Rajalakshmi Engineering College

Name: Steve Anderson S
Email: 240701537@rajalakshmi.edu.in
Roll no: 240701537
Phone: 8925524775
Branch: REC
Department: I CSE FE
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

Akila is a tech enthusiast and wants to write a program to add two polynomials. Each polynomial is represented as a linked list, where each node in the list represents a term in the polynomial.

A term in the polynomial is represented in the format ax^b, where a is the coefficient and b is the exponent.

Akila needs your help to implement a program that takes two polynomials as input, adds them, and stores the result in ascending order in a new polynomial-linked list. Write a program to help her.

Input Format

The input consists of lines containing pairs of integers representing the

coefficients and exponents of polynomial terms.

Each line represents a single term, with the coefficient and exponent separated by a space.

The input for each polynomial ends with a line containing "0 0".

Output Format

The output consists of three lines representing the first, second, and resulting polynomial after the addition operation, with terms sorted in ascending order of exponents.

Each line contains terms of the polynomial in the format "coefficientx^exponent", separated by " + ".

If the resulting polynomial is zero, the output is "0".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3 4
2 3
1 2
0 0
1 2
2 3
3 4
0 0

Output: 1x^2 + 2x^3 + 3x^4
1x^2 + 2x^3 + 3x^4
2x^2 + 4x^3 + 6x^4

Answer

```
// You are using GCC
#include <stdio.h>
#include <stdlib.h>
typedef struct Node{
    int coeff,expo;
    struct Node* next;
```

```
}Poly;
Poly* insert(Poly* head, int coeff, int expo);
Poly* add(Poly* p1,Poly* p2);
void display(Poly*head);
int main(){
   Poly* poly1=NULL,*poly2=NULL,*result=NULL;
   int coeff,expo;
   while(scanf("%d %d",&coeff,&expo)&&(coeff||expo))
      poly1=insert(poly1,coeff,expo);
   while(scanf("%d %d",&coeff,&expo)&&(coeff||expo))
      poly2=insert(poly2,coeff,expo);
   result=add(poly1,poly2);
   display(poly1);
   display(poly2);
   display(result);
   return 0;
}
Poly* insert(Poly*head, int coeff,int expo){
   Poly* newNode=(Poly*)malloc(sizeof(Poly)),*temp=head,*prev=NULL;
   newNode->coeff=coeff;
   newNode->expo=expo;
   newNode->next=NULL;
   if(!head||expo<head->expo){
      newNode->next=head;
      return newNode;
   }
   while(temp&&temp->expo<expo){
      prev=temp;
      temp=temp->next;
   }
   if(temp&&temp->expo==expo){
      temp->coeff+=coeff;
      return head;
   }
   prev->next=newNode;
   newNode->next=temp;
   return head;
}
Poly* add(Poly* p1,Poly*p2){
   Poly*result=NULL;
   while(p1||p2){
      if(!p2||(p1&&p1->expo<p2->expo)){
```

```
            result=insert(result,p1->coeff,p1->expo);
            p1=p1->next;
        }
        else if(!p1||(p2&&p1->expo>p2->expo)){
            result=insert(result,p2->coeff,p2->expo);
            p2=p2->next;
        }
        else{
            if(p1->coeff+p2->coeff)
                result=insert(result,p1->coeff+p2->coeff,p1->expo);
            p1=p1->next;
            p2=p2->next;
        }
    }
    return result;
}
void display(Poly*head){
    if(!head){
        printf("0\n");
        return;
    }
    while(head){
        printf("%dx^%d",head->coeff,head->expo);
        if(head->next) printf(" + ");
        head=head->next;
    }
    printf("\n");
}
```

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

2. 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>
typedef struct Node{
    int coeff,expo;
    struct Node* next;
}Poly;

Poly* insert(Poly* head,int coeff, int expo);
Poly* multiply(Poly* p1,Poly* p2);
void display(Poly* head);
Poly* deletepoly(Poly*head,int expo);

int main(){
    Poly *poly1=NULL,*poly2=NULL,*result=NULL;
    int coeff,expo,n,m;
    scanf("%d",&n);
    for (int i=0;i<n;i++){
        scanf("%d %d",&coeff,&expo);
        poly1=insert(poly1,coeff,expo);
    }
    scanf("%d",&m);
    for (int i=0;i<m;i++){
        scanf("%d %d",&coeff,&expo);
        poly2=insert(poly2,coeff,expo);
    }
    result=multiply(poly1,poly2);
    printf("Result of the multiplication: ");
    display(result);
    scanf("%d",&expo);
    result=deletepoly(result,expo);
    printf("Result after deleting the term: ");
    display(result);
    return 0;
}

