MA 322: Lab Assignment #4

Due on Sunday, August 30, 2015

Jiten Chandra Kalita

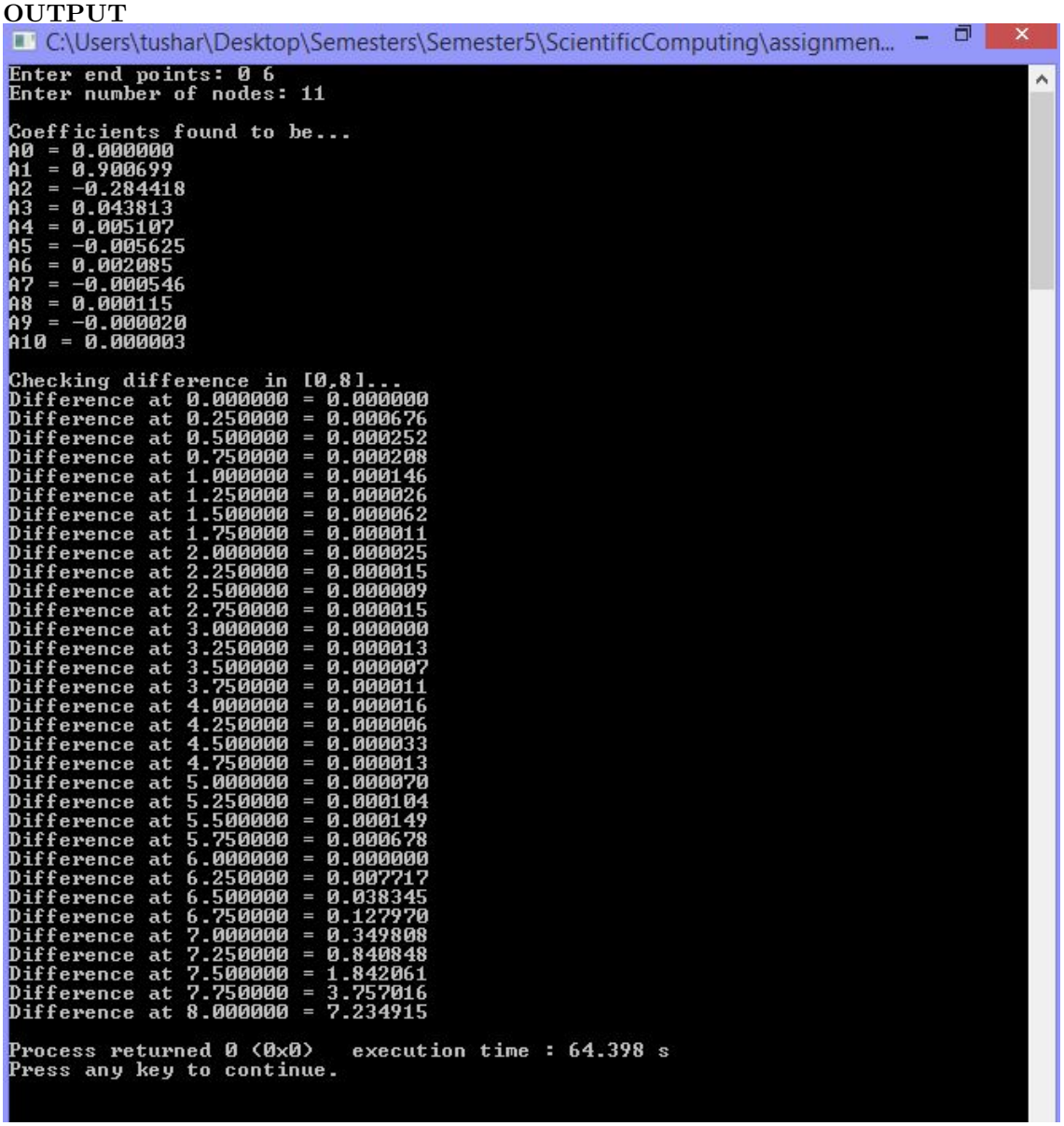
Tushar Sircar - 130123038

PROBLEM 1

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

double getF(double x)
{
    return atan(x);
}
double getP(double p,double a[],double x[],int noOfPoints)
{
    double mul = 1;
    double ans = 0;
    for(int i=0; i<noOfPoints; i++)
    {
        ans += a[i]*mul;
        mul = mul * (p - x[i]);
    }
    return ans;
}

int main()
{
    double left,right; //end points
    int noOfPoints; //number of equally spaced points
    double h; //spacing between consequetive nodes

    cout<<"Enter end points: ";
    cin>>left>>right;
    cout<<"Enter number of nodes: ";
    cin>>noOfPoints;
    h = (right-left)/((double)(noOfPoints-1));

    double x[noOfPoints]; //nodes of the interpolating polynomial
    double dividedDiff[noOfPoints][noOfPoints];


    //store all nodes
    x[0] = left;
    for(int i=1; i<noOfPoints; i++)
        x[i] = x[i-1] + h;

    for(int i=0; i<noOfPoints; i++)
        dividedDiff[i][i] = getF(x[i]);

    //calculate divided differences
    for(int i=1; i<noOfPoints; i++)
    {
        int r = 0;
        int c = i;
        while(c<noOfPoints)
```

```
        {
            dividedDiff[r][c] = (dividedDiff[r+1][c] - dividedDiff[r][c-1])/(x[c] - x[r]);
            r++;
            c++;
        }
    }

