IT300 Assignment 3

NAME: SUYASH CHINTAWAR
ROLL NO.: 191IT109
TOPIC: FFT

Q1. Implement the recursive FFT, and iterative FFT algorithms and test the output of your program by comparing it with the in-built library function. Compute the DFT of the input vector (0,2,3, -1,4,5,7,9) using algorithms you implemented.

SOLUTION:

Code for recursive FFT:

```
#include<bits/stdc++.h>
using namespace std;
#define pi acos(-1)
typedef complex<double> cd;

void fft(vector<cd> &coef, bool inverse) {
    int n = coef.size();
    if (n == 1) return;

    vector<cd> left(n/2),right(n/2);

    for(int i=0;i<(n/2);i++)
    {
     left[i]=coef[2*i];
     right[i]=coef[2*i+1];
    }

    fft(left, inverse);
    fft(right, inverse);

    double angle = 2*pi/n;
    if(inverse==false) angle*=-1;

    cd w(1), wn(cos(angle),sin(angle));

    for(int i=0;i<(n/2);i++)
    {
     coef[i]=left[i]+w*right[i];
     coef[i+n/2]=left[i]-w*right[i];
     if (inverse == true)
     {
            coef[i]/=2;
            coef[i+n/2]/=2;
        }
        w*=wn;
    }
}

int main()
{
     int n;
    cout<<"Enter size of array:";
    cin>>n;
    vector<int> v(n);
    cout<<"Enter the "<<n<<" elements (space separated integers): \n";
    for(int i=0;i<n;i++)
    {
        cin>>v[i];
    }
    vector<cd> coef(v.begin(),v.end());
    fft(coef,false);
    cout<<"\nDFT of the input array is:\n";
    for (int i=0;i<n;i++)
    {
        if(coef[i].imag()>=0)
            cout<<coef[i].real()<<"+"<<coef[i].imag()<<"i"<<endl;
        else cout<<coef[i].real()<<coef[i].imag()<<"i"<<endl;
    }
}
```

