Rajalakshmi Engineering College

Name: Yashwanth Kumar V
Email: 240701609@rajalakshmi.edu.in
Roll no: 240701609
Phone: 8015927564
Branch: REC
Department: I CSE FF
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_week 1_CY

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Keerthi is a tech enthusiast and is fascinated by polynomial expressions. She loves to perform various operations on polynomials.

Today, she is working on a program to multiply two polynomials and delete a specific term from the result.

Keerthi needs your help to implement this program. She wants to take the coefficients and exponents of the terms of the two polynomials as input, perform the multiplication, and then allow the user to specify an exponent for deletion from the resulting polynomial, and display the result.

Input Format

The first line of input consists of an integer n, representing the number of terms

in the first polynomial.

The following n lines of input consist of two integers, each representing the coefficient and the exponent of the term in the first polynomial.

The next line consists of an integer m, representing the number of terms in the second polynomial.

The following m lines of input consist of two integers, each representing the coefficient and the exponent of the term in the second polynomial.

The last line consists of an integer, representing the exponent of the term that Keerthi wants to delete from the multiplied polynomial.

Output Format

The first line of output displays the resulting polynomial after multiplication.

The second line displays the resulting polynomial after deleting the specified term.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
2 2
3 1
4 0
2
1 2
2 1
2
Output: Result of the multiplication: 2x^4 + 7x^3 + 10x^2 + 8x
Result after deleting the term: 2x^4 + 7x^3 + 8x

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
```

```
struct node{
   int coeff,expo;
   struct node* next;
}*poly1=NULL,*poly2=NULL,*res=NULL;

typedef struct node poly;

void insert(poly**list,int c,int e){
   poly* newnode=(poly*)malloc(sizeof(poly));
   newnode->coeff=c;
   newnode->expo=e;
   newnode->next=NULL;
   if(*list==NULL){
      *list=newnode;
   }
   else{
      poly*pos=*list;
      while(pos->next!=NULL){
         pos=pos->next;
      }
      pos->next=newnode;
   }
}
void multi(poly*l1,poly*l2){
   poly* pos1=l1;
   while(pos1!=NULL){
      poly* pos2=l2;
      while(pos2!=NULL){
         insert(&res,pos1->coeff*pos2->coeff,pos1->expo+pos2->expo);
         pos2=pos2->next;
      }
      pos1=pos1->next;
   }
}
void sortl(poly*l1){
   poly* cur=l1;
   while(cur!=NULL){
      poly* run=cur->next;
      poly* pre=cur;
      while(run!=NULL){
         if(cur->expo==run->expo){
            cur->coeff+=run->coeff;
```

```
            pre->next=run->next;
            free(run);
            run=pre->next;
        }
        else{
            pre=run;
            run=run->next;
        }
    }
    cur=cur->next;
  }
}
void del(poly** list,int x){
  poly* pos=*list;
  poly* pre= NULL;
  if(*list==NULL)
    return;
  if(pos->expo==x){
    *list=pos->next;
    return;
  }
  else{
    while(pos!=NULL && pos->expo!=x){
      pre=pos;
      pos=pos->next;
    }
    if(pos==NULL)
      return;
    pre->next=pos->next;
  }
}
void display(poly* lis){
  poly* pos=lis;
  if(lis==NULL)
    return;
  while(pos->next!=NULL){
    if(pos->expo==1)
      printf("%dx + ",pos->coeff);
    else if(pos->expo==0)
      printf("%d + ",pos->coeff);
    else
      printf("%dx^%d + ",pos->coeff,pos->expo);
```

```
        pos=pos->next;
    }
    if(pos->expo==1)
        printf("%dx",pos->coeff);
    else if(pos->expo==0)
        printf("%d",pos->coeff);
    else
        printf("%dx^%d",pos->coeff,pos->expo);
    printf("\n");