    //print coefficients
    double a[noOfPoints];
    cout<<"\nCoefficients found to be...\n";
    for(int i=0; i<noOfPoints; i++)
    {
        a[i] = dividedDiff[0][i];
        printf("A%d = %f\n",i,dividedDiff[0][i]);
    }

    cout<<"\nChecking difference in [0,8]...\n";
    h = (double)8/(double)32;
    for(int i=0; i<33; i++)
    {
        double p = i*h;
        printf("Difference at %f = %f\n",p,abs(getF(p) - getP(p,a,x,noOfPoints)));
    }



    return 0;
}
```

OUTPUT



CONCLUSION

(a)Coefficients of the 10 degree interpolating polynomial are denoted by Ai, i = 0 to 11.

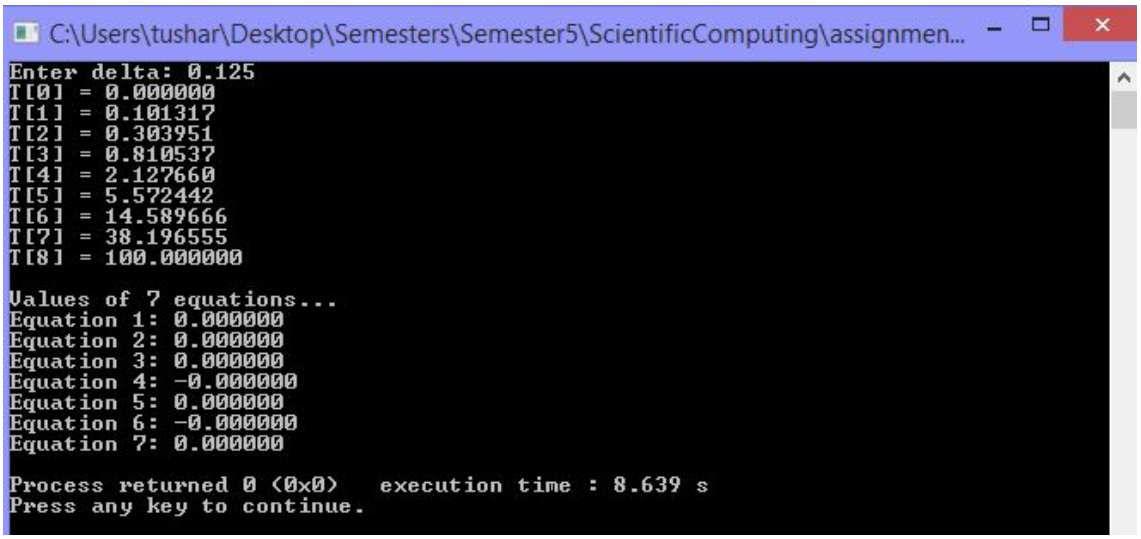
(b)Difference between the actual and interpolating polynomial is very small (around 0.0001) in [0,6] but from [6,8] the difference is large.(around 0.0001) in [0,6] but

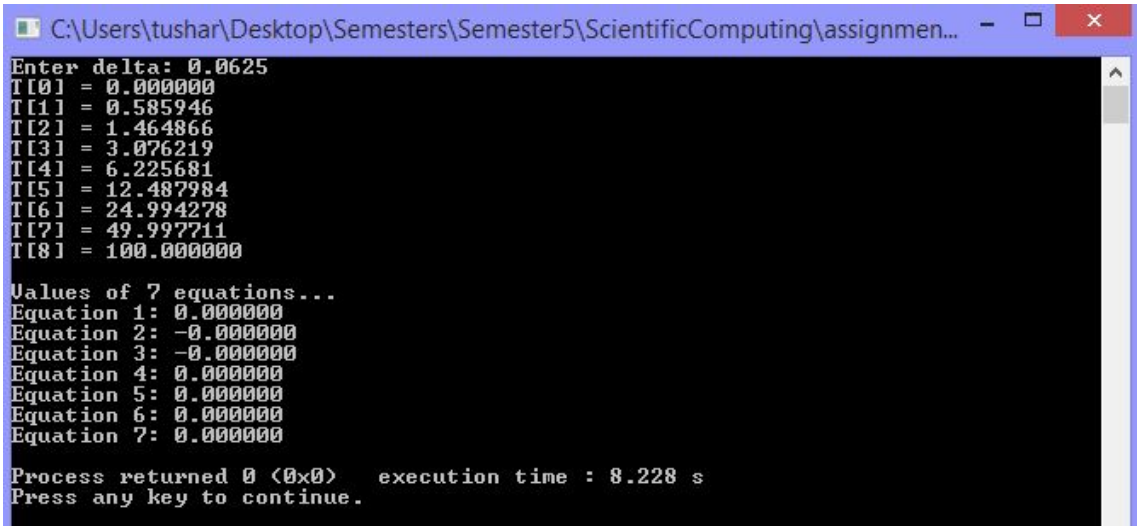
PROBLEM 2

```
#include<iostream>
#include<stdio.h>
using namespace std;

int main()
{
    double alpha = 2.0;
    double delta;
    cout<<"Enter delta: ";
    cin>>delta;
    double t[9];
    double f[9];
    t[0] = f[0] = 0;
    t[8] = f[8] = 100;

    //initialising the matrix
    double a[8],b[8],c[8];
    for(int i=1; i<=7; i++)
    {
        a[i] = c[i] = 1;
        b[i] = -(2 + (2*alpha*alpha*delta));
        f[i] = 0;
    }