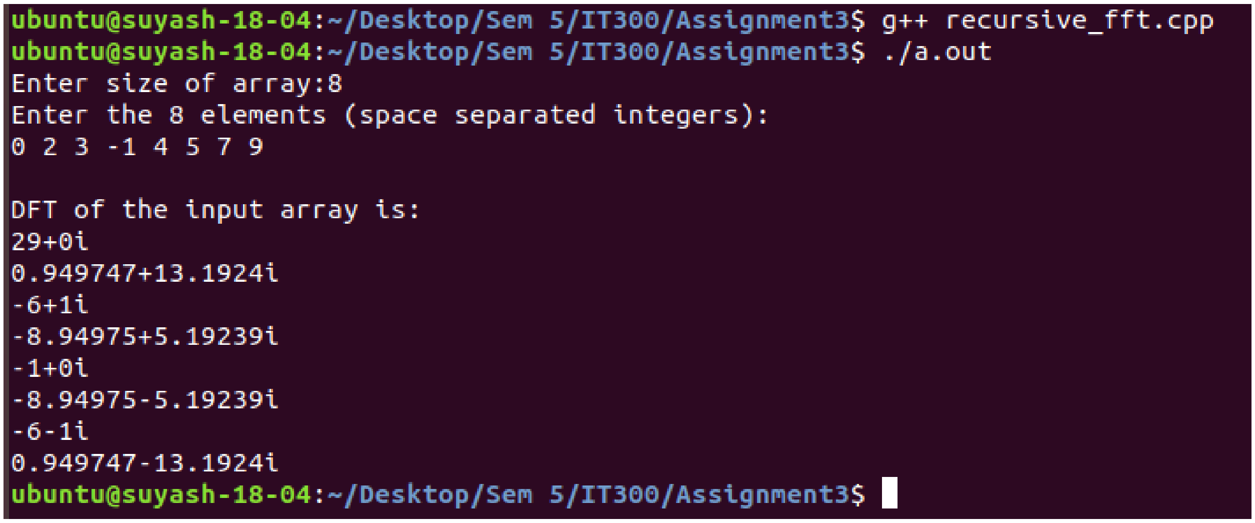
Code for iterative FFT:

```
#include<bits/stdc++.h>
using namespace std;
#define pi acos(-1)
#define mod 1000000007
typedef complex<double> cd;
```

```
int pow(int a,int b)
{
    int res=1;
    while(b>0)
    {
        if(b%2==1) res=(res*a)%mod;
        a=(a*a)%mod;
        b/=2;
    }
    return res;
}

int reverse(int num, int logn)
{
    int res = 0;
    for(int i=0;i<logn;i++)
    {
        if(num&pow(2,i)) res|=pow(2,logn-i-1);
    }
    return res;
}

void fft(vector<cd> &coef, bool inverse)
{
    int n = coef.size();
    int logn = log2(n);

    for(int i=0;i<n;i++)
    {
        if(i<reverse(i,logn))
     {
          int idx = reverse(i,logn);
          cd temp = coef[idx];
          coef[idx]=coef[i];
          coef[i]=temp;
     }
    }

    for(int l=2;l<=n;l*=2)
    {
        double angle = 2*pi/l;
     if(inverse==false) angle*=-1;
        cd wl(cos(angle),sin(angle));
        for(int i=0;i<n;i+=l)
        {
            cd w(1);
            for(int j=0;j<l/2;j++)
            {
                cd u = coef[i+j];
                cd v = coef[i+j+l/2]*w;
                coef[i+j]=u+v;
                coef[i+j+l/2]=u-v;
                w*=wl;
            }
        }
    }

    if(inverse==true)
    {
        for(auto x:coef) x /= n;
    }
}

int main()
{
     int n;
     cout<<"Enter size of array:";
     cin>>n;
     vector<int> v(n);
     cout<<"Enter the "<<n<<" elements (space separated integers): \n";
     for(int i=0;i<n;i++)
     {
          cin>>v[i];
     }
     vector<cd> coef(v.begin(),v.end());
    fft(coef,false);
    cout<<"\nDFT of the input array is:\n";
    for (int i=0;i<n;i++)
     {
          if(coef[i].imag()>=0)
              cout<<coef[i].real()<<"+"<<coef[i].imag()<<"i"<<endl;
          else cout<<coef[i].real()<<coef[i].imag()<<"i"<<endl;
     }
}
```

<u>Output on the given input vector:</u>

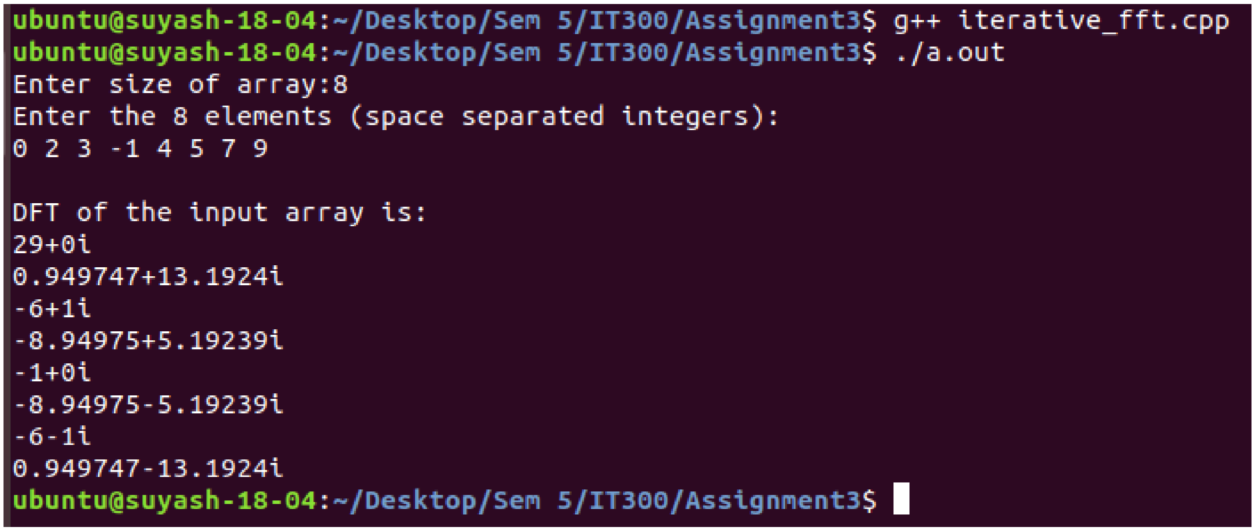
(a) Recursive FFT:



```
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ recursive_fft.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter size of array:8
Enter the 8 elements (space separated integers):
0 2 3 -1 4 5 7 9

DFT of the input array is:
29+0i
0.949747+13.1924i
-6+1i
-8.94975+5.19239i
-1+0i
-8.94975-5.19239i
-6-1i
0.949747-13.1924i
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ 
```

