```
#include <iostream>
#include <string>
#include <vector>

using namespace std;



class Node {
public:
    int coefficient;
    int exponent;
    Node *next;
    Node(int coefficient, int exponent) {
        this->coefficient = coefficient;
        this->exponent = exponent;
        this->next = NULL;
    }
};

class Polynomial {
public:
    Node *head;
    Polynomial() {
        head = NULL;
    }
    void add(int coefficient, int exponent) {
        Node *newNode = new Node(coefficient, exponent);
        if (head == NULL) {
            head = newNode;
        } else {
            Node *temp = head;
            while (temp->next != NULL) {
                temp = temp->next;
            }
            temp->next = newNode;
        }
    }
    void print() {
        Node *temp = head;
        while (temp != NULL) {
            cout << temp->coefficient << "x^" << temp->exponent;
            if (temp->next != NULL) {
                cout << " + ";
            }
            temp = temp->next;
```

```
        }
        cout << endl;
    }
    Polynomial addition(Polynomial p) {
        Polynomial result;
        Node *temp1 = head;
        Node *temp2 = p.head;
        while (temp1 != NULL && temp2 != NULL) {
            if (temp1->exponent == temp2->exponent) {
                result.add(temp1->coefficient + temp2->coefficient,
temp1->exponent);
                temp1 = temp1->next;
                temp2 = temp2->next;
            } else if (temp1->exponent > temp2->exponent) {
                result.add(temp1->coefficient, temp1->exponent);
                temp1 = temp1->next;
            } else {
                result.add(temp2->coefficient, temp2->exponent);
                temp2 = temp2->next;
            }
        }
        while (temp1 != NULL) {
            result.add(temp1->coefficient, temp1->exponent);
            temp1 = temp1->next;
        }
        while (temp2 != NULL) {
            result.add(temp2->coefficient, temp2->exponent);
            temp2 = temp2->next;
        }
        return result;
    }
    Polynomial multiplication(Polynomial p) {
        Polynomial result;
        Node *temp1 = head;
        Node *temp2 = p.head;
        while (temp1 != NULL) {
            while (temp2 != NULL) {
                result.add(temp1->coefficient * temp2->coefficient,
temp1->exponent + temp2->exponent);
                temp2 = temp2->next;
            }
            temp1 = temp1->next;
            temp2 = p.head;
        }
        return result;
    }
```

```
};

int main(){
    Polynomial p1;
    p1.add(2, 3);
    p1.add(3, 2);
    p1.add(4, 1);
    p1.add(5, 0);
    p1.print();
    Polynomial p2;
    p2.add(1, 2);
    p2.add(2, 1);
    p2.add(3, 0);
    p2.print();
    Polynomial p3 = p1.addition(p2);
    p3.print();
    Polynomial p4 = p1.multiplication(p2);
    p4.print();
    return 0;
}

// Language: cpp
// BigO: O(n^2)

Output:
2x^3 + 3x^2 + 4x^1 + 5x^0
1x^2 + 2x^1 + 3x^0
2x^3 + 4x^2 + 6x^1 + 8x^0
2x^5 + 4x^4 + 6x^3 + 3x^4 + 6x^3 + 9x^2 + 4x^3 + 8x^2 + 12x^1 + 5x^2 +
10x^1 + 15x^0
```