    //forward sweep
    f[1] = f[1] - (a[1]*t[0]);
    f[7] = f[7] - (c[7]*t[8]);

    b[1] = b[1]; //unchanged
    //c's are unchanged
    for(int i=2; i<=7; i++)
    {
        b[i] = b[i] - ((c[i-1]*a[i])/b[i-1]);
        f[i] = f[i] - ((f[i-1]*a[i])/b[i-1]);
    }

    //find solutions using backward sweep
    t[7] = f[7]/b[7];
    for(int i=6; i>=1; i--)
        t[i] = (f[i] - (c[i]*t[i+1]))/b[i];

    cout<"Solution...\n";
    for(int i=0; i<9; i++)
        printf("T[%d] = %f\n",i,t[i]);

    cout<<"\nValues of 7 equations...\n";
    for(int i=1; i<=7; i++)
        printf("Equation %d: %f\n",i,(1*t[i-1]) + ((-2 - 2*alpha*alpha*delta)*t[i]) + (1*t[i+1]));
```

```
    return 0;
}
```

OUTPUT



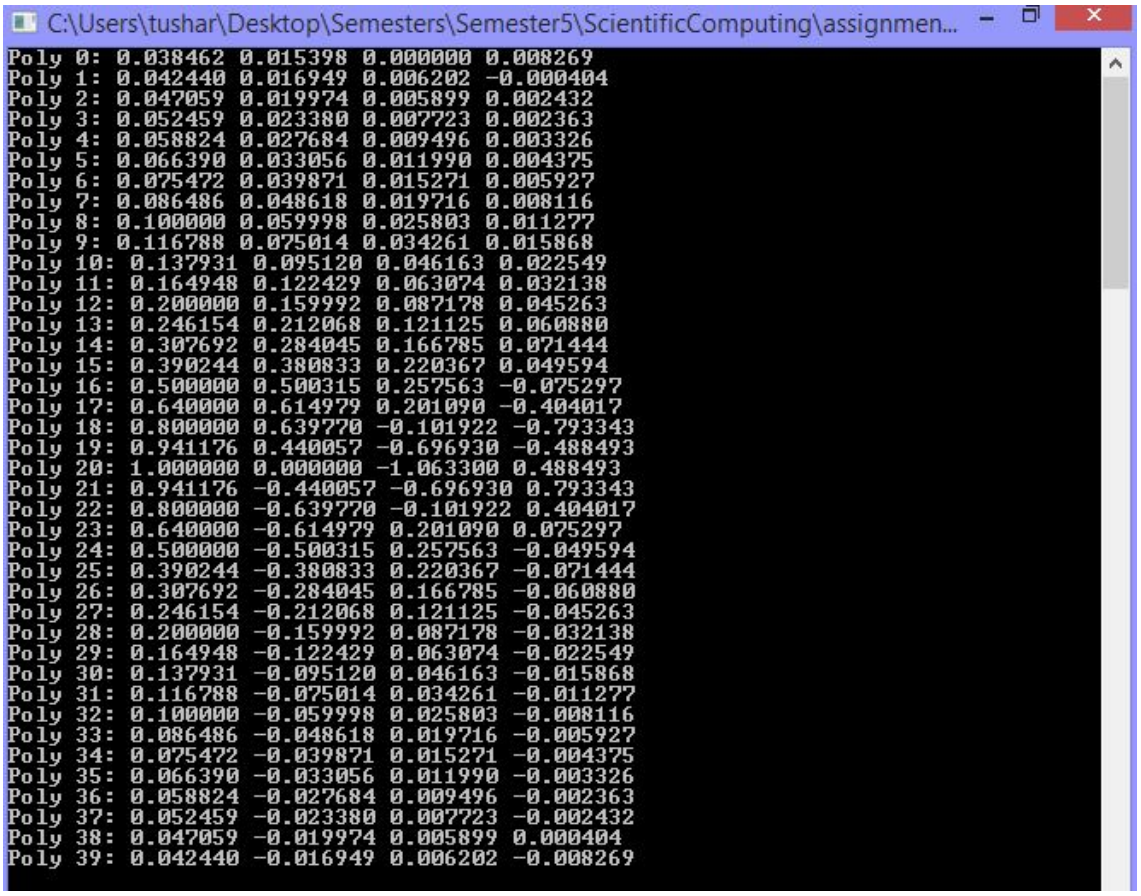


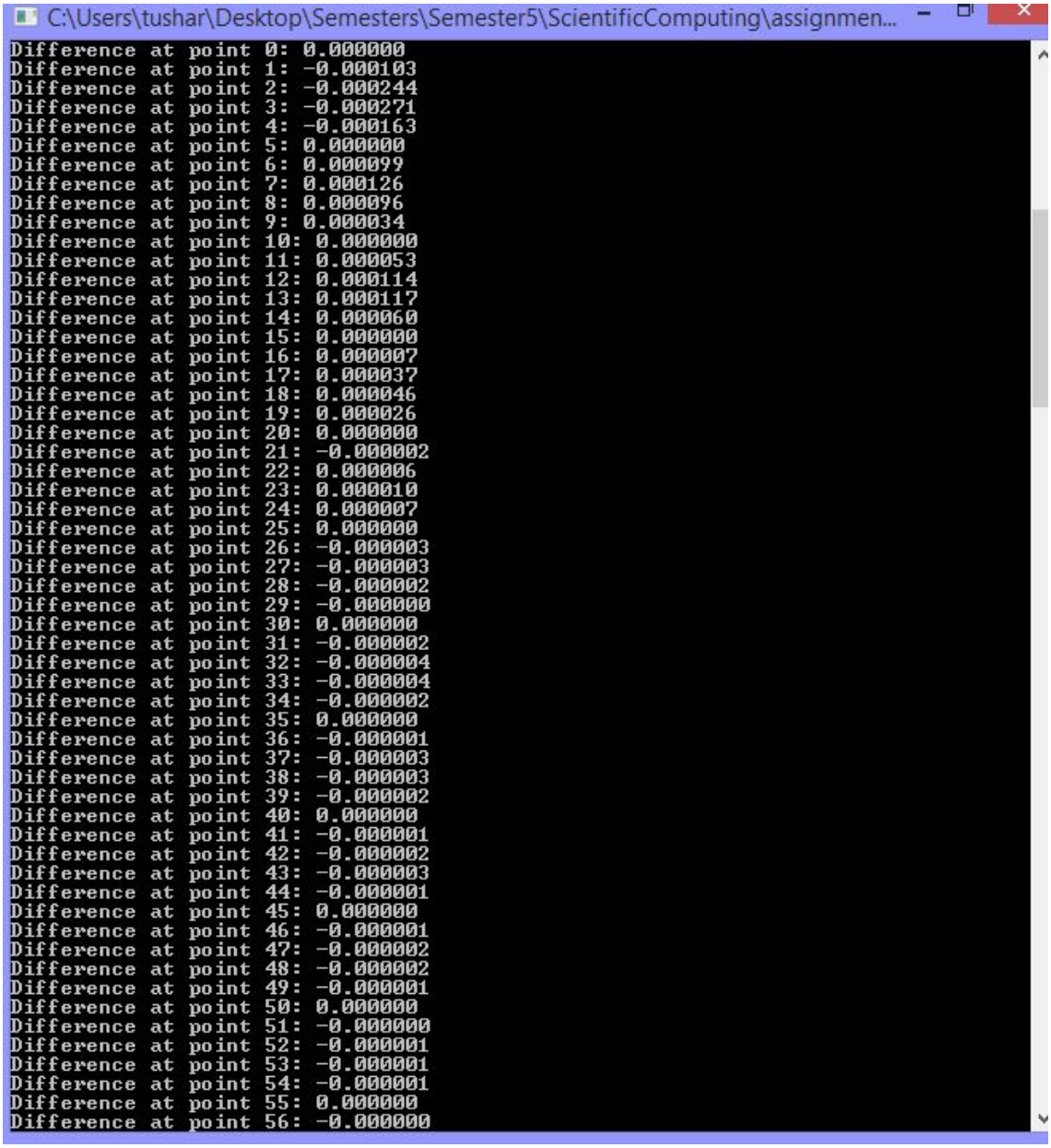
PROBLEM 3(PART (a))

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

void solveThomas(double a[],double b[],double c[],double t[],double f[],int k);
double getF(double x)
{
    return 1/(1 + (x*x));
}

int main()
{
    double left = -5,right = 5;
    int noOfPoints = 41;
    double h = (right-left)/(noOfPoints-1);
    double x[noOfPoints];

    //nodes for interpolation
    x[0] = left;
    for(int i=1; i<noOfPoints; i++)
        x[i] = x[i-1] + h;

    //no of polynomials will be 40
    double a[noOfPoints],b[noOfPoints],c[noOfPoints],d[noOfPoints];
    for(int i=0; i<noOfPoints; i++)
        a[i] = getF(x[i]);


    c[0] = c[noOfPoints-1] = 0;

    int k = noOfPoints - 1; //40
    double tA[k],tB[k],tC[k],tF[k],tX[k];