(continued...)

(b) Iterative FFT:



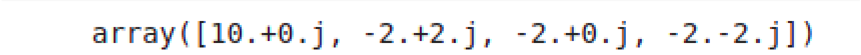
```
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ iterative_fft.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter size of array:8
Enter the 8 elements (space separated integers):
0 2 3 -1 4 5 7 9

DFT of the input array is:
29+0i
0.949747+13.1924i
-6+1i
-8.94975+5.19239i
-1+0i
-8.94975-5.19239i
-6-1i
0.949747-13.1924i
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ 
```

<u>TESTING ON SOME OTHER INPUTS:</u>

<u>NOTE:</u> The inbuilt library is only used to verify the outputs obtained by me. Screenshots of recursive and iterative FFT and the output obtained by using the in-built library has been attached for each of the 2 test cases.

<u>Test Case 1:</u> Input vector is (1,2,3,4)

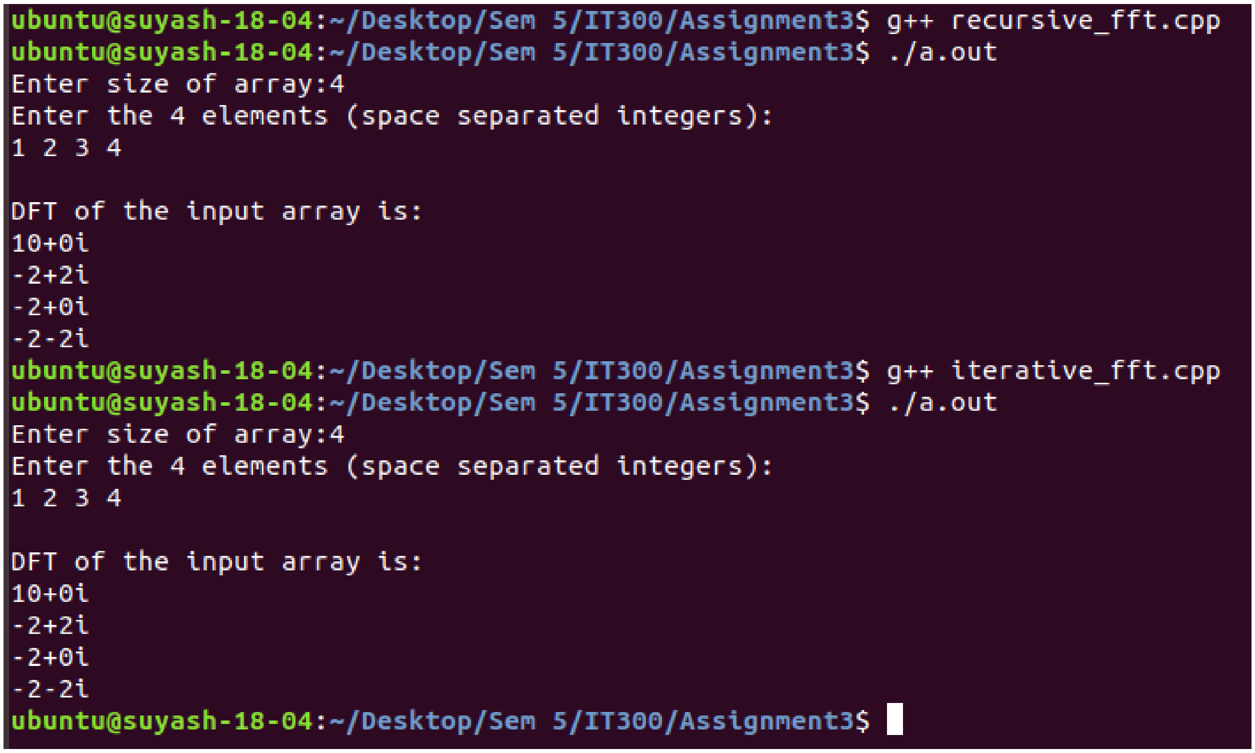
<u>Expected Output:</u>



```
array([10.+0.j, -2.+2.j, -2.+0.j, -2.-2.j])
```

