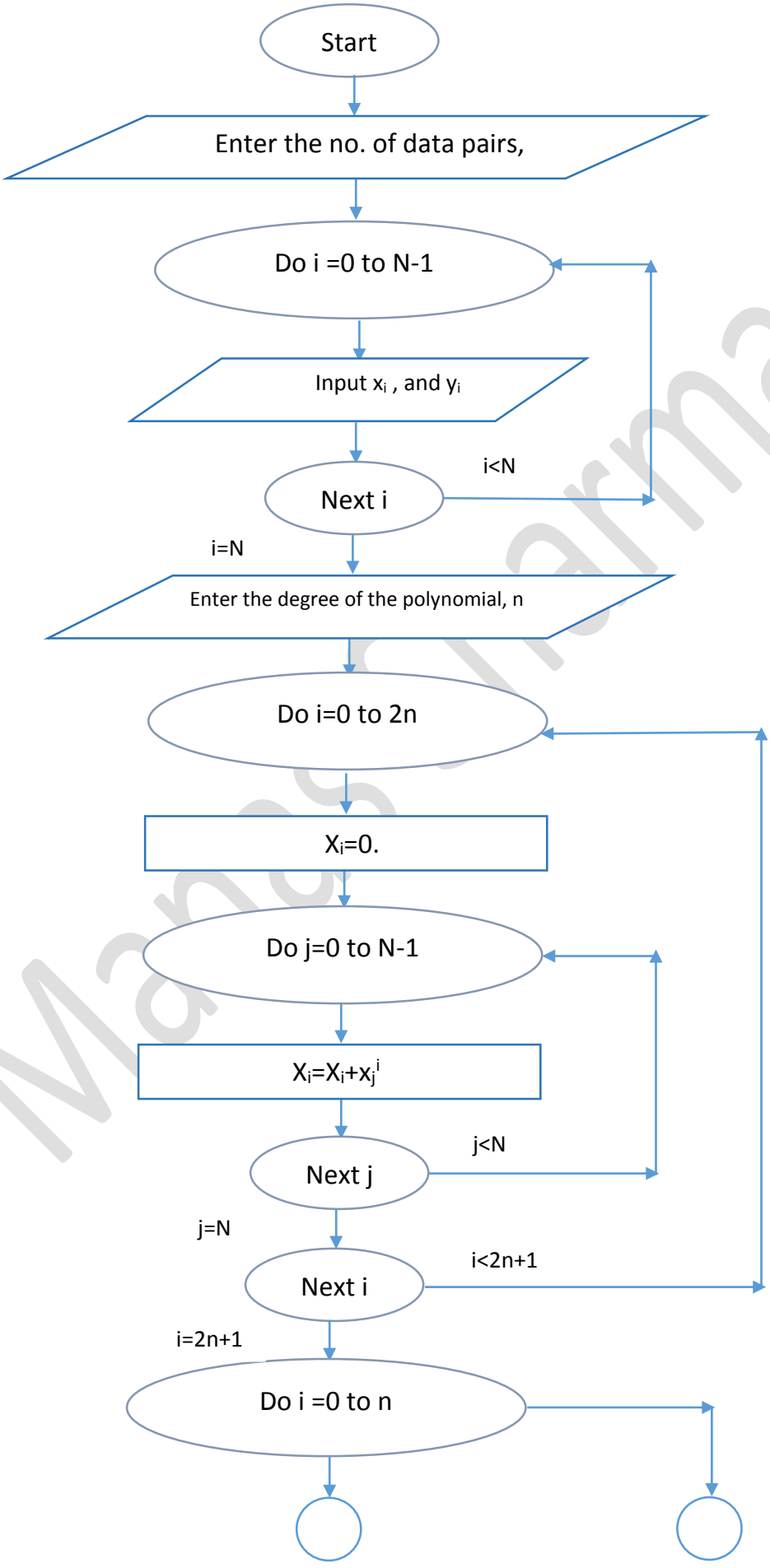
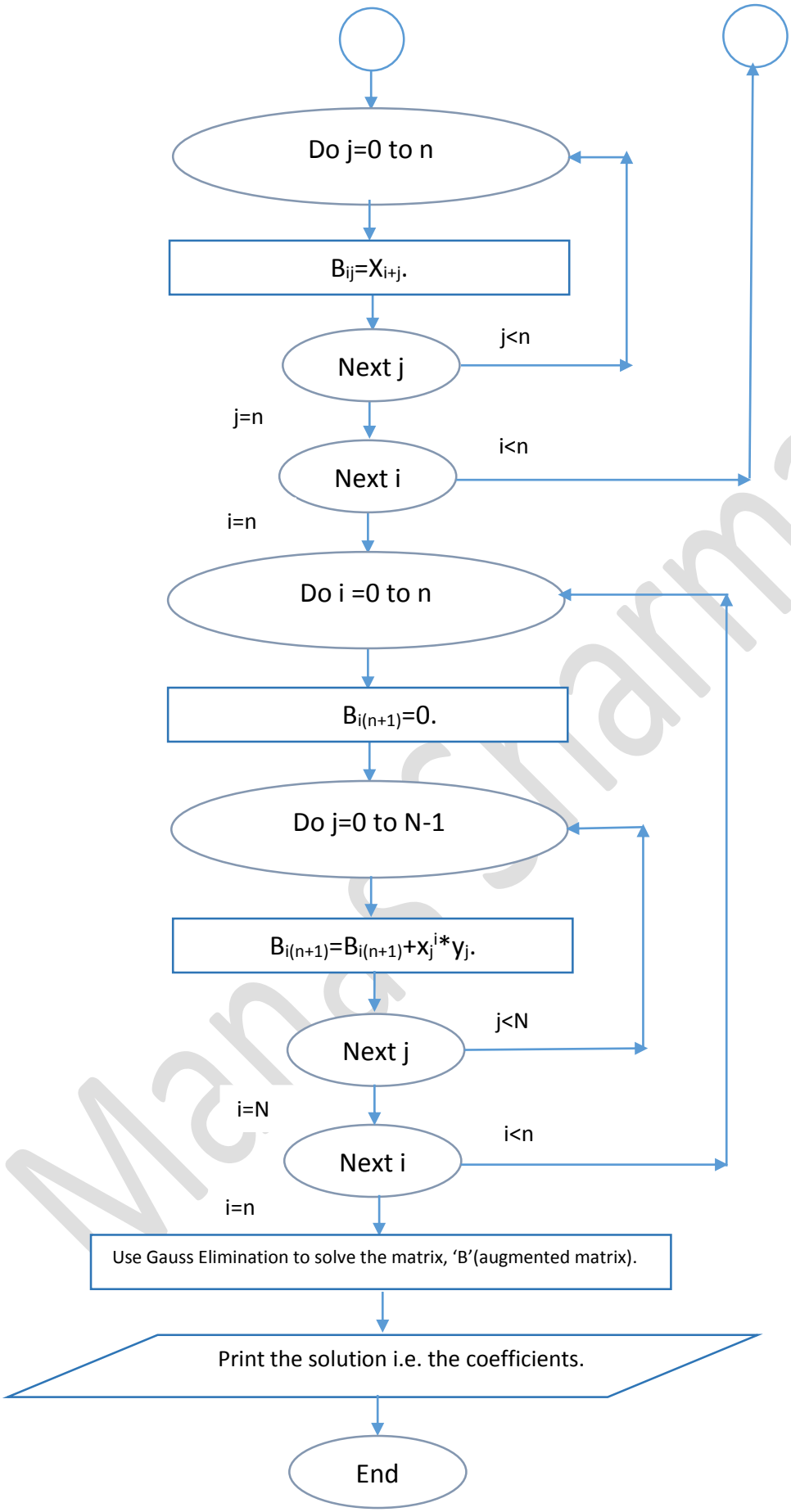
Aim: To find the best polynomial fit for a given set of data.

