Rajalakshmi Engineering College

Name: Sruthi Dinesh
Email: 240701536@rajalakshmi.edu.in
Roll no: 240701536
Phone: 7845725087
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_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Janani is a tech enthusiast who loves working with polynomials. She wants to create a program that can add polynomial coefficients and provide the sum of their coefficients.

The polynomials will be represented as a linked list, where each node of the linked list contains a coefficient and an exponent. The polynomial is represented in the standard form with descending order of exponents.

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 output prints the sum of the coefficients of the polynomials.

Sample Test Case

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

Output: 18

Answer

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

// Define the structure for a polynomial term
struct Node {
    int coeff;
    int expo;
    struct Node* next;
};

// Function to create a new node
struct Node* createNode(int coeff, int expo) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->coeff = coeff;
    newNode->expo = expo;
    newNode->next = NULL;
    return newNode;
}
```

```
// Function to insert a term in descending order of exponent
void insertTerm(struct Node** head, int coeff, int expo) {
    struct Node* newNode = createNode(coeff, expo);
    if (*head == NULL || (*head)->expo < expo) {
        newNode->next = *head;
        *head = newNode;
    } else {
        struct Node* current = *head;
        struct Node* prev = NULL;
        while (current != NULL && current->expo > expo) {
            prev = current;
            current = current->next;
        }
        if (current != NULL && current->expo == expo) {
            current->coeff += coeff;
            free(newNode);
        } else {
            newNode->next = current;
            if (prev) prev->next = newNode;
        }
    }
}

// Function to add two polynomials
struct Node* addPolynomials(struct Node* poly1, struct Node* poly2) {
    struct Node* result = NULL;
    while (poly1 != NULL) {
        insertTerm(&result, poly1->coeff, poly1->expo);
        poly1 = poly1->next;
    }
    while (poly2 != NULL) {
        insertTerm(&result, poly2->coeff, poly2->expo);
        poly2 = poly2->next;
    }
    return result;
}

// Function to calculate sum of coefficients
int sumOfCoefficients(struct Node* head) {
    int sum = 0;
    while (head != NULL) {
        sum += head->coeff;
```

```
        head = head->next;
    }
    return sum;
}

int main() {
    int n, m, coeff, expo;
    struct Node* poly1 = NULL;
    struct Node* poly2 = NULL;

    // Input first polynomial
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        scanf("%d %d", &coeff, &expo);
        insertTerm(&poly1, coeff, expo);
    }

    // Input second polynomial
    scanf("%d", &m);
    for (int i = 0; i < m; i++) {
        scanf("%d %d", &coeff, &expo);
        insertTerm(&poly2, coeff, expo);
    }

    // Add polynomials
    struct Node* sumPoly = addPolynomials(poly1, poly2);

    // Output sum of coefficients
    int sum = sumOfCoefficients(sumPoly);
    printf("%d\n", sum);

    return 0;
}
```

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