    for(int i=1; i<k; i++)
    {
        tA[i] = h;
        tB[i] = (4*h);
        tC[i] = h;
        tF[i] = ((double)3/h)*(a[i+1] - a[i] - a[i] + a[i-1]);

    }
    tX[0] = tX[k] = 0;
    tF[0] = tF[k] = 0;
    solveThomas(tA,tB,tC,tX,tF,k);


    for(int i=0; i<=k; i++)
        c[i] = tX[i];
```

```
    for(int i=0; i<k; i++)
    {
        b[i] = ((a[i+1]-a[i])/h) - (h/3)*((2*c[i])+c[i+1]);
        d[i] = (c[i+1]-c[i])/(3*h);
    }

    for(int i=0; i<k; i++)
        printf("Poly %d: %f %f %f %f\n",i,a[i],b[i],c[i],d[i]);

    double low = 0;
    double high = 5;
    int points = 101;
    double H = (high-low)/(points-1);

    cout<<endl;
    for(int i=0; i<points; i++)
    {
        double X = low + (i*H);
        double orig = getF(X);
        int polyNumber = (X+5)/h;
        double estimate = a[polyNumber] + (b[polyNumber]*(X - x[polyNumber])) +  (c[polyNumber]*pow(X
        printf("Difference at point %d: %f\n",i,estimate-orig);
    }

    return 0;
}

void solveThomas(double a[],double b[],double c[],double t[],double f[],int k)
{

    //forward sweep
    f[1] = f[1] - (a[1]*t[0]);
    f[k-1] = f[k-1] - (c[k-1]*t[k]);

    b[1] = b[1]; //unchanged
    //c's are unchanged
    for(int i=2; i<=k-1; i++)
    {
        b[i] = b[i] - ((c[i-1]*a[i])/b[i-1]);
        f[i] = f[i] - ((f[i-1]*a[i])/b[i-1]);
    }

    //find solutions using backward sweep
    t[k-1] = f[k-1]/b[k-1];
    for(int i=k-2; i>=1; i--)
        t[i] = (f[i] - (c[i]*t[i+1]))/b[i];


}
```