Algorithm:

1. Enter the no. of data pairs, N.
2. Begin For i=0 to N-1
   Input x_i , and y_i.
   End for.
3. Enter the degree of polynomial to be used, n.
4. Create an array, 'X' of size, (2n+1).
5. Begin For i=0 to 2n
   X_i=0.
   For j=0 to N-1
   $X_i=X_i+x_j^i$
   End For j.
   End for i.
6. Create a matrix $B_{(n+1)x(n+2)}$ which is the Normal matrix containing the Normal equations.
7. Create an array 'a' of size (n+1), that will store the coefficients.
8. For i=0 to n
   For j=0 to n
   $B_{ij}=X_{i+j}$.
   End For j.
   End For i.
9. For i=0 to n
   $B_{i(n+1)}$=0.
   For j=0 to N-1.
   $B_{i(n+1)}=B_{i(n+1)}+x_j^i*y_j$.
   End For j.
   End For i.
10. Use Gauss Elimination to solve the matrix, 'B'(augmented matrix).
11. Print the solution obtained after Gaussian Elimination which are the coefficients of the fitting Polynomial.
12. End.

Flow Chart:





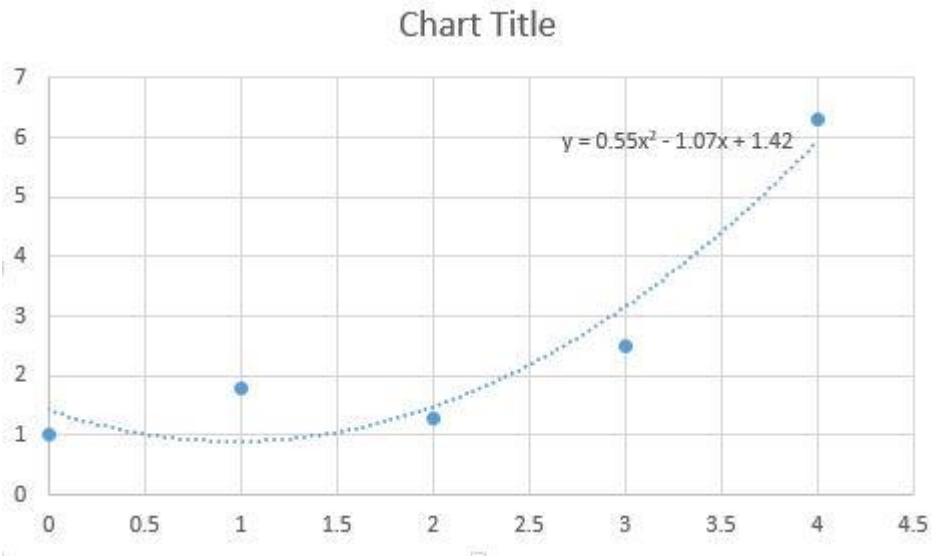
Program:

```
//Polynomial Fit
#include<iostream>
#include<iomanip>
#include<cmath>
using namespace std;
int main()
{
    int i,j,k,n,N;
    cout.precision(4);                          //set precision
    cout.setf(ios::fixed);
    cout<<"\nEnter the no. of data pairs to be entered:\n";       //To
find the size of arrays that will store x,y, and z values
    cin>>N;
    double x[N],y[N];
    cout<<"\nEnter the x-axis values:\n";               //Input x-values
    for (i=0;i<N;i++)
        cin>>x[i];
    cout<<"\nEnter the y-axis values:\n";               //Input y-values
    for (i=0;i<N;i++)
        cin>>y[i];
    cout<<"\nWhat degree of Polynomial do you want to use for the fit?\n";
    cin>>n;                                 // n is the degree of Polynomial
    double X[2*n+1];                        //Array that will store the
values of sigma(xi),sigma(xi^2),sigma(xi^3)....sigma(xi^2n)
    for (i=0;i<2*n+1;i++)
    {
        X[i]=0;
        for (j=0;j<N;j++)
            X[i]=X[i]+pow(x[j],i);        //consecutive positions of the
array will store N,sigma(xi),sigma(xi^2),sigma(xi^3)....sigma(xi^2n)
    }
    double B[n+1][n+2],a[n+1];              //B is the Normal
matrix(augmented) that will store the equations, 'a' is for value of the
final coefficients
    for (i=0;i<=n;i++)
        for (j=0;j<=n;j++)
            B[i][j]=X[i+j];            //Build the Normal matrix by storing
the corresponding coefficients at the right positions except the last
column of the matrix
    double Y[n+1];                     //Array to store the values of
sigma(yi),sigma(xi*yi),sigma(xi^2*yi)...sigma(xi^n*yi)
    for (i=0;i<n+1;i++)
    {
        Y[i]=0;
        for (j=0;j<N;j++)
        Y[i]=Y[i]+pow(x[j],i)*y[j];        //consecutive positions will
store sigma(yi),sigma(xi*yi),sigma(xi^2*yi)...sigma(xi^n*yi)
    }
    for (i=0;i<=n;i++)
        B[i][n+1]=Y[i];                //load the values of Y as the last
column of B(Normal Matrix but augmented)
    n=n+1;                //n is made n+1 because the Gaussian Elimination
part below was for n equations, but here n is the degree of polynomial and
for n degree we get n+1 equations
    cout<<"\nThe Normal(Augmented Matrix) is as follows:\n";
    for (i=0;i<n;i++)            //print the Normal-augmented matrix
    {
```

```
        for (j=0;j<=n;j++)
            cout<<B[i][j]<<setw(16);
        cout<<"\n";
    }
    for (i=0;i<n;i++)                    //From now Gaussian Elimination
starts(can be ignored) to solve the set of linear equations (Pivotisation)
        for (k=i+1;k<n;k++)
            if (B[i][i]<B[k][i])
                for (j=0;j<=n;j++)
                {
                    double temp=B[i][j];
                    B[i][j]=B[k][j];
                    B[k][j]=temp;
                }

    for (i=0;i<n-1;i++)            //loop to perform the gauss elimination
        for (k=i+1;k<n;k++)
            {
                double t=B[k][i]/B[i][i];
                for (j=0;j<=n;j++)
                    B[k][j]=B[k][j]-t*B[i][j];    //make the elements below
the pivot elements equal to zero or elimnate the variables
            }
    for (i=n-1;i>=0;i--)                 //back-substitution
    {                         //x is an array whose values correspond to the
values of x,y,z..
        a[i]=B[i][n];                //make the variable to be calculated
equal to the rhs of the last equation
        for (j=0;j<n;j++)
            if (j!=i)            //then subtract all the lhs values except
the coefficient of the variable whose value
is being calculated
                a[i]=a[i]-B[i][j]*a[j];
        a[i]=a[i]/B[i][i];            //now finally divide the rhs by the
coefficient of the variable to be calculated
    }
    cout<<"\nThe values of the coefficients are as follows:\n";
    for (i=0;i<n;i++)
        cout<<"x^"<<i<<"="<<a[i]<<endl;            // Print the values of
x^0,x^1,x^2,x^3,....
    cout<<"\nHence the fitted Polynomial is given by:\ny=";
    for (i=0;i<n;i++)
        cout<<" + ("<<a[i]<<")"<<"x^"<<i;
    cout<<"\n";
    return 0;
}//output attached as .jpg
```

