#include <iostream>

using namespace std;

struct Poly {

    int coeff;   // Coefficient

    int power;   // Power of the term

    Poly\* next;  // Pointer to the next term

};

class PolynomialOperations {

    Poly \*head1, \*head2, \*result;

public:

    PolynomialOperations() : head1(NULL), head2(NULL), result(NULL) {}

    // Function to create a new term

    Poly\* createTerm(int coeff, int power) {

        Poly\* newTerm = new Poly;

        newTerm->coeff = coeff;

        newTerm->power = power;

        newTerm->next = NULL;

        return newTerm;

    }

    // Function to insert a term into a polynomial

    void insertTerm(Poly\*& head, int coeff, int power) {

        Poly\* newTerm = createTerm(coeff, power);

        if (!head) {

            head = newTerm;

        } else {

            Poly\* temp = head;

            while (temp->next)

                temp = temp->next;

            temp->next = newTerm;

        }

    }

    // Input polynomial terms

    void inputPolynomial(Poly\*& head) {

        int coeff, power, choice;

        do {

            cout << "Enter coefficient: ";

            cin >> coeff;

            cout << "Enter power: ";

            cin >> power;

            insertTerm(head, coeff, power);

            cout << "Add another term? (1 for yes, 0 for no): ";

            cin >> choice;

        } while (choice == 1);

    }

    // Display a polynomial

    void displayPolynomial(Poly\* head) {

        if (!head) {

            cout << "0\n";

            return;

        }

        Poly\* temp = head;

        while (temp) {

            cout << temp->coeff << "x^" << temp->power;

            if (temp->next)

                cout << " + ";

            temp = temp->next;

        }

        cout << endl;

    }

    // Perform addition of two polynomials

    void addPolynomials() {

        Poly \*p1 = head1, \*p2 = head2;

        while (p1 && p2) {

            if (p1->power == p2->power) {

                insertTerm(result, p1->coeff + p2->coeff, p1->power);

                p1 = p1->next;

                p2 = p2->next;

            } else if (p1->power > p2->power) {

                insertTerm(result, p1->coeff, p1->power);

                p1 = p1->next;

            } else {

                insertTerm(result, p2->coeff, p2->power);

                p2 = p2->next;

            }

        }

        // Add remaining terms from p1

        while (p1) {

            insertTerm(result, p1->coeff, p1->power);

            p1 = p1->next;

        }

        // Add remaining terms from p2

        while (p2) {

            insertTerm(result, p2->coeff, p2->power);

            p2 = p2->next;

        }

    }

    // Execute the polynomial addition process

    void execute() {

        cout << "Enter the first polynomial:\n";

        inputPolynomial(head1);

        cout << "First Polynomial: ";

        displayPolynomial(head1);

        cout << "Enter the second polynomial:\n";

        inputPolynomial(head2);

        cout << "Second Polynomial: ";

        displayPolynomial(head2);

        addPolynomials();

        cout << "Resultant Polynomial after addition: ";

        displayPolynomial(result);

    }

};

int main() {

    PolynomialOperations polyOps;

    polyOps.execute();

    return 0;

}