}
int main(){
    int n,m,i,ele,ele1,r;
    scanf("%d",&n);
    for(i=0;i<n;i++){
        scanf("%d %d",&ele,&ele1);
        insert(&poly1,ele,ele1);
    }
    scanf("%d",&m);
    for(i=0;i<m;i++){
        scanf("%d %d",&ele,&ele1);
        insert(&poly2,ele,ele1);
    }
    multi(poly1,poly2);
    sortl(res);
    printf("Result of the multiplication: ");
    display(res);

    scanf("%d",&r);
    del(&res,r);
    printf("Result after deleting the term: ");
    display(res);
    return 0;
}
```

Status : Correct **Marks : 10/10**

2. Problem Statement

Timothy wants to evaluate polynomial expressions for his mathematics homework. He needs a program that allows him to input the coefficients of

a polynomial based on its degree and compute the polynomial's value for a given input of x. Implement a function that takes the degree, coefficients, and the value of x, and returns the evaluated result of the polynomial.

Example

Input:

degree of the polynomial = 2

coefficient of x2 = 13

coefficient of x1 = 12

coefficient of x0 = 11

x = 1

Output:

36

Explanation:

Calculate the value of 13x2: 13 * 12 = 13.

Calculate the value of 12x1: 12 * 11 = 12.

Calculate the value of 11x0: 11 * 10 = 11.

Add the values of x2, x1, and x0 together: 13 + 12 + 11 = 36.

Input Format

The first line of input consists of an integer representing the degree of the polynomial.

The second line consists of an integer representing the coefficient of x2.

The third line consists of an integer representing the coefficient of x1.

The fourth line consists of an integer representing the coefficient of x0.

The fifth line consists of an integer representing the value of x, at which the polynomial should be evaluated.

Output Format

The output is an integer value obtained by evaluating the polynomial at the given value of x.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
13
12
11
1
Output: 36

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
struct node{
   int x1,x2,x3;
   struct node* next;
} *head=NULL;
typedef struct node Node;
void display(int m){
   Node* pos1=head;
   int e=(((pos1->x1)*(m*m))+((pos1->x2)*m)+(pos1->x3));
   printf("%d",e);

}

void insert(int x1,int x2,int x3){
   Node* newnode=(Node*)malloc(sizeof(Node));
   newnode->x1=x1;
   newnode->x2=x2;
   newnode->x3=x3;
   newnode->next=NULL;
   if(head==NULL)
      head=newnode;
   else{
      Node* pos=head;
```

```
        while(pos->next!=NULL){
            pos=pos->next;
        }
        pos->next=newnode;
    }

}

int main(){
    int x,x1,x2,x3,i,n;
    scanf("%d",&n);
    scanf("%d %d %d",&x1,&x2,&x3);
    scanf("%d",&x);
    insert(x1,x2,x3);
    display(x);
    return 0;
}
```

Status : Correct **Marks : 10/10**

3. Problem Statement

Lisa is studying polynomials in her class. She is learning about the multiplication of polynomials.

To practice her understanding, she wants to write a program that multiplies two polynomials and displays the result. Each polynomial is represented as a linked list, where each node contains the coefficient and exponent of a term.

Example

Input:

4 3

y

3 1

y

1 0

n

2 2

y

3 1

y

2 0

n

Output:

8x^5 + 12x^4 + 14x^3 + 11x^2 + 9x + 2

Explanation

1. Poly1: 4x^3 + 3x + 1
2. Poly2: 2x^2 + 3x + 2

Multiplication Steps:

1. Multiply 4x^3 by Poly2:
   -> 4x^3 * 2x^2 = 8x^5
   -> 4x^3 * 3x = 12x^4
   -> 4x^3 * 2 = 8x^3
2. Multiply 3x by Poly2:
   -> 3x * 2x^2 = 6x^3
   -> 3x * 3x = 9x^2
   -> 3x * 2 = 6x
3. Multiply 1 by Poly2:
   -> 1 * 2x^2 = 2x^2

-> 1 * 3x = 3x

-> 1 * 2 = 2

Combine the results: 8x^5 + 12x^4 + (8x^3 + 6x^3) + (9x^2 + 2x^2) + (6x + 3x) + 2

The combined polynomial is: 8x^5 + 12x^4 + 14x^3 + 11x^2 + 9x + 2

Input Format

The input consists of two sets of polynomial terms.

