```
#include<bits/stdc++.h>
using namespace std;

double del[5][5];
double x[5],y[5];

void forwardDiff()
{
	for(int i=0; i<5; i++) del[0][i] = y[i];

	for(int i=1; i<5; i++)
	{
		for(int j=0; j<5-i; j++)
			del[i][j] = del[i-1][j+1]-del[i-1][j];
	}
}

double cal1stOrder(double u, double h)
{
	return (1/h) * (del[1][0] + (((2*u-1)/2)*del[2][0]) + (((3*u*u-6*u+2)/6)*del[3][0]) + (((4*u*u*u-
18*u*u+22*u-6)/24)*del[4][0]));
}

double cal2ndOrder(double u, double h)
{
	return (1/(h*h)) * (del[2][0] + ((u-1)*del[3][0]) + (((12*u*u-36*u+22)/24)*del[4][0]));
}
int main()
{
	cout << "Enter the x values: ";
	for(int i=0; i<5; ++i) cin >> x[i];
	cout << "Enter the y values: ";
	for(int i=0; i<5; i++) cin >> y[i];

	double xp;
	cout << "Enter the point: ";
	cin >> xp; //xp=1

	forwardDiff();

	double h = x[1]-x[0];
	double u = (xp-x[0])/h;
	cout << "1st order derivative = "<< cal1stOrder(u,h) << endl << "2nd order derivative = " <<
cal2ndOrder(u,h) << endl;
	return 0;
}
/*
1 2 3 4 5
1 8 27 64 125
1
*/
```

```
#include<bits/stdc++.h>
using namespace std;

double del[5][5];
double x[5],y[5];

void forwardDiff()
{
	for(int i=0; i<5; i++)   del[0][i] = y[i];

	for(int i=1; i<5; i++)
	{
		for(int j=0; j<5-i; j++)
			del[i][j] = del[i-1][j+1]-del[i-1][j];
	}
}

double cal1stOrder(double u, double h)
{
	return (1/h) * (del[1][0] + (((2*u-1)/2)*del[2][0]) + (((3*u*u-6*u+2)/6)*del[3][0]) + (((4*u*u*u-
18*u*u+22*u-6)/24)*del[4][0]));
}

double cal2ndOrder(double u, double h)
{
	return (1/(h*h)) * (del[2][0] + ((u-1)*del[3][0]) + (((12*u*u-36*u+22)/24)*del[4][0]));
}
int main()
{
	cout << "Enter the x values: ";
	for(int i=0; i<5; ++i) cin >> x[i];
	cout << "Enter the y values: ";
	for(int i=0; i<5; i++) cin >> y[i];

	double xp;
	cout << "Enter the point: ";
	cin >> xp; //xp=1.5

	forwardDiff();

	double h = x[1]-x[0];
	double u = (xp-x[0])/h;
	cout << "1st order derivative = "<< cal1stOrder(u,h) << endl << "2nd order derivative = " <<
cal2ndOrder(u,h) << endl;
	return 0;
}
/*
1 2 3 4 5
1 8 27 64 125
1.5
*/
```

```
#include<bits/stdc++.h>
using namespace std;

const int mx = 1000;
double y[mx];

void calDel(double a,double b,double h, int n)
{
	for(int i=0; i<=n; i++)
	{
		double x = a+i*h;
		y[i] = 5*log10(x);
//		cout << x << " " << y[i] << " " << log10(x) << " ";
	}
}

double trapezoid(double h,int n)
{
	double I=0;
	for(int i=1;i<n;i++) I += y[i];
	I += (h/2) * (y[0]+2*I+y[n]);
	return I;
}

int main()
{
	double a,b;
	int n;
	cout << "lower & upper limit, a & b : ";
	cin >> a >>b;
	cout << "Number of intervals, n: ";
	cin >> n;

	if(a<1 || b<1 || b<a){
		cout << "the entered interval limit [a,b] is not valid" << endl; return 0;
	}

	double h = (b-a)/(n*1.0);
	calDel(a,b,h,n);
