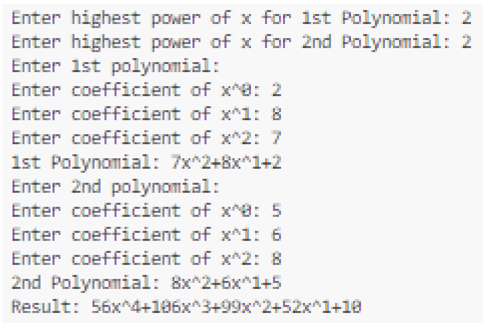
POLYNOMIAL MULTIPLICATION

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

typedef complex<double> cd;
const double pi = acos(-1);

cd *FFT(const cd *S, int N, bool inverse)
{
   cd *A = new cd[N];
   if (N == 1)
   {
      A[0] = S[0];
      return A;
   }
   else
   {
      double power = -2.0 * pi / N * (inverse ? -1 : 1);
      cd w(cos(power), sin(power));
      cd p = 1.0;
      cd *X = new cd[N / 2];
      cd *Y = new cd[N / 2];
      for (int i = 0; i < N / 2; i++)
      {
         X[i] = S[2 * i];
         Y[i] = S[2 * i + 1];
      }
```

```
        cd *B = FFT(X, N / 2, inverse);
        cd *C = FFT(Y, N / 2, inverse);
        for (int k = 0; k < N / 2; k++)
        {
            cd term = p * C[k];
            A[k] = B[k] + term;
            A[k + N / 2] = B[k] - term;
            if (inverse)
            {
                A[k] /= 2;
                A[k + N / 2] /= 2;
            }
            p *= w;
        }
        delete[] X;
        delete[] Y;
        delete[] B;
        delete[] C;
        return A;
    }
}

// Polynomial class
class Polynomial
{
private:
    cd *coefficient;
    int limit;
```

```
public:
    Polynomial(int lim) : limit(lim), coefficient(new cd[lim]) {}
    ~Polynomial()
    {
        delete[] coefficient;
    }
    Polynomial operator*(const Polynomial &other) const;
    inline cd operator[](int index) const
    {
        return coefficient[index];
    }
    int operator()(int x) const;
    friend ostream &operator<<(ostream &os, const Polynomial &obj);
    friend istream &operator>>(istream &is, Polynomial &obj)
    {
        for (int i = 0; i < obj.limit; ++i)
        {
            int coeff;
            cout << "Enter coefficient of x^" << i << ": ";
            is >> coeff;
            obj.coefficient[i] = cd(coeff, 0);
        }
        return is;
    }
};

int Polynomial ::operator()(int x) const
{
    int i = limit;
    int p = coefficient[i].real();
    while (i >= 1)
    {
        p = p * x + coefficient[--i].real();
    }
    return p;
}

Polynomial Polynomial ::operator*(const Polynomial &other) const
{
    int max = limit + other.limit;
    if ((log(max) / log(2) - (int)(log(max) / log(2))) > 0.0)
    {
        int temp = log(max) / log(2);
        max = 2;
        max = max << temp;
    }
    cd *fA = new cd[max];
    for (int i = 0; i < max; ++i)
    {
        if (i < limit)
            fA[i] = coefficient[i];
        else
            fA[i] = {0.0, 0.0};
    }
    fA = FFT(fA, max, false);
    cd *fB = new cd[max];
    for (int i = 0; i < max; ++i)
    {
        if (i < other.limit)
            fB[i] = other.coefficient[i];
        else
            fB[i] = {0.0, 0.0};
    }
    fB = FFT(fB, max, false);
    for (int i = 0; i < max; ++i)
    {
        fA[i] *= fB[i];
    }
    fA = FFT(fA, max, true);
    Polynomial result(max);
    for (int i = 0; i < result.limit; ++i)
    {
        result.coefficient[i] = fA[i];
    }
    delete[] fA;
    delete[] fB;
    return result;
}

ostream &operator<<(ostream &os, const Polynomial &obj)
{
    bool flag = false;
    int i = obj.limit - 1;
    while ((int)obj[i].real() == 0)
        --i;
    while (i >= 0)
    {
        double val = obj[i].real();
        if (val == 1 && i != 0)
            os << "x^" << i;
        else if (val == -1 && i != 0)
            os << "-x^" << i;
        else if (i == 0)
            os << val;
        else
            os << val << "x^" << i;
        if (i != 0 && obj[i - 1].real() >= 0)
            cout << "+";
        --i;
    }
    return os;
}

int main()
{
    int N, N2;
    cout << "Enter highest power of x for 1st Polynomial: ";
    cin >> N;
    cout << "Enter highest power of x for 2nd Polynomial: ";
    cin >> N2;
    Polynomial P1(N + 1), P2(N2 + 1);
    cout << "Enter 1st polynomial: " << endl;
    cin >> P1;
    cout << "1st Polynomial: ";
    cout << P1 << endl;
    cout << "Enter 2nd polynomial: " << endl;
    cin >> P2;
    cout << "2nd Polynomial: ";
    cout << P2 << endl;
    Polynomial Result = P1 * P2;
    cout << "Result: " << Result << endl;
    return 0;
}
```

OUTPUT



```
Enter highest power of x for 1st Polynomial: 2
Enter highest power of x for 2nd Polynomial: 2
Enter 1st polynomial:
Enter coefficient of x^0: 2
Enter coefficient of x^1: 8
Enter coefficient of x^2: 7
1st Polynomial: 7x^2+8x^1+2
Enter 2nd polynomial:
Enter coefficient of x^0: 5
Enter coefficient of x^1: 6
Enter coefficient of x^2: 8
2nd Polynomial: 8x^2+6x^1+5
Result: 56x^4+106x^3+99x^2+52x^1+10
```