stdio.h>
#include <stdlib.h>
struct node{
int pow;
int coeff;
struct node *link;
}node;

struct node* createNode(){
return (struct node*)malloc(sizeof(struct node));
}
void displayPoly(struct node *head){
struct node *ptr;
if(head->link==NULL){
printf("Polynimial Empty\n");
return;
}
ptr=head->link;
while(ptr->link!=NULL){
printf("%dx^%d +",ptr->coeff,ptr->pow);
ptr=ptr->link;
}
```

```
printf("%dx^%d\n",ptr->coeff,ptr->pow);
}
void inputPoly(struct node **head,int n){
int i;
struct node *ptr,*new;
*head=createNode();
ptr=createNode();
if(n){
(*head)->link=ptr;
scanf("%d%d",&ptr->coeff,&ptr->pow);
ptr->link=NULL;
for(i=1;i<n;i++){
new=createNode();
scanf("%d%d",&new->coeff,&new->pow);
new->link=NULL;
ptr->link=new;
ptr=ptr->link;
}
}
}
void sum(struct node *headA,struct node *headB,struct node **headS){
struct node *ptrA,*ptrB,*ptrS;
*headS=createNode();
if(headA==NULL&&headB==NULL){
printf("Polynomials empty: Sum is 0.\n");
(*headS)->link=NULL;
return;
}
ptrA=headA->link,ptrB=headB->link,ptrS=*headS;
while(ptrA!=NULL&&ptrB!=NULL){
ptrS->link=createNode();
ptrS=ptrS->link;
if(ptrA->pow==ptrB->pow){
ptrS->coeff=ptrA->coeff+ptrB->coeff;
ptrS->pow=ptrA->pow;
ptrA=ptrA->link;
ptrB=ptrB->link;
}
else if(ptrA->pow>ptrB->pow){
*ptrS=*ptrA;
ptrA=ptrA->link;
}
else{
*ptrS=*ptrB;
ptrB=ptrB->link;
}
```

```
}
while(ptrA!=NULL){
ptrS->link=createNode();
ptrS=ptrS->link;
*ptrS=*ptrA;
ptrA=ptrA->link;
}
while(ptrB!=NULL){
ptrS->link=createNode();
ptrS=ptrS->link;
*ptrS=*ptrB;
ptrB=ptrB->link;
}
ptrS->link=NULL;
printf("The sum is: ");
displayPoly(*headS);
}
void remove_dup(struct node * head){
struct node *i,*j,*dup;
i=head;
while(i->link!=NULL){
j=i;
while(j->link!=NULL){
if(i->pow==j->link->pow){
i->coeff=i->coeff+j->link->coeff;
dup=j->link;
j->link=j->link->link;
free(dup);
}
else
j=j->link;
}
i=i->link;
}
}
void product(struct node *headA,struct node *headB,struct node **headP){
struct node *ptrA,*ptrB,*ptrP;
*headP=createNode();
ptrA=headA->link;
ptrP=*headP;
while(ptrA!=NULL){
ptrB=headB->link;
while(ptrB!=NULL){
ptrP->link=createNode();
ptrP=ptrP->link;
ptrP->coeff=(ptrA->coeff)*(ptrB->coeff);
ptrP->pow=ptrA->pow+ptrB->pow;
ptrB=ptrB->link;
}
ptrA=ptrA->link;
}
ptrP->link=NULL;
remove_dup(*headP);
printf("The product is: ");
displayPoly(*headP);
}
int main(void){
int n1,n2,choice=1;
struct node *headA,*headB,*headS,*headP;
printf("Enter number of terms in Polynomial A: ");
scanf("%d",&n1);
printf("Enter coefficients and Powers of Polyomial A: ");
inputPoly(&headA,n1);
printf("Enter number of terms in Polynomial B: ");
scanf("%d",&n2);
printf("Enter coefficients and Powers of Polyomial B: ");
inputPoly(&headB,n2);
printf("The polynommial A: ");
displayPoly(headA);
printf("The polynommial B: ");
displayPoly(headB);
sum(headA,headB,&headS);
product(headA,headB,&headP);
}
```

1.3 Sample Output



```
Enter number of terms in Polynomial A: 2
Enter coefficients and Powers of Polyomial A: 1 3 2 1
Enter number of terms in Polynomial B: 3
Enter coefficients and Powers of Polyomial B: 3 4 2 3 6 1
The polynommial A: 1x^3 +2x^1
The polynommial B: 3x^4 +2x^3 +6x^1
The sum is: 3x^4 +3x^3 +8x^1
The product is: 3x^7 +2x^6 +10x^4 +6x^5 +12x^2
```

Figure 1: Input and Output

1.4 Result

A C program was made to input polynomials, then to calculate their sum and product. A function to display the polynomial represented using linked list was also made. Each node in the linked list stores a coefficient, an exponent and the link to the next term.