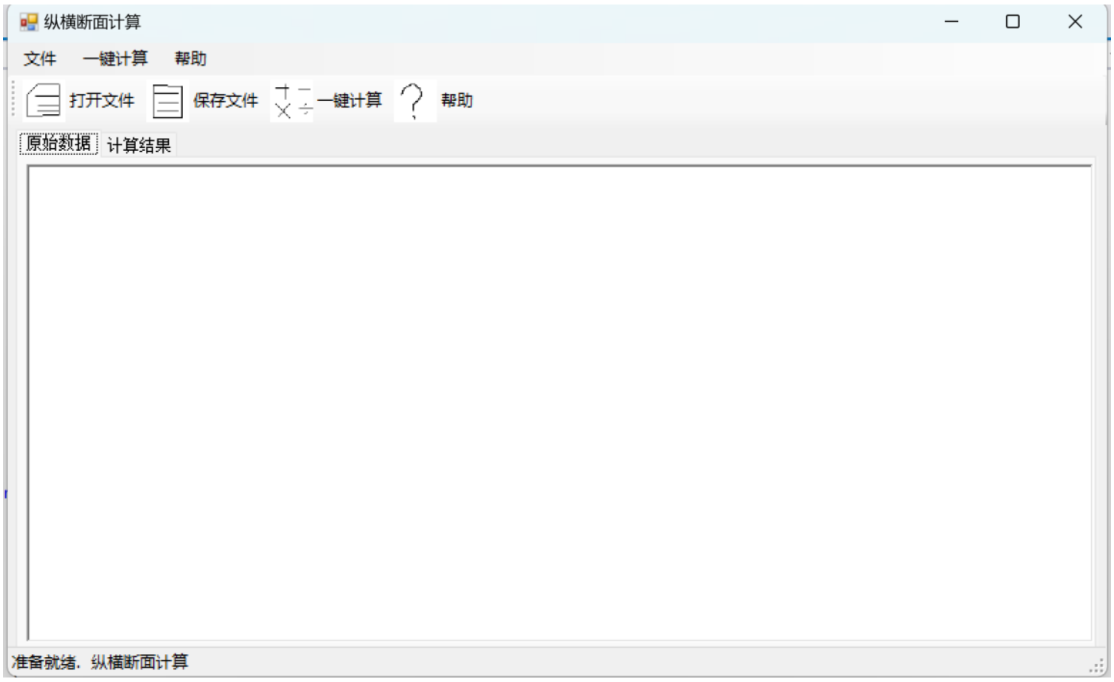
报告文档

程序优化性说明

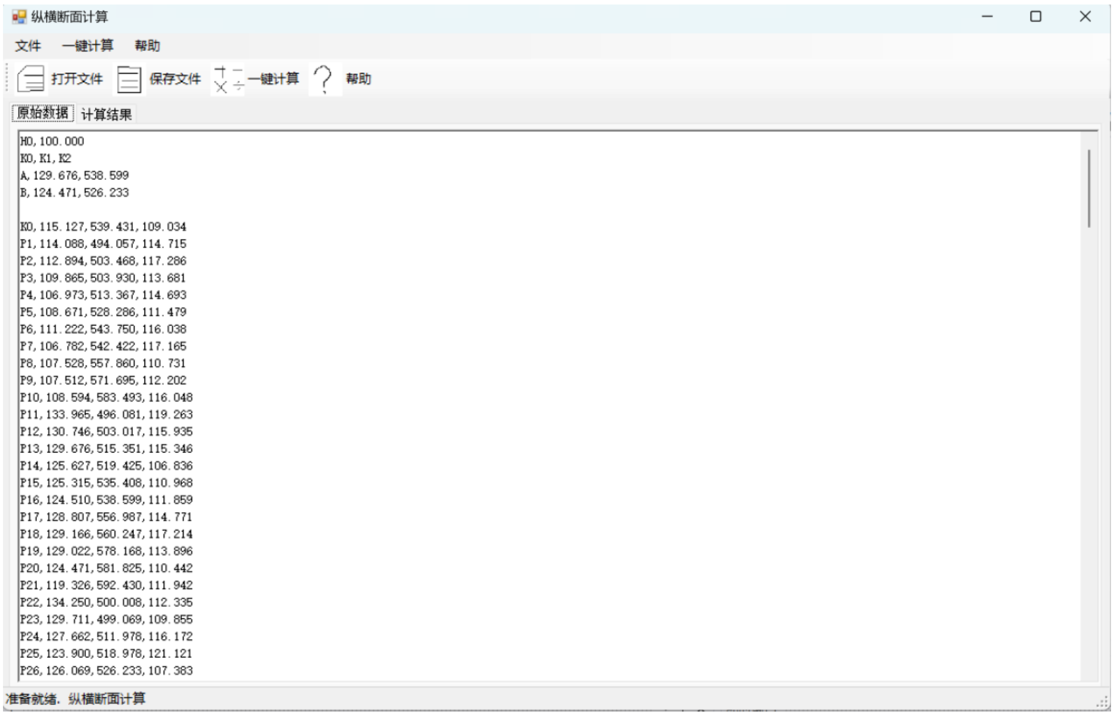
用户交互界面说明

交互界面设计有文件操作按钮，一键计算按钮和帮助按钮



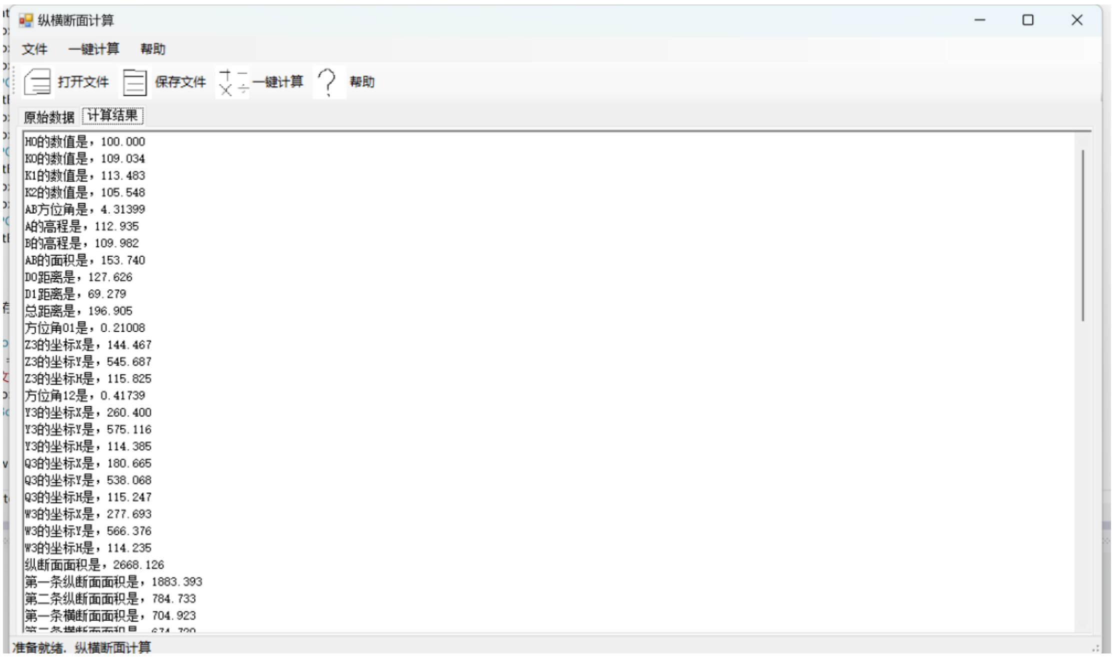
程序运行过程说明

打开文件



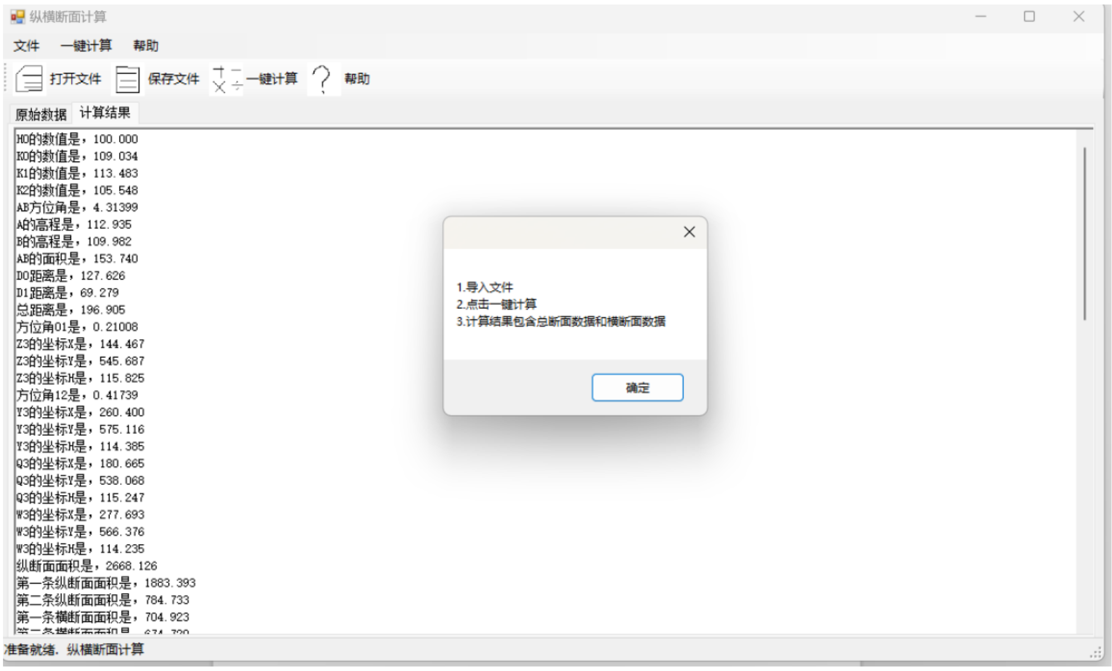
```
H0,100.000
K0,K1,K2
A,129.676,538.599
B,124.471,526.233

K0,115.127,539.431,109.034
P1,114.088,494.057,114.715
P2,112.894,503.468,117.286
P3,109.865,503.930,113.681
P4,106.973,513.367,114.693
P5,108.671,528.286,111.479
P6,111.222,543.750,116.038
P7,106.782,542.422,117.165