OUTPUT


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Poly 0: 0.038462 0.015398 0.000000 0.008269
Poly 1: 0.042440 0.016949 0.006202 -0.000404
Poly 2: 0.047059 0.019974 0.005899 0.002432
Poly 3: 0.052459 0.023380 0.007723 0.002363
Poly 4: 0.058824 0.027684 0.009496 0.003326
Poly 5: 0.066390 0.033056 0.011990 0.004375
Poly 6: 0.075472 0.039871 0.015271 0.005927
Poly 7: 0.086486 0.048618 0.019716 0.008116
Poly 8: 0.100000 0.059998 0.025803 0.011277
Poly 9: 0.116788 0.075014 0.034261 0.015868
Poly 10: 0.137931 0.095120 0.046163 0.022549
Poly 11: 0.164948 0.122429 0.063074 0.032138
Poly 12: 0.200000 0.159992 0.087178 0.045263
Poly 13: 0.246154 0.212068 0.121125 0.060880
Poly 14: 0.307692 0.284045 0.166785 0.071444
Poly 15: 0.390244 0.380833 0.220367 0.049594
Poly 16: 0.500000 0.500315 0.257563 -0.075297
Poly 17: 0.640000 0.614979 0.201090 -0.404017
Poly 18: 0.800000 0.639770 -0.101922 -0.793343
Poly 19: 0.941176 0.440057 -0.696930 -0.488493
Poly 20: 1.000000 0.000000 -1.063300 0.488493
Poly 21: 0.941176 -0.440057 -0.696930 0.793343
Poly 22: 0.800000 -0.639770 -0.101922 0.404017
Poly 23: 0.640000 -0.614979 0.201090 0.075297
Poly 24: 0.500000 -0.500315 0.257563 -0.049594
Poly 25: 0.390244 -0.380833 0.220367 -0.071444
Poly 26: 0.307692 -0.284045 0.166785 -0.060880
Poly 27: 0.246154 -0.212068 0.121125 -0.045263
Poly 28: 0.200000 -0.159992 0.087178 -0.032138
Poly 29: 0.164948 -0.122429 0.063074 -0.022549
Poly 30: 0.137931 -0.095120 0.046163 -0.015868
Poly 31: 0.116788 -0.075014 0.034261 -0.011277
Poly 32: 0.100000 -0.059998 0.025803 -0.008116
Poly 33: 0.086486 -0.048618 0.019716 -0.005927
Poly 34: 0.075472 -0.039871 0.015271 -0.004375
Poly 35: 0.066390 -0.033056 0.011990 -0.003326
Poly 36: 0.058824 -0.027684 0.009496 -0.002363
Poly 37: 0.052459 -0.023380 0.007723 -0.002432
Poly 38: 0.047059 -0.019974 0.005899 0.000404
Poly 39: 0.042440 -0.016949 0.006202 -0.008269


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Difference at point 0: 0.000000
Difference at point 1: -0.000103
Difference at point 2: -0.000244
Difference at point 3: -0.000271
Difference at point 4: -0.000163
Difference at point 5: 0.000000
Difference at point 6: 0.000099
Difference at point 7: 0.000126
Difference at point 8: 0.000096
Difference at point 9: 0.000034
Difference at point 10: 0.000000
Difference at point 11: 0.000053
Difference at point 12: 0.000114
Difference at point 13: 0.000117
Difference at point 14: 0.000060
Difference at point 15: 0.000000
Difference at point 16: 0.000007
Difference at point 17: 0.000037
Difference at point 18: 0.000046
Difference at point 19: 0.000026
Difference at point 20: 0.000000
Difference at point 21: -0.000002
Difference at point 22: 0.000006
Difference at point 23: 0.000010
Difference at point 24: 0.000007
Difference at point 25: 0.000000
Difference at point 26: -0.000003
Difference at point 27: -0.000003
Difference at point 28: -0.000002
Difference at point 29: -0.000000
Difference at point 30: 0.000000
Difference at point 31: -0.000002
Difference at point 32: -0.000004
Difference at point 33: -0.000004
Difference at point 34: -0.000002
Difference at point 35: 0.000000
Difference at point 36: -0.000001
Difference at point 37: -0.000003
Difference at point 38: -0.000003
Difference at point 39: -0.000002
Difference at point 40: 0.000000
Difference at point 41: -0.000001
Difference at point 42: -0.000002
Difference at point 43: -0.000003
Difference at point 44: -0.000001
Difference at point 45: 0.000000
Difference at point 46: -0.000001
Difference at point 47: -0.000002
Difference at point 48: -0.000002
Difference at point 49: -0.000001
Difference at point 50: 0.000000
Difference at point 51: -0.000000
Difference at point 52: -0.000001
Difference at point 53: -0.000001
Difference at point 54: -0.000001
Difference at point 55: 0.000000
Difference at point 56: -0.000000

