Rajalakshmi Engineering College

Name: NISHANTH B
Email: 240701364@rajalakshmi.edu.in
Roll no: 240701364
Phone: 7904264876
Branch: REC
Department: I CSE FD
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 : 23

Section 1 : Coding

1. Problem Statement

Hasini is studying polynomials in her class. Her teacher has introduced a new concept of two polynomials using linked lists.

The teacher provides Hasini with a program that takes two polynomials as input, represented as linked lists, and then displays them together. The polynomials are simplified and should be displayed in the format ax^b, where a is the coefficient and b is the exponent.

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: 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: the coefficient and the exponent of the term in the second polynomial.

Output Format

The first line of output prints the first polynomial.

The second line of output prints the second polynomial.

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

Refer to the sample output for formatting specifications.

Sample Test Case

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

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

Answer

```
#include<stdio.h>
#include<stdlib.h>
typedef struct Node{
   int coeff;
   int expo;
   struct Node*next;
}Node;
void insertatend(Node **head,int coeff,int expo){
   Node*n=(Node*)malloc(sizeof(Node));
```

```
    n->coeff=coeff;
    n->expo=expo;
    n->next=NULL;
    if(*head==NULL){
        *head=n;
        return;
    }
    Node *pos=*head;
    for(;pos->next!=NULL;pos=pos->next);
    pos->next=n;
}

void display(Node*head){
    Node*n;
    for(n=head;n->next!=NULL;n=n->next){
        if(n->expo>=2)
        printf("%dx^%d + ",n->coeff,n->expo);
        else if (n->expo==1)
        printf("%dx + ",n->coeff);
        else
        printf("%d + ",n->coeff);
    }
    if(n->expo>=2)
        printf("%dx^%d\n",n->coeff,n->expo);
        else if (n->expo==1)
        printf("%dx\n",n->coeff);
        else
        printf("%d\n",n->coeff);
}
int main(){
    int n,m;
    scanf("%d",&n);
    Node*head1=NULL,*head2=NULL;
    for(int i=0;i<n;i++){
        int c,e;
        scanf("%d %d",&c,&e);
        insertatend(&head1,c,e);
    }
    scanf("%d",&m);
    for(int i=0;i<m;i++){
        int c,e;
        scanf("%d %d",&c,&e);
```

```
        insertatend(&head2,c,e);
    }
    display(head1);
    display(head2);
    return 0;
}
```

Status : Partially correct **Marks : 3/10**

2. 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

```
#include<stdio.h>
#include<stdlib.h>
typedef struct Node{
    int coeff;
    int exp;
    struct Node*next;
}Node;
Node* createNode(int coeff,int exp){
    Node* newNode=(Node*)malloc(sizeof(Node));
    newNode->coeff=coeff;
    newNode->exp=exp;
    newNode->next=NULL;
    return newNode;
}
void insertTerm(Node** poly,int coeff,int exp){
    if(coeff==0) return;
    Node* newNode=createNode(coeff,exp);
    if(*poly==NULL || (*poly)->exp>exp){
        newNode->next=*poly;
        *poly=newNode;
        return;
```

```
    }
    Node* temp=*poly;
    while(temp->next!=NULL&&temp->next->exp<exp){
        temp=temp->next;
    }
    if(temp->next!=NULL&&temp->next->exp==exp){
        temp->next->coeff+=coeff;
        if(temp->next->coeff==0){
            Node* toDelete=temp->next;
            temp->next=temp->next->next;
            free(toDelete);
        }
        free(newNode);
    }else{
        newNode->next=temp->next;
        temp->next=newNode;
    }
}
Node* readPolynomial(){
    Node* poly=NULL;
    int coeff,exp;
    while(1){
        scanf("%d %d",&coeff,&exp);
        if(coeff==0&&exp==0)
        break;
        insertTerm(&poly,coeff,exp);
    }
    return poly;
}
void printPolynomial(Node* poly){
    if(poly==NULL){
        printf("0\n");
        return;
    }
    while(poly!=NULL){
        printf("%dx^%d",poly->coeff,poly->exp);
        if(poly->next!=NULL){
            printf(" + ");

        }
        poly=poly->next;
    }
```

```
    printf("\n");
}
Node* addPolynomials(Node* poly1,Node* poly2){
    Node* result=NULL;
    while(poly1!=NULL || poly2!=NULL){
        if(poly1==NULL){
            insertTerm(&result,poly2->coeff,poly2->exp);
            poly2=poly2->next;
        }else if(poly2==NULL){
            insertTerm(&result,poly1->coeff,poly1->exp);
            poly1=poly1->next;
        }else if(poly1->exp==poly2->exp){
            insertTerm(&result,poly1->coeff+poly2->coeff,poly1->exp);
            poly1=poly1->next;
            poly2=poly2->next;
        }else if(poly1->exp<poly2->exp){
            insertTerm(&result,poly1->coeff,poly1->exp);
            poly1=poly1->next;
        }else{
            insertTerm(&result,poly2->coeff,poly2->exp);
            poly2=poly2->next;
        }

    }
    return result;
}
void freePolynomial(Node*poly){
    Node* temp;
    while(poly!=NULL){
        temp=poly;
        poly=poly->next;
        free(temp);
    }
}
int main(){
    Node* poly1=readPolynomial();
    Node* poly2=readPolynomial();
    Node*sum=addPolynomials(poly1,poly2);
    printPolynomial(poly1);
    printPolynomial(poly2);
    printPolynomial(sum);
    freePolynomial(poly1);
```

```
    freePolynomial(poly2);
    freePolynomial(sum);
    return 0;
}
```

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

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
typedef struct node{
    int coeff,exp;
    struct node*next;
}node;
node*create(int coeff,int exp){
    node*newnode=(node*)malloc(sizeof(node));
    newnode->coeff=coeff;
    newnode->exp=exp;
    newnode->next=0;
    return newnode;
}
void insert(node**head,int coeff,int exp){
    node*newnode=create(coeff,exp);
    if(*head==0){
        *head=newnode;
    }
    else{
        node *temp=*head;
```

```
        while(temp->next!=0){
            temp=temp->next;
        }
        temp->next=newnode;
    }
}
int areequal(node *head1,node*head2){
    while(head1&&head2){
        if(head1->coeff!=head2->coeff || head1->exp!=head2->exp)
        return 0;
        head1=head1->next;
        head2=head2->next;
    }
    return(head1==0&&head2==0);
}
void printList(node *head){
    node *temp=head;
    while(temp!=0){
        printf("(%dx^%d)",temp->coeff,temp->exp);
        if(temp->next)
        printf(" + ");
        temp=temp->next;
    }
    printf("\n");
}
int main(){
    node *head1=0,*head2=0;
    int n,m,coeff,exp;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d %d",&coeff,&exp);
        insert(&head1,coeff,exp);
    }
    scanf("%d",&m);
    for(int i=0;i<m;i++){
        scanf("%d %d",&coeff,&exp);
        insert(&head2,coeff,exp);
    }
    printf("Polynomial 1: ");
    printList(head1);
    printf("Polynomial 2: ");
    printList(head2);
```

```
    if(areequal(head1,head2))
    printf("Polynomials are Equal.");
    else
    printf("Polynomials are Not Equal.");
}
```

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