P8,107.528,557.860,110.731
P9,107.512,571.695,112.202
P10,108.594,583.493,116.048
P11,133.965,496.081,119.263
P12,130.746,503.017,115.935
P13,129.676,515.351,115.346
P14,125.627,519.425,106.836
P15,125.315,535.408,110.968
P16,124.510,538.599,111.859
P17,128.807,556.987,114.771
P18,129.166,560.247,117.214
P19,129.022,578.168,113.896
P20,124.471,581.825,110.442
P21,119.326,592.430,111.942
P22,134.250,500.008,112.335
P23,129.711,499.069,109.855
P24,127.662,511.978,116.172
P25,123.900,518.978,121.121
P26,126.069,526.233,107.383
```

点击计算手动跳转计算结果



```
H0的数值是，100.000
K0的数值是，109.034
K1的数值是，113.483
K2的数值是，105.548
AB方位角是，4.31399
A的高程是，112.935
B的高程是，109.982
AB的面积是，153.740
D0距离是，127.626
D1距离是，69.279
总距离是，196.905
方位角01是，0.21008
Z3的坐标X是，144.467
Z3的坐标Y是，545.687
Z3的坐标H是，115.825
方位角12是，0.41739
Y3的坐标X是，260.400
Y3的坐标Y是，575.116
Y3的坐标H是，114.385
Q3的坐标X是，180.665
Q3的坐标Y是，538.068
Q3的坐标H是，115.247
W3的坐标X是，277.693
W3的坐标Y是，566.376
W3的坐标H是，114.235
纵断面面积是，2668.126
第一条纵断面面积是，1883.393
第二条纵断面面积是，784.733
第一条横断面面积是，704.923
第二条横断面面积是，674.720
```