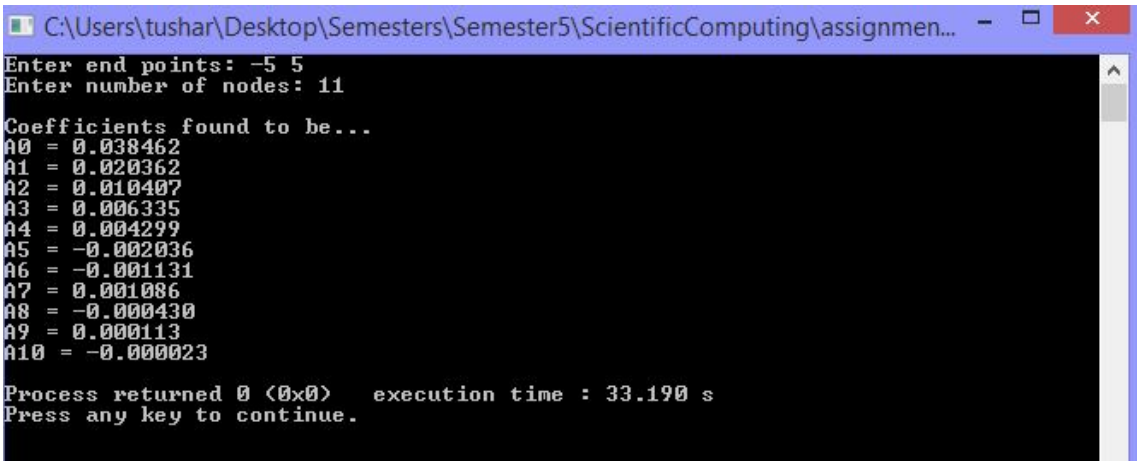
```
Difference at point 57: -0.000001
Difference at point 58: -0.000001
Difference at point 59: -0.000000
Difference at point 60: 0.000000
Difference at point 61: -0.000000
Difference at point 62: -0.000001
Difference at point 63: -0.000001
Difference at point 64: -0.000000
Difference at point 65: 0.000000
Difference at point 66: -0.000000
Difference at point 67: -0.000000
Difference at point 68: -0.000000
Difference at point 69: -0.000000
Difference at point 70: 0.000000
Difference at point 71: -0.000000
Difference at point 72: -0.000000
Difference at point 73: -0.000000
Difference at point 74: -0.000000
Difference at point 75: 0.000000
Difference at point 76: -0.000000
Difference at point 77: -0.000000
Difference at point 78: -0.000000
Difference at point 79: 0.000000
Difference at point 80: 0.000000
Difference at point 81: -0.000000
Difference at point 82: -0.000001
Difference at point 83: -0.000001
Difference at point 84: -0.000000
Difference at point 85: 0.000000
Difference at point 86: 0.000001
Difference at point 87: 0.000001
Difference at point 88: 0.000002
Difference at point 89: 0.000001
Difference at point 90: 0.000000
Difference at point 91: -0.000003
Difference at point 92: -0.000006
Difference at point 93: -0.000007
Difference at point 94: -0.000006
Difference at point 95: 0.000000
Difference at point 96: 0.000010
Difference at point 97: 0.000020
Difference at point 98: 0.000026
Difference at point 99: 0.000021
Difference at point 100: 0.000000

Process returned 0 (0x0)   execution time : 0.502 s
Press any key to continue.
```

PROBLEM 3(PART (b))

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

double getF(double x)
{
    return 1/(1 + (x*x));
}
double getP(double p,double a[],double x[],int noOfPoints)
{
    double mul = 1;
    double ans = 0;
    for(int i=0; i<noOfPoints; i++)
    {
        ans += a[i]*mul;
        mul = mul * (p - x[i]);
    }
```

```
    return ans;
}

int main()
{
    double left,right; //end points
    int noOfPoints; //number of equally spaced points
    double h; //spacing between consequetive nodes

    cout<<"Enter end points: ";
    cin>>left>>right;
    cout<<"Enter number of nodes: ";
    cin>>noOfPoints;
    h = (right-left)/((double)(noOfPoints-1));

    double x[noOfPoints]; //nodes of the interpolating polynomial
    double dividedDiff[noOfPoints][noOfPoints];


    //store all nodes
    x[0] = left;
    for(int i=1; i<noOfPoints; i++)
        x[i] = x[i-1] + h;

    for(int i=0; i<noOfPoints; i++)
        dividedDiff[i][i] = getF(x[i]);

    //calculate divided differences
    for(int i=1; i<noOfPoints; i++)
    {
        int r = 0;
        int c = i;
        while(c<noOfPoints)
        {
            dividedDiff[r][c] = (dividedDiff[r+1][c] - dividedDiff[r][c-1])/(x[c] - x[r]);
            r++;
            c++;
        }
    }