	cout << "The approximiate area is = " << trapezoid(h,n) << endl;

	return 0;
}
/*
1 100
10
*/
```

```
#include<bits/stdc++.h>

#define PI acos(-1.0)
#define toradian(degree) (PI*degree)/180
#define todegree(radian) (radian*180)/PI
#define sinD(degree) sin((degree * PI) / 180.0)
#define cosD(degree) cos((degree * PI) / 180.0)
#define tanD(degree) tan((degree * PI) / 180.0)
#define cotD(degree) (1.0 / tanD(degree))
#define secD(degree) (1.0 / cosD(degree))
#define cosecD(degree) (1.0 / sinD(degree))
#define asinD(value) (asin(value) * 180.0) / PI
#define acosD(value) (acos(value) * 180.0) / PI
#define atanD(value) (atan(value) * 180.0) / PI
using namespace std;


const int mx = 1000;
double y[mx];

void calDel(double a,double b,double h, int n)
{
	for(int i=0; i<=n; i++)
	{
		double x = a+i*h;
		y[i] = (PI/2)* exp(todegree(sinD(x)));
	}
}

double simpson13(double h,int n)
{
	double I=0,s1=0,s2=0;
	for(int i=1;i<n;i+=2) s1 += y[i];
	for(int i=2;i<n-1;i+=2) s2 += y[i];
	I += (h/3) * (y[0]+ 4*s1 + 2*s2 +y[n]);
	return I;
}

int main()
{
	double a,b;
	int n;
	cout << "lower & upper limit, a & b : ";
	cin >> a >>b;
	cout << "Number of intervals, n: ";
	cin >> n;

	if(b<a){
		cout << "the entered interval limit [a,b] is not valid" << endl; return 0;
	}
	double h = (b-a)/(n*1.0);
	calDel(a,b,h,n);
	cout << "The approximiate area is = " << simpson13(h,n) << endl;

	return 0;
}
```

```
#include<bits/stdc++.h>

#define PI acos(-1.0)
#define toradian(degree) (PI*degree)/180
#define todegree(radian) (radian*180)/PI
#define sinD(degree) sin((degree * PI) / 180.0)
#define cosD(degree) cos((degree * PI) / 180.0)
#define tanD(degree) tan((degree * PI) / 180.0)
#define cotD(degree) (1.0 / tanD(degree))
#define secD(degree) (1.0 / cosD(degree))
#define cosecD(degree) (1.0 / sinD(degree))
#define asinD(value) (asin(value) * 180.0) / PI
#define acosD(value) (acos(value) * 180.0) / PI
#define atanD(value) (atan(value) * 180.0) / PI
using namespace std;


const int mx = 1000;
double y[mx];

void calDel(double a,double b,double h, int n)
{
	for(int i=0; i<=n; i++)
	{
		double x = a+i*h;
		y[i] = x/(1+(x*x));
	}
}

double simpson38(double h,int n)
{
	double I=0,s1=0,s2=0;
	for(int i=1;i<n;i++) {if(i%3==0) continue; s1 += y[i];}
	for(int i=3;i<n-2;i+=3) s2 += y[i];
	I += (3*h/8) * (y[0]+ 3*s1 + 2*s2 +y[n]);
	return I;
}

int main()
{
	double a,b;
	int n;
	cout << "lower & upper limit, a & b : ";
	cin >> a >>b;
	cout << "Number of intervals, n: ";
	cin >> n;

	if(b<a){
		cout << "the entered interval limit [a,b] is not valid" << endl; return 0;
	}
	double h = (b-a)/(n*1.0);
	calDel(a,b,h,n);
	cout << "The approximiate area is = " << simpson38(h,n) << endl;

	return 0;
}
```

```
#include<bits/stdc++.h>
using namespace std;

double a[5][5];