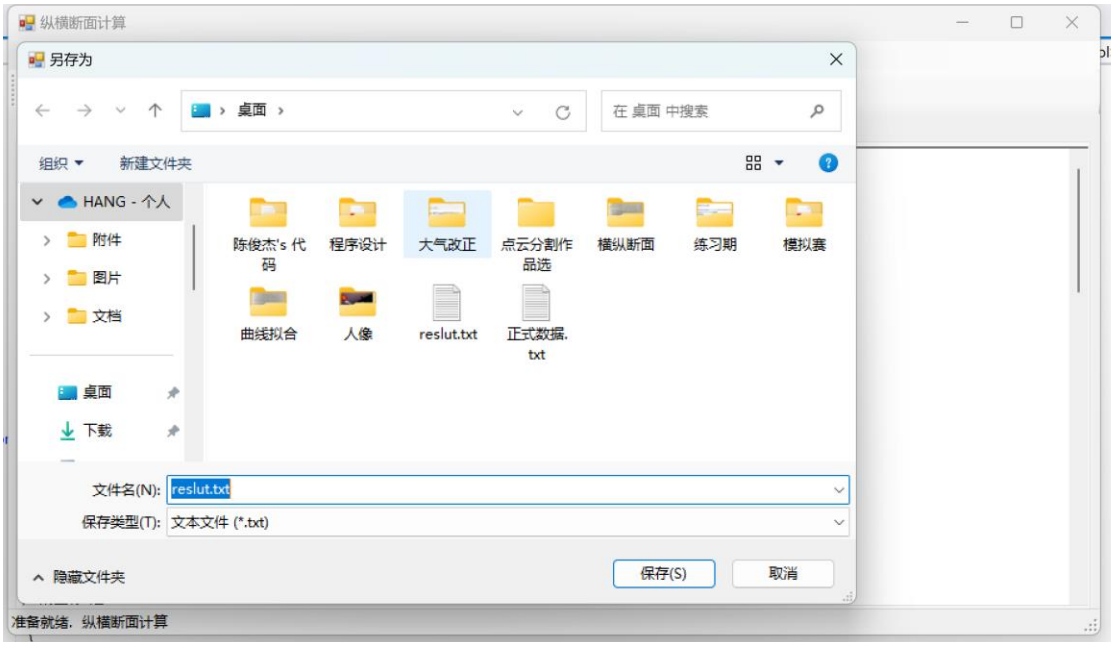
帮助按钮



```
H0的数值是，100.000
K0的数值是，109.034
K1的数值是，113.483
K2的数值是，105.548
AB方位角是，4.31399
A的高程是，112.935
B的高程是，109.982
AB的面积是，153.740
D0距离是，127.626
D1距离是，69.279
总距离是，196.905
方位角01是，0.21008
Z3的坐标X是，144.467
Z3的坐标Y是，545.687
Z3的坐标H是，115.825
方位角12是，0.41739
Y3的坐标X是，260.400
Y3的坐标Y是，575.116
Y3的坐标H是，114.385
Q3的坐标X是，180.665
Q3的坐标Y是，538.068
Q3的坐标H是，115.247
W3的坐标X是，277.693
W3的坐标Y是，566.376
W3的坐标H是，114.235
纵断面面积是，2668.126
第一条纵断面面积是，1883.393
第二条纵断面面积是，784.733
第一条横断面面积是，704.923
第二条横断面面积是，674.720
```

1.导入文件
2.点击一键计算
3.计算结果包含总断面数据和横断面数据

保存文件



文件名(N): reslut.txt

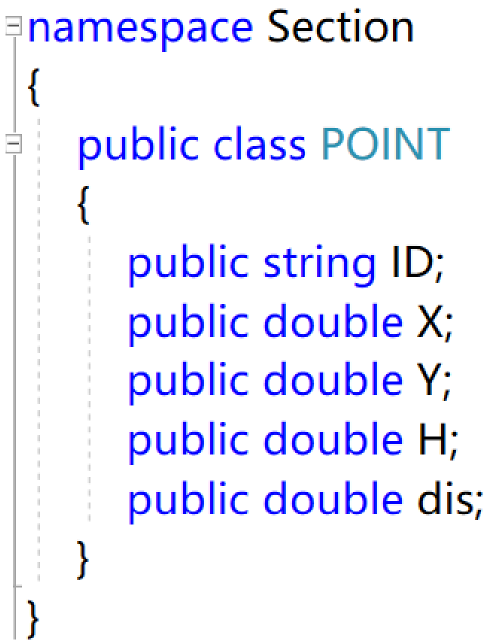
保存类型(T): 文本文件 (*.txt)

程序规范性说明

程序功能与结构设计说明

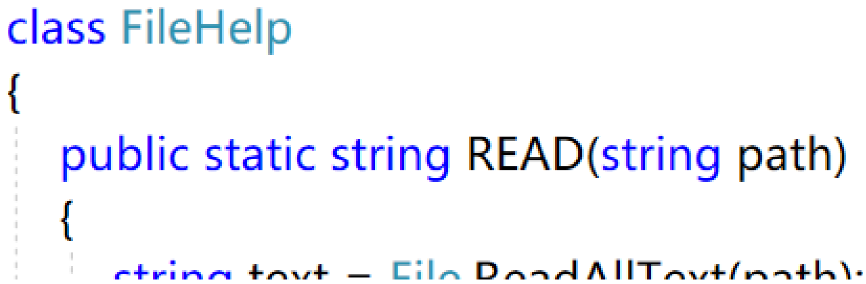
程序主要实现了横纵断面的计算, 在程序运行时若在未计算时点击保存会提醒用户先导入文件并计算, 若还未导入文件便点击计算会提醒用户先导入文件。

