1.想法:

我參考課本、PPT 以及上網查資料來拼湊出這個程式

2. Algorithm Design & Programming

```
#include <iostream>
#include <cmath>
using namespace std;

class Polynomial {
private:
    int* coefficients;
    int degree;
public:
    Polynomial() : degree(0), coefficients(NULL) {}
    explicit Polynomial(int deg) : degree(deg) {
        coefficients = new int[degree + 1]{0};
    }
    ~Polynomial() {
        delete[] coefficients;
    }
    void setCoefficient(int power, int coeff) {
        if (power > degree) {
            int* newCoefficients = new int[power + 1]{0};
            for (int i = 0; i <= degree; ++i) {
                newCoefficients[i] = coefficients[i];
            }
            delete[] coefficients;
            coefficients = newCoefficients;
            degree = power;
        }
        coefficients[power] = coeff;
    }
    int getCoefficient(int power) const {
        if (power > degree) return 0;
        return coefficients[power];
    }
    int getDegree() const {
        return degree;
    }
    friend istream& operator>>(istream& in, Polynomial& poly) {
        cout << "請輸入多項式的最高次方數 :";
        in >> poly.degree;
        delete[] poly.coefficients;
```

```
    poly.coefficients = new int[poly.degree + 1]{0};
    cout << "請從最低到最高依次輸入係數（共 " << poly.degree + 1 << " 項):\n";
    for (int i = 0; i <= poly.degree; ++i) {
        cout << "x^" << i << " 的系數:";
        in >> poly.coefficients[i];
    }
    return in;
}
friend ostream& operator<<(ostream& out, const Polynomial& poly) {
    bool first = true;
    for (int i = poly.degree; i >= 0; --i) {
        if (poly.coefficients[i] != 0) {
            if (!first && poly.coefficients[i] > 0) out << " + ";
            if (poly.coefficients[i] < 0) out << " - ";
            if (abs(poly.coefficients[i]) != 1 || i == 0) out << abs(poly.coefficients[i]);
            if (i > 0) out << "x";
            if (i > 1) out << "^" << i;
            first = false;
        }
    }
    if (first) out << "0";
    return out;
}
//加法
Polynomial operator+(const Polynomial& other) const {
    int maxDegree = max(degree, other.degree);
    Polynomial result(maxDegree);
    for (int i = 0; i <= maxDegree; ++i) {
        result.setCoefficient(i, this->getCoefficient(i) + other.getCoefficient(i));
    }
    return result;
}
//減法
Polynomial operator-(const Polynomial& other) const {
    int maxDegree = max(degree, other.degree);
    Polynomial result(maxDegree);
    for (int i = 0; i <= maxDegree; ++i) {
        result.setCoefficient(i, this->getCoefficient(i) - other.getCoefficient(i));
    }
```

```
        return result;
    }
    //乘法
    Polynomial operator*(const Polynomial& other) const {
        int resultDegree = this->degree + other.degree;
        Polynomial result(resultDegree);
        for (int i = 0; i <= this->degree; ++i) {
            for (int j = 0; j <= other.degree; ++j) {
                result.coefficients[i + j] += this->coefficients[i] * other.coefficients[j];
            }
        }
        return result;
    }
};
int main() {
    Polynomial p1, p2;
    cout << "請輸入第一個多項式 :" << endl;
    cin >> p1;
    cout << "請輸入第二個多項式 :" << endl;
    cin >> p2;
    Polynomial sum = p1 + p2;
    Polynomial diff = p1 - p2;
    Polynomial product = p1 * p2;
    cout << "和 :" << sum << endl;
    cout << "差 :" << diff << endl;
    cout << "積 :" << product << endl;
    return 0;
}
```

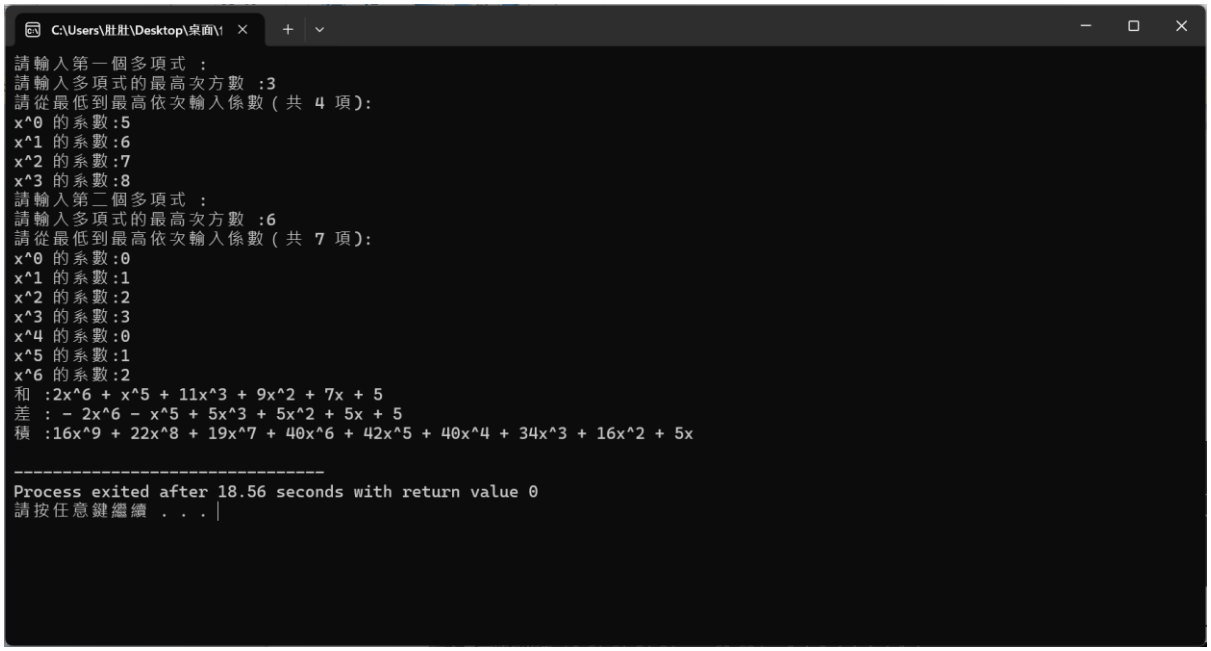
3.效能分析

時間複雜度: O(n m)　n， m 是兩個多項式的次數。

空間複雜度:

```
Process exited after 18.56 seconds with return value 0
請按任意鍵繼續 . . .
```

4.測試與驗證


C:\Users\肚肚\Desktop\桌面\1
請輸入第一個多項式 :
請輸入多項式的最高次方數 :3
請從最低到最高依次輸入係數(共 4 項):
x^0 的系數:5
x^1 的系數:6
x^2 的系數:7
x^3 的系數:8
請輸入第二個多項式 :
請輸入多項式的最高次方數 :6
請從最低到最高依次輸入係數(共 7 項):
x^0 的系數:0
x^1 的系數:1
x^2 的系數:2
x^3 的系數:3
x^4 的系數:0
x^5 的系數:1
x^6 的系數:2
和 :2x^6 + x^5 + 11x^3 + 9x^2 + 7x + 5
差 : - 2x^6 - x^5 + 5x^3 + 5x^2 + 5x + 5
積 :16x^9 + 22x^8 + 19x^7 + 40x^6 + 42x^5 + 40x^4 + 34x^3 + 16x^2 + 5x
----
Process exited after 18.56 seconds with return value 0
請按任意鍵繼續 . . .