double b[4][4];

double calculateDet3(int col)
{
	for(int i=2, bi=1; i<5; i++, bi++)
	{
		for(int j=1, bj=1;j<5;j++)
		{
			if(j==col) continue;
			b[bi][bj] = a[i][j];
			bj++;
		}
	}

	return (b[1][1]*(b[2][2]*b[3][3] - b[2][3]*b[3][2]) - b[1][2]*(b[2][1]*b[3][3] - b[2][3]*b[3][1]) +
b[1][3]*(b[2][1]*b[3][2] - b[2][2] * b[3][1]));
}
double calculateDet4()
{
	double det = a[1][1]*calculateDet3(1) - a[1][2]*calculateDet3(2) + a[1][3]*calculateDet3(3) - a[1]
[4]*calculateDet3(4);
	return det;
}

int main()
{
	cout << "Enter the values of 4x4 matrix:\n";
	for (int i = 1; i < 5; ++i)
	{
		for (int j = 1; j < 5; ++j)
			cin >> a[i][j];
	}
	cout << "Determinant of the entered matrix is = "<< calculateDet4() << endl;
	return 0;
}
/*
2 -1 3 0
-3 1 0 4
-2 1 4 1
-1 3 0 -2
*/
```

```
#include<bits/stdc++.h>
using namespace std;

double a[4][4], con[4];
double c[4][4], x[4], b[3][3];

double calculateDet3()
{
	return (a[1][1]*(a[2][2]*a[3][3] - a[2][3]*a[3][2]) - a[1][2]*(a[2][1]*a[3][3] - a[2][3]*a[3][1]) +
a[1][3]*(a[2][1]*a[3][2] - a[2][2] * a[3][1]));
}

double cofactor(int row,int col)
{
	for(int i=1, bi=1; i<4; i++)
	{
		if(i==row) continue;
		for(int j=1, bj=1;j<4;j++)
		{
			if(j==col) continue;
			b[bi][bj] = a[i][j];
			bj++;
		}
		bi++;
	}

	return (pow(-1,row+col) * (b[1][1]*b[2][2] - b[1][2]*b[2][1]));
}

void getAInverse(double det)
{
	for(int i=1; i<4; i++)   ///get C matrix
	{
		for(int j=1;j<4;j++) c[i][j] = cofactor(i,j);
	}
	for(int i=1; i<4; i++)   ///get Adjoint of A = Transpose of C
	{
		for(int j=i;j<4;j++) swap(c[i][j],c[j][i]);
	}
	for(int i=1; i<4; i++)
	{
		for(int j=0; j<4; j++)
			c[i][j] /= det;
	}
}

void solveX(double det)
{
	getAInverse(det);
	for (int i = 1; i < 4; ++i)
		x[i] = c[i][1]*con[1] + c[i][2]*con[2] + c[i][3]*con[3];
}

int main()
{
	cout << "Enter the coefficient & constant values :\n";
```

```
	for (int i = 1; i < 4; ++i)
	{
		for (int j = 1; j < 4; ++j)
			cin >> a[i][j];
		cin >> con[i];
	}

	double detA = calculateDet3();
	if(detA == 0) { cout << "The solution is not possible since the det is 0."; return 0; }
	solveX(detA);

	cout << "x = " << x[1] << endl;
	cout << "y = " << x[2] << endl;
	cout << "z = " << x[3] << endl;
	return 0;
}
/*
1 1 1 1
1 2 3 6
1 3 4 6
*/
```

```
#include <bits/stdc++.h>
using namespace std;

double a[4][4], con[4];
double tempA[4][5];
double det[4];

double detfun(int col)
{
        if(col!=0){
                for(int i=1; i<4; i++)
                        for(int j=0; j<4; j++)
                                tempA[i][j] = a[i][j];
                for(int i=1; i<4; i++)   tempA[i][col] = con[i];
        }

   return (tempA[1][1]*(tempA[2][2]*tempA[3][3] - tempA[2][3]*tempA[3][2]) - tempA[1]