Each polynomial term is represented by two integers separated by a space:

- The first integer represents the coefficient of the term.
- The second integer represents the exponent of the term.

After entering a polynomial term, the user is prompted to input a character indicating whether to continue adding more terms to the polynomial.

If the user inputs 'y' or 'Y', the program continues to accept more terms.

If the user inputs 'n' or 'N', the program moves on to the next polynomial.

Output Format

The output consists of a single line representing the resulting polynomial after multiplying the two input polynomials.

Each term of the resulting polynomial is formatted as follows:

- The coefficient and exponent are separated by 'x^' if the exponent is greater than 1.
- If the exponent is 1, only 'x' is displayed without the exponent.
- If the exponent is 0, only the coefficient is displayed.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 4 3
y
3 1
y
1 0
n
2 2
y
3 1
y
2 0
n

Output: 8x^5 + 12x^4 + 14x^3 + 11x^2 + 9x + 2

Answer

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

struct node{
   int coeff,expo;
   struct node* next;
}*poly1=NULL,*poly2=NULL,*res=NULL;

typedef struct node poly;

void insert(poly** list,int c, int e){
   poly* newnode=(poly*)malloc(sizeof(poly));
   newnode->coeff=c;
   newnode->expo=e;
   newnode->next=NULL;
   if(*list==NULL)
      *list=newnode;
   else{
      poly* pos= *list;
      while(pos->next!=NULL)
```

```
        pos=pos->next;
      pos->next=newnode;
    }
}

void multi(poly* l1,poly* l2){
    poly* pos1=l1;
    while(pos1!=NULL){
        poly* pos2=l2;
        while(pos2!=NULL){
            insert(&res,pos1->coeff*pos2->coeff,pos1->expo+pos2->expo);
            pos2=pos2->next;
        }
        pos1=pos1->next;
    }
}

void sortl(poly* res){
    poly* cur=res;
    while(cur!=NULL){
        poly* run=cur->next;
        poly* pre=cur;
        while(run!=NULL){
            if(cur->expo==run->expo){
                cur->coeff+=run->coeff;
                pre->next=run->next;
                free(run);
                run=pre->next;
            }
            else{
                pre=run;
                run=run->next;
            }
        }
        cur=cur->next;
    }
}
void display(poly*l1){
    poly*pos=l1;
    if(l1==NULL)
        return;
    while(pos->next!=NULL){
```

```
        if(pos->expo==1)
            printf("%dx + ",pos->coeff);
        else if(pos->expo==0)
            printf("%d + ",pos->coeff);
        else
            printf("%dx^%d + ",pos->coeff,pos->expo);
        pos = pos -> next;
    }
    if(pos->expo==1)
        printf("%dx",pos->coeff);
    else if(pos->expo==0)
        printf("%d",pos->coeff);
    else
        printf("%dx^%d",pos->coeff,pos->expo);
}


int main(){
    int ele,ele1,i;
    char ch;
    do{
        scanf("%d %d",&ele,&ele1);
        insert(&poly1,ele,ele1);
        getchar();
        scanf("%c",&ch);


    }while(tolower(ch)!='n');

    do{
        scanf("%d %d",&ele,&ele1);
        insert(&poly2,ele,ele1);
        getchar();
        scanf("%c",&ch);
    }while(tolower(ch)!='n');
    multi(poly1,poly2);
    sortl(res);
    display(res);
    return 0;
}
```

Status : Correct *Marks : 10/10*