Poly* insert(Poly* head,int coeff,int expo){
    Poly* newNode=(Poly*)malloc(sizeof(Poly));
    newNode->coeff=coeff;
```

```
    newNode->expo=expo;
    newNode->next=NULL;
    if(!head){
        return newNode;
    }
    Poly* temp=head;
    Poly* prev=NULL;
    while(temp){
        if(temp->expo==expo){
            temp->coeff+=coeff;
            free(newNode);
            return head;
        }
        prev=temp;
        temp=temp->next;
    }
    prev->next=newNode;
    return head;
}

Poly* multiply(Poly* p1,Poly* p2){
    if(!p1||!p2) return NULL;
    Poly* result=NULL;
    for(Poly* temp1=p1;temp1;temp1=temp1->next){
        for(Poly* temp2=p2;temp2;temp2=temp2->next){
            result=insert(result,temp1->coeff*temp2->coeff,temp1->expo+temp2-
>expo);
        }
    }
    return result;
}

void display(Poly*head){
    if(!head){
        printf("0\n");
        return;
    }
    Poly*temp=head;
    while(temp){
        if(temp->coeff<0) printf(" - ");
        else if(temp!=head) printf(" + ");
        int coeff=abs(temp->coeff);
```

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

Poly* deletepoly(Poly* head, int expo){
    Poly *temp=head, *prev=NULL;
    if(head->expo==expo){
        head=head->next;
        free(temp);
        return head;
    }
    while(temp&&temp->expo!=expo){
        prev=temp;
        temp=temp->next;
    }
    if(!temp) return head;
    prev->next=temp->next;
    free(temp);
    return head;
}
```

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

3. Problem Statement

Hayley loves studying polynomials, and she wants to write a program to compare two polynomials represented as linked lists and display whether they are equal or not.

The polynomials are expressed as a series of terms, where each term consists of a coefficient and an exponent. The program should read the polynomials from the user, compare them, and then display whether they are equal or not.

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 of input 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.

Output Format

The first line of output prints "Polynomial 1: " followed by the first polynomial.

The second line prints "Polynomial 2: " followed by the second polynomial.

The polynomials should be displayed in the format ax^b, where a is the coefficient and b is the exponent.

If the two polynomials are equal, the third line prints "Polynomials are Equal."

If the two polynomials are not equal, the third line prints "Polynomials are Not Equal."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 2
1 2
2 1
2
1 2
2 1

Output: Polynomial 1: (1x^2) + (2x^1)
Polynomial 2: (1x^2) + (2x^1)
Polynomials are Equal.

Answer

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

```c
#include <stdlib.h>
typedef struct Node{
    int coeff,expo;
    struct Node* next;
}Poly;

Poly*insert(Poly*head,int coeff,int expo);
int isEqual(Poly*p1,Poly*p2);
void display(Poly*head);

int main(){
    Poly*poly1=NULL,*poly2=NULL,*result=NULL;
    int coeff,expo,n,m;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d %d",&coeff,&expo);
        poly1=insert(poly1,coeff,expo);
    }
    scanf("%d",&m);
    for(int i=0;i<m;i++){
        scanf("%d %d",&coeff,&expo);
        poly2=insert(poly2,coeff,expo);
    }
    printf("Polynomial 1: ");
    display(poly1);
    printf("Polynomial 2: ");
    display(poly2);
    if(!isEqual(poly1,poly2)) printf("Polynomials are Not Equal.");
    else printf("Polynomials are Equal.");
}

Poly* insert(Poly*head,int coeff,int expo){
    Poly* newNode=(Poly*)malloc(sizeof(Poly));
    newNode->coeff=coeff;
    newNode->expo=expo;
    newNode->next=NULL;
    if(!head) return newNode;
    Poly*temp=head;
    while(temp->next){
        temp=temp->next;
    }
    temp->next=newNode;
```

```
    return head;
}

int isEqual(Poly*p1,Poly*p2){
    while(p1&&p2){
        if(p1->coeff!=p2->coeff||p1->expo!=p2->expo) return 0;
        p1=p1->next;
        p2=p2->next;
    }
    return(p1==NULL&&p2==NULL);
}

void display(Poly*head){
    while(head){
        printf("(%dx^%d)",head->coeff,head->expo);
        if(head->next) printf(" + ");
        head=head->next;
    }
    printf("\n");
}
```

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