    //print coefficients
    double a[noOfPoints];
    cout<<"\nCoefficients found to be...\n";
    for(int i=0; i<noOfPoints; i++)
    {
        a[i] = dividedDiff[0][i];
        printf("A%d = %f\n",i,dividedDiff[0][i]);
    }


    return 0;
}
```

OUTPUT


C:\Users\tushar\Desktop\Semesters\Semester5\ScientificComputing\assignmen...
Enter end points: -5 5
Enter number of nodes: 11
Coefficients found to be...
A0 = 0.038462
A1 = 0.020362
A2 = 0.010407
A3 = 0.006335
A4 = 0.004299
A5 = -0.002036
A6 = -0.001131
A7 = 0.001086
A8 = -0.000430
A9 = 0.000113
A10 = -0.000023
Process returned 0 (0x0) execution time : 33.190 s
Press any key to continue.

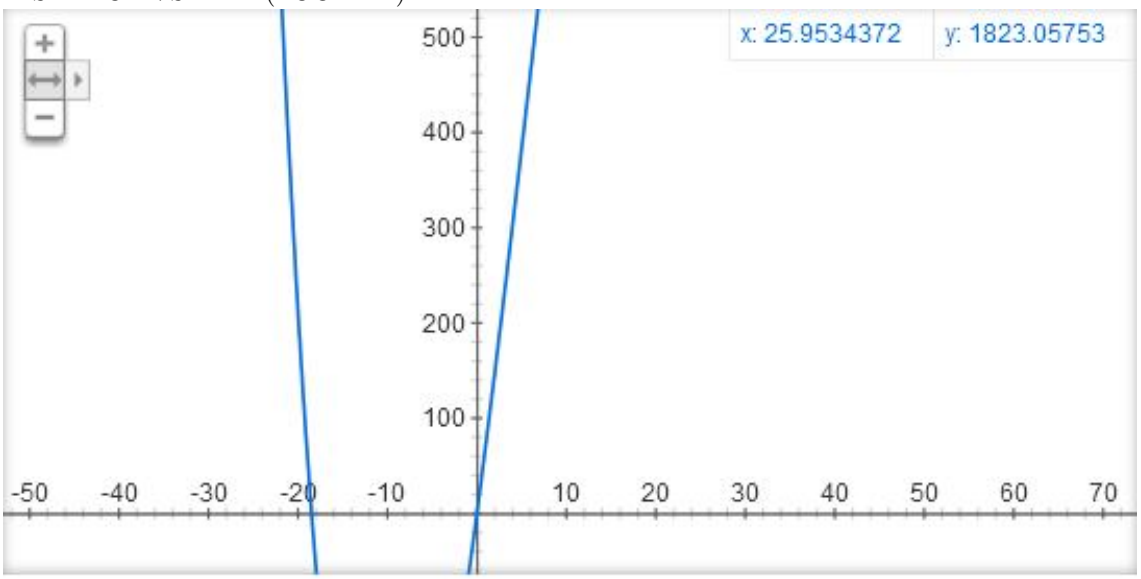
PROBLEM 4(PART (a))

Code to interpolate Distance VS Time

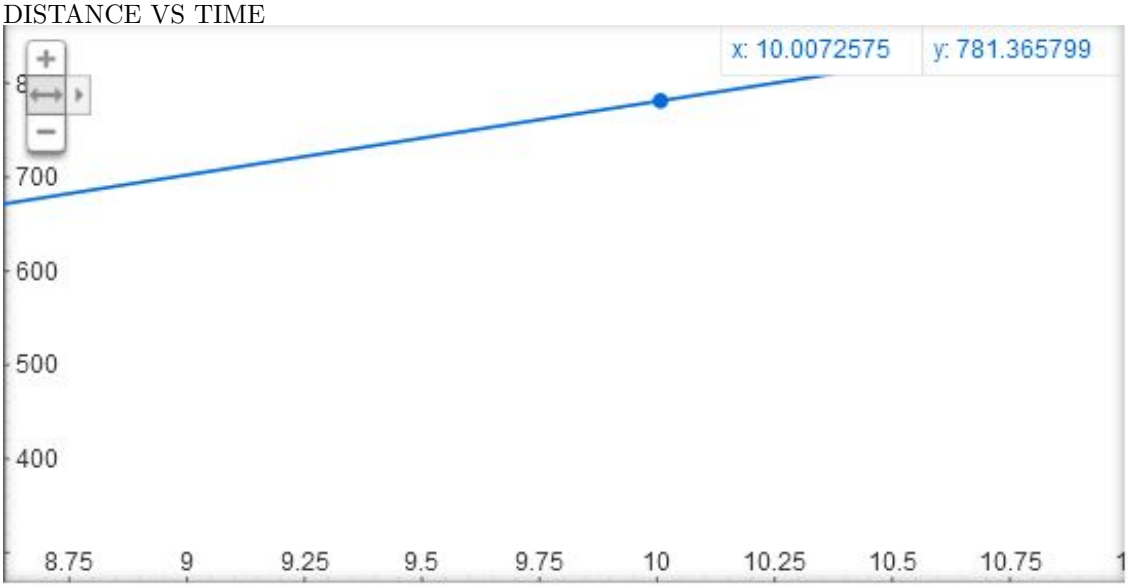
```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

double getF(double x)
{
    if(x == 0)
        return 0;
    if(x == 3)
        return 225;
    if(x == 5)
        return 383;
    if(x == 8)
        return 623;
    if(x == 13)
        return 993;
}
double getP(double p,double a[],double x[],int noOfPoints)
{
    double mul = 1;
    double ans = 0;
    for(int i=0; i<noOfPoints; i++)
    {
        ans += a[i]*mul;
        mul = mul * (p - x[i]);
    }
    return ans;
}

int main()
{
    int noOfPoints = 5;
    double x[5] = {0,3,5,8,13}; //nodes of the interpolating polynomial
    double dividedDiff[noOfPoints][noOfPoints];

    for(int i=0; i<noOfPoints; i++)
        dividedDiff[i][i] = getF(x[i]);