(continued...)

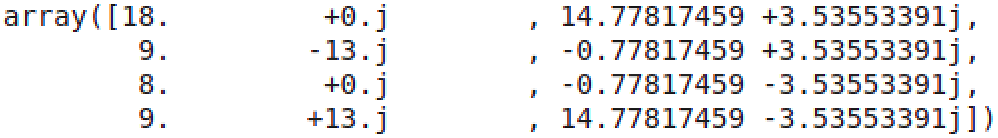
Obtained Output:



```
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ recursive_fft.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter size of array:4
Enter the 4 elements (space separated integers):
1 2 3 4

DFT of the input array is:
10+0i
-2+2i
-2+0i
-2-2i
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ iterative_fft.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter size of array:4
Enter the 4 elements (space separated integers):
1 2 3 4

DFT of the input array is:
10+0i
-2+2i
-2+0i
-2-2i
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ 
```

Test Case 2: Input vector is (9,6,1,-6,2,3,1,2)

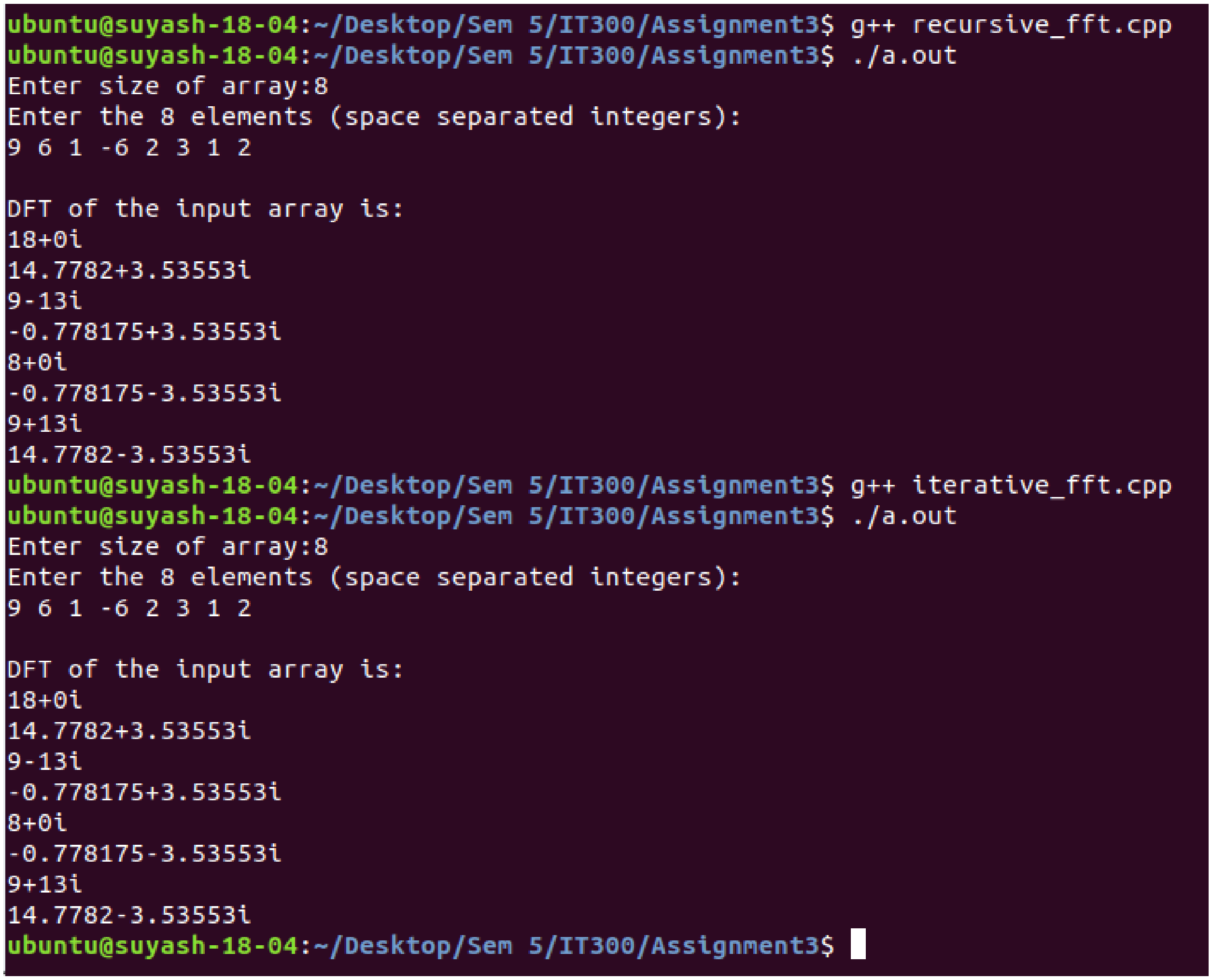
Expected Output:



```
array([18.        +0.j        , 14.77817459 +3.53553391j,
        9.       -13.j        , -0.77817459 +3.53553391j,