[2]*(tempA[2][1]*tempA[3][3] - tempA[2][3]*tempA[3][1]) + tempA[1][3]*(tempA[2][1]*tempA[3][2]
- tempA[2][2] * tempA[3][1]));
}

int main()
{
        cout << "Enter the coefficient & constant values :\n";
   for(int i=1; i<4; i++)
   {
      for(int j=1; j<4; j++)
      {
         cin >> a[i][j];
         tempA[i][j] = a[i][j];
      }
      cin >> con[i];
   }

   det[0] = detfun(0);
   det[1] = detfun(1);
   det[2] = detfun(2);
   det[3] = detfun(3);

   if(det[0]==0){
      cout << "No unique solution" << endl << endl; return 0;
   }

   double x = det[1]/det[0];
   double y = det[2]/det[0];
   double z = det[3]/det[0];

   cout << "Unique solution: " << setprecision(3) << fixed << x << " " << y << " " << z << endl << endl;



   return 0;
}
```

```
#include<bits/stdc++.h>
using namespace std;

double x[6],y[6];
double xi,yi,xi2,xy;
double a,b;

void calculateSums()
{
	for (int i = 1; i < 6; ++i)
	{
		xi += x[i];
		yi += y[i];
		xi2 += x[i]*x[i];
		xy += x[i]*y[i];
	}
}

void calculateLine(int n)
{
	calculateSums();

	b = ((n*xy - xi*yi)/(n*xi2 - xi*xi));
	a = (yi/n) - b*(xi/n);
}

int main()
{
	cout << "X values are : ";
	for (int i = 1; i < 6; ++i)	cin >> x[i];
	cout << "Y values are : ";
	for (int i = 1; i < 6; ++i)	cin >> y[i];

	calculateLine(5);
	cout << "The line is = " << a << " + " << b << "x" << endl;
	return 0;
}
```

```
#include<bits/stdc++.h>
using namespace std;

double x[6],y[6];
double xi[4][4], tempA[4][4], con[4];
double a,b,c;

void calculateSums()
{
	xi[1][1] = 5;
	for (int i = 1; i < 6; ++i)
	{
		xi[1][2] += x[i];
		xi[2][1] = xi[1][2];

		xi[1][3] += x[i]*x[i];
		xi[2][2] = xi[3][1]= xi[1][3];

		xi[2][3] += x[i]*x[i]*x[i];
		xi[3][2] = xi[2][3];


		xi[3][3] += x[i]*x[i]*x[i]*x[i];

		con[1] += y[i];
		con[2] += x[i]*y[i];
		con[3] += x[i]*x[i]+y[i];
	}
}

double detfun(int col)
{
	for(int i=1; i<4; i++)
		for(int j=0; j<4; j++)
			tempA[i][j] = xi[i][j];
	if(col>0)
		{ for(int i=1; i<4; i++)tempA[i][col] = con[i]; }

   return (tempA[1][1]*(tempA[2][2]*tempA[3][3] - tempA[2][3]*tempA[3][2]) - tempA[1]
[2]*(tempA[2][1]*tempA[3][3] - tempA[2][3]*tempA[3][1]) + tempA[1][3]*(tempA[2][1]*tempA[3][2]
- tempA[2][2] * tempA[3][1]));
}

void solveEq()
{
	double det[4];
	det[0] = detfun(0);
   det[1] = detfun(1);
   det[2] = detfun(2);
   det[3] = detfun(3);

   a = det[1]/det[0];
	b = det[2]/det[0];
   c = det[3]/det[0];
}
```

```
void calculateParabolla(int n)
{
	calculateSums();
	solveEq();
}

int main()
{
	cout << "X values are : ";
	for (int i = 1; i < 6; ++i)	cin >> x[i];
	cout << "Y values are : ";
	for (int i = 1; i < 6; ++i)	cin >> y[i];

	calculateParabolla(5);
	cout << "The parabolla is = " << a << " + " << b << "x " << c << "x^2" << endl;
	return 0;
}

/*
-2 1 0 1 2
1 2 3 4 5
*/
```