Output:

```
Enter the no. of data pairs to be entered:
5

Enter the x-axis values:
0       1       2       3       4

Enter the y-axis values:
1       1.8     1.3     2.5     6.3

What degree of Polynomial do you want to use for the fit?
2

The Normal(Augmented Matrix) is as follows:
5.0000          10.0000          30.0000          12.9000
10.0000          30.0000         100.0000          37.1000
30.0000         100.0000         354.0000         130.3000

The values of the coefficients are as follows:
x^0=1.4200
x^1=-1.0700
x^2=0.5500

Hence the fitted Polynomial is given by:
y= + (1.4200)x^0 + (-1.0700)x^1 + (0.5500)x^2
```

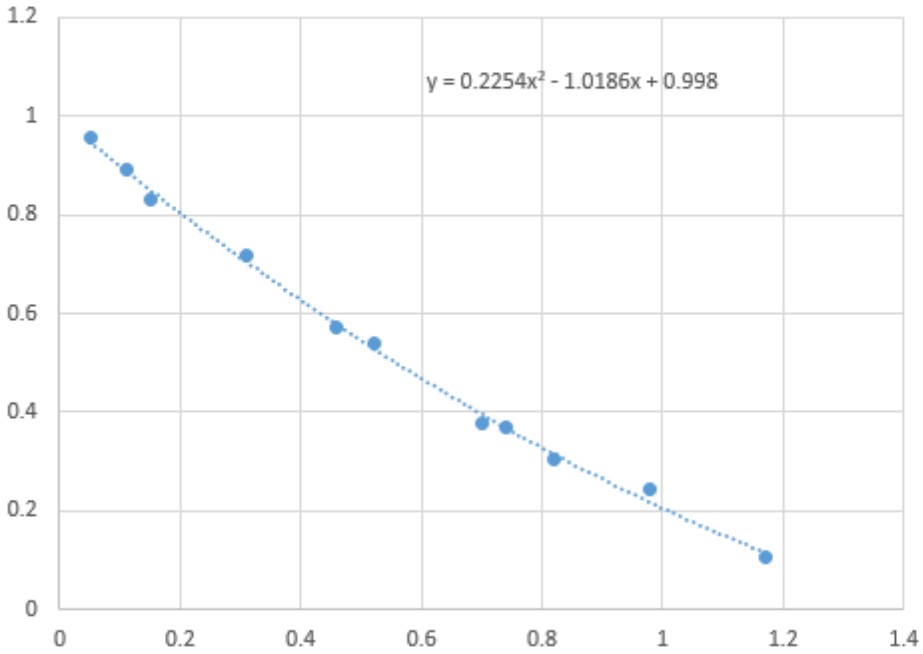


```
Enter the no. of data pairs to be entered:
11

Enter the x-axis values:
.05     .11     .15     .31     .46     .52     .70     .74     .82     .98     1.171

Enter the y-axis values:
0.956
.890    .832    .717    .571    .539    .378    .370    .306    .242    .104

What degree of Polynomial do you want to use for the fit?
2

The Normal(Augmented Matrix) is as follows:
11.0000          6.0110          4.6568          5.9050
6.0110          4.6568          4.1191          2.1840
4.6568          4.1191          3.9225          1.3360

The values of the coefficients are as follows:
x^0=0.9980
x^1=-1.0186
x^2=0.2254

Hence the fitted Polynomial is given by:
y= + (0.9980)x^0 + (-1.0186)x^1 + (0.2254)x^2
```