        8.        +0.j        , -0.77817459 -3.53553391j,
        9.       +13.j        , 14.77817459 -3.53553391j])
```

(continued...)

Obtained Output:



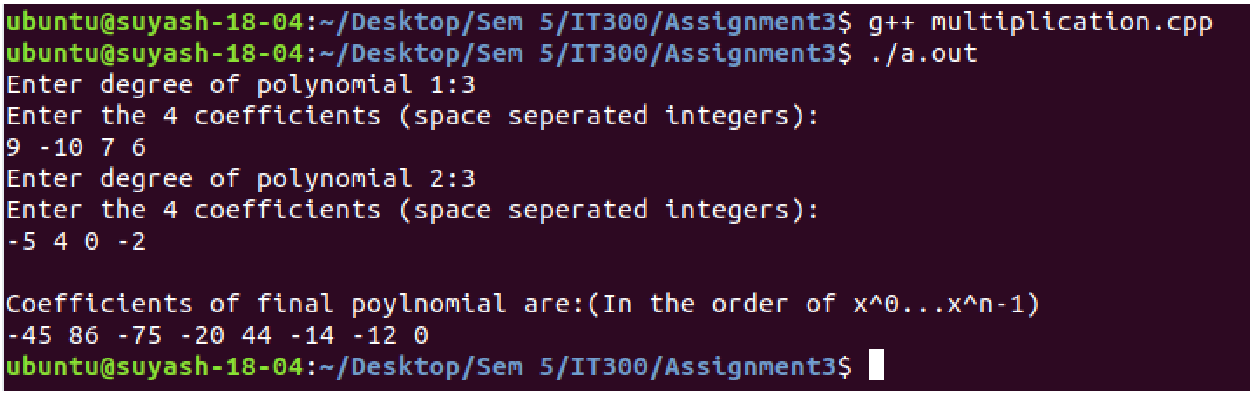
```
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ recursive_fft.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter size of array:8
Enter the 8 elements (space separated integers):
9 6 1 -6 2 3 1 2

DFT of the input array is:
18+0i
14.7782+3.53553i
9-13i
-0.778175+3.53553i
8+0i
-0.778175-3.53553i
9+13i
14.7782-3.53553i
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ iterative_fft.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter size of array:8
Enter the 8 elements (space separated integers):
9 6 1 -6 2 3 1 2

DFT of the input array is:
18+0i
14.7782+3.53553i
9-13i
-0.778175+3.53553i
8+0i
-0.778175-3.53553i
9+13i
14.7782-3.53553i
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ 
```