    //calculate divided differences
    for(int i=1; i<noOfPoints; i++)
    {
        int r = 0;
        int c = i;
        while(c<noOfPoints)
        {
            dividedDiff[r][c] = (dividedDiff[r+1][c] - dividedDiff[r][c-1])/(x[c] - x[r]);
            r++;
            c++;
        }
```

```
    }

    //print coefficients
    double a[noOfPoints];
    cout<<"\nPolynomial to predict Distance VS Time...\nCoefficients found to be...\n";
    for(int i=0; i<noOfPoints; i++)
    {
        a[i] = dividedDiff[0][i];
        printf("A%d = %f\n",i,dividedDiff[0][i]);
    }

    cout<<"\nChecking difference in [0,8]...\n";
    for(int i=0; i<noOfPoints; i++)
    {
        printf("Difference at %d = %f\n",i,abs(getF(x[i]) - getP(x[i],a,x,noOfPoints)));
    }



    return 0;
}
```

DISTANCE VS TIME(ZOOMED)



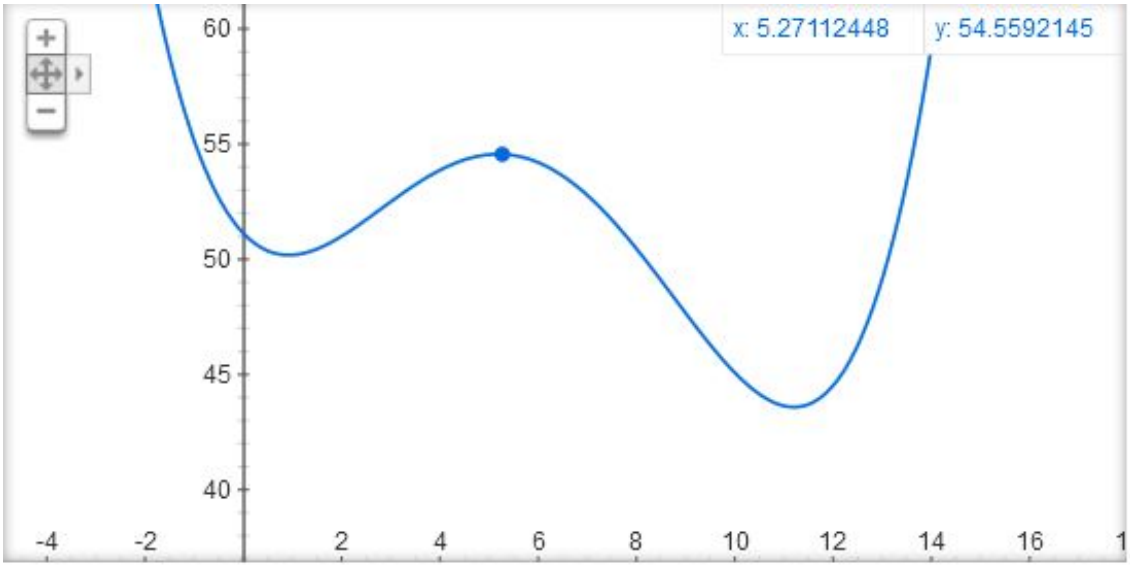
DISTANCE VS TIME



Approximate position at t = 10 is 781.365 feet!
Approximate speed at t = 10 is 37.50 miles/r **PROBLEM 4(PART (b))**

```
#include<iostream>
#include<stdio.h>
#include<cmath>
using namespace std;

double getMiles(int k)
{
    k = k*3600;
    return k*0.00018939;
}

double getF(double x)
{
    if(x == 0)
        return getMiles(75);
    if(x == 3)
        return getMiles(77);
    if(x == 5)
        return getMiles(80);
    if(x == 8)
        return getMiles(74);
    if(x == 13)
        return getMiles(72);
}
double getP(double p,double a[],double x[],int noOfPoints)
{
    double mul = 1;
    double ans = 0;
    for(int i=0; i<noOfPoints; i++)
    {
```

```
        ans += a[i]*mul;
        mul = mul * (p - x[i]);
    }
    return ans;
}

int main()
{
    int noOfPoints = 5;
    double x[5] = {0,3,5,8,13}; //nodes of the interpolating polynomial
    double dividedDiff[noOfPoints][noOfPoints];

    for(int i=0; i<noOfPoints; i++)
        dividedDiff[i][i] = getF(x[i]);

    //calculate divided differences
    for(int i=1; i<noOfPoints; i++)
    {
        int r = 0;
        int c = i;
        while(c<noOfPoints)
        {
            dividedDiff[r][c] = (dividedDiff[r+1][c] - dividedDiff[r][c-1])/(x[c] - x[r]);
            r++;
            c++;
        }
    }

    //print coefficients
    double a[noOfPoints];
    cout<<"\nPolynomial to predict Distance VS Time...\nCoefficients found to be...\n";
    for(int i=0; i<noOfPoints; i++)
    {
        a[i] = dividedDiff[0][i];
        printf("A%d = %f\n",i,dividedDiff[0][i]);
    }

    cout<<"\nChecking difference in [0,8]...\n";
    for(int i=0; i<noOfPoints; i++)
    {
        printf("Difference at %d = %f\n",i,abs(getF(x[i]) - getP(x[i],a,x,noOfPoints)));
    }



    return 0;
}
```



Max speed seems to be 54.55 miles/hr at t = 5.27!