The result is a_0 = 0.998, a_1=-1.018, a_2=0.225, so the Least Squares Method gives

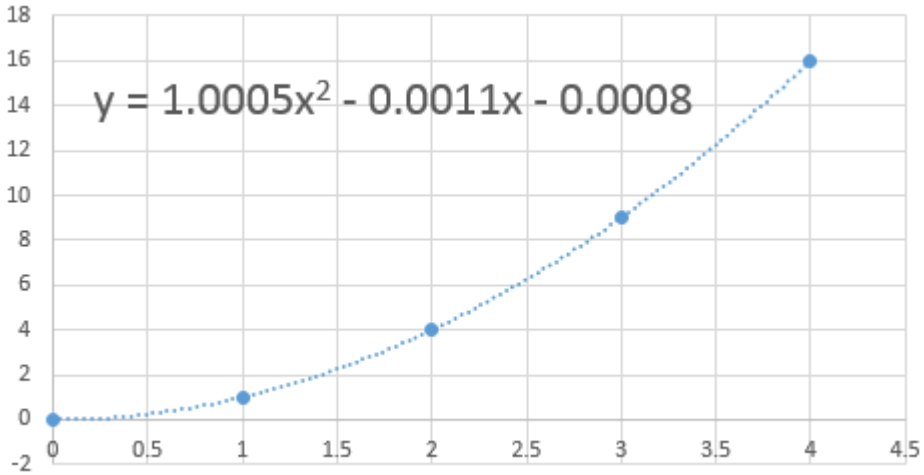
y= 0.998 – 1.018x +0.225x^2 (Answer)

We can verify this result using Excel to find the fitting polynomial:



As we can see the answers are almost identical.

```
Enter the no. of data pairs to be entered:
5

Enter the x-axis values:
0       1       2       3       4

Enter the y-axis values:
0       0.998   3.996   9.005   16.001

What degree of Polynomial do you want to use for the fit?
2

The Normal(Augmented Matrix) is as follows:
5.0000          10.0000          30.0000          30.0000
10.0000          30.0000         100.0000         100.0090
30.0000         100.0000         354.0000         354.0430

The values of the coefficients are as follows:
x^0=-0.0008
x^1=-0.0011
x^2=1.0005

Hence the fitted Polynomial is given by:
y= + (-0.0008)x^0 + (-0.0011)x^1 + (1.0005)x^2
```



```
Enter the no. of data pairs to be entered:
5

Enter the x-axis values:
0       1       2       3       4

Enter the y-axis values:
1       1.8     1.3     2.5     6.3

What degree of Polynomial do you want to use for the fit?
2

The Normal(Augmented Matrix) is as follows:
5.0000          10.0000          30.0000          12.9000
10.0000          30.0000         100.0000          37.1000
30.0000         100.0000         354.0000         130.3000

The values of the coefficients are as follows:
x^0=1.4200
x^1=-1.0700
x^2=0.5500

Hence the fitted Polynomial is given by:
y= + (1.4200)x^0 + (-1.0700)x^1 + (0.5500)x^2
```