Q2. Implement a program to demonstrate how FFT can be used to multiply the polynomials. Show the result of multiplication of following two polynomials.
A(x)= 6x^3 + 7x^2 - 10x+ 9 and B(x)= -2x^3 + 4x-5
SOLUTION:

```
#include<bits/stdc++.h>
using namespace std;
#define pi acos(-1)
typedef complex<double> cd;

void fft(vector<cd> &coef, bool inverse) {
    int n = coef.size();
    if (n == 1) return;

    vector<cd> left(n/2),right(n/2);

    for(int i=0;2*i<n;i++)
    {
     left[i]=coef[2*i];
     right[i]=coef[2*i+1];
    }

    fft(left, inverse);
    fft(right, inverse);

    double angle = 2*pi/n;
    if(inverse==false) angle*=-1;

    cd w(1), wn(cos(angle),sin(angle));

    for(int i=0;2*i<n;i++)
    {
     coef[i]=left[i]+w*right[i];
     coef[i+n/2]=left[i]-w*right[i];
     if (inverse == true)
     {
            coef[i]/=2;
            coef[i+n/2]/=2;
        }
        w*=wn;
    }
}

int main()
{
     int n1,n2;
    cout<<"Enter degree of polynomial 1:";
    cin>>n1;
    vector<int> p1(n1+1);
    cout<<"Enter the "<<n1+1<<" coefficients (space seperated integers):
\n";
    for(int i=0;i<n1+1;i++)
    {
        cin>>p1[i];
    }
    cout<<"Enter degree of polynomial 2:";
    cin>>n2;
    vector<int> p2(n2+1);
    cout<<"Enter the "<<n2+1<<" coefficients (space seperated integers):
\n";
    for(int i=0;i<n2+1;i++)
    {
        cin>>p2[i];
    }

    int n = 1;
    while (n < p1.size()+p2.size())
        n <<= 1;

    vector<cd> coef_p1(p1.begin(),p1.end());
    vector<cd> coef_p2(p2.begin(),p2.end());
    coef_p1.resize(n);
    coef_p2.resize(n);
    fft(coef_p1,false);
    fft(coef_p2,false);

    for (int i = 0; i < n; i++)
        coef_p1[i] *= coef_p2[i];
    fft(coef_p1, true);

    vector<int> result(n);
    for (int i = 0; i < n; i++)
        result[i] = round(coef_p1[i].real());

    cout<<"\nCoefficients of final poylnomial are:";
    cout<<"(In the order of x^0...x^n-1)\n";

    for (int i=0;i<result.size();i++)
    {
        cout<<result[i]<<' ';
    }
    cout<<endl;
}
```

Output:

The coefficients are supplied in the order of powers of x from 0 to n-1. Same pattern is followed in printing the output coefficients.



```
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ g++ multiplication.cpp
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$ ./a.out
Enter degree of polynomial 1:3
Enter the 4 coefficients (space seperated integers):
9 -10 7 6
Enter degree of polynomial 2:3
Enter the 4 coefficients (space seperated integers):
-5 4 0 -2

Coefficients of final poylnomial are:(In the order of x^0...x^n-1)
-45 86 -75 -20 44 -14 -12 0
ubuntu@suyash-18-04:~/Desktop/Sem 5/IT300/Assignment3$
```

THANK YOU