点的设计如下



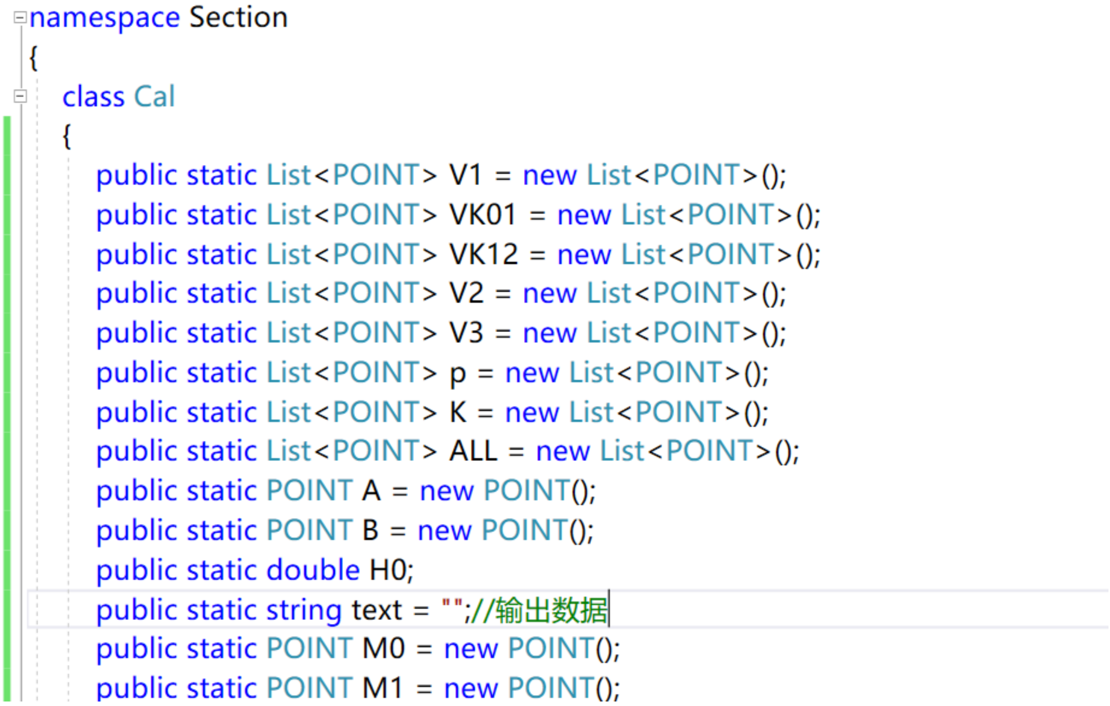
```
namespace Section
{
    public class POINT
    {
        public string ID;
        public double X;
        public double Y;
        public double H;
        public double dis;
    }
}
```

在 filehelp 类中主要用于打开文件包括了读文件一个方法



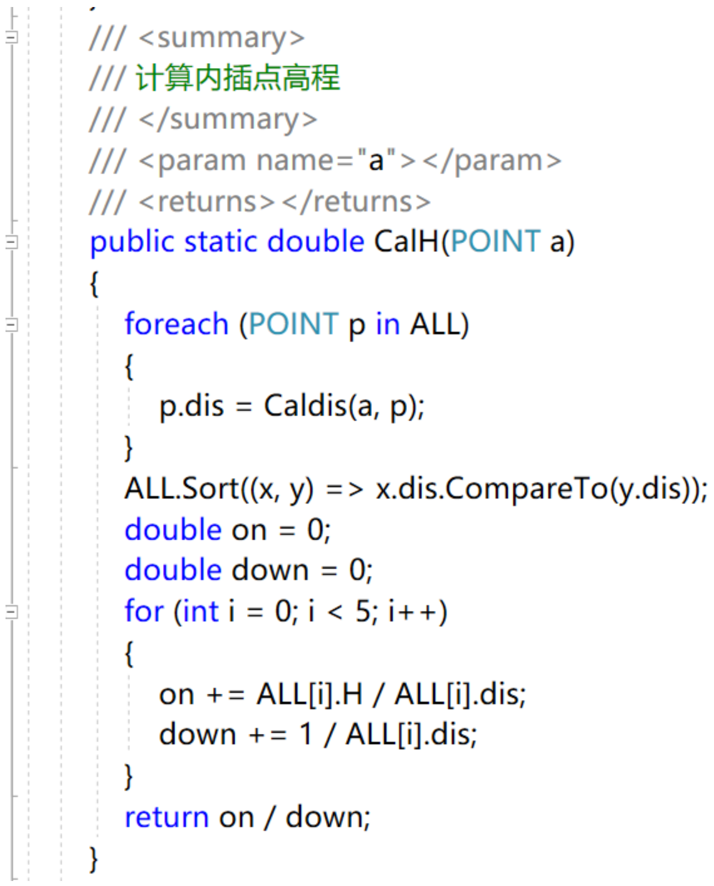
```
class FileHelp
{
    public static string READ(string path)
    {
        string text = File.ReadAllText(path);
```

在 cal 类中包括了所有的计算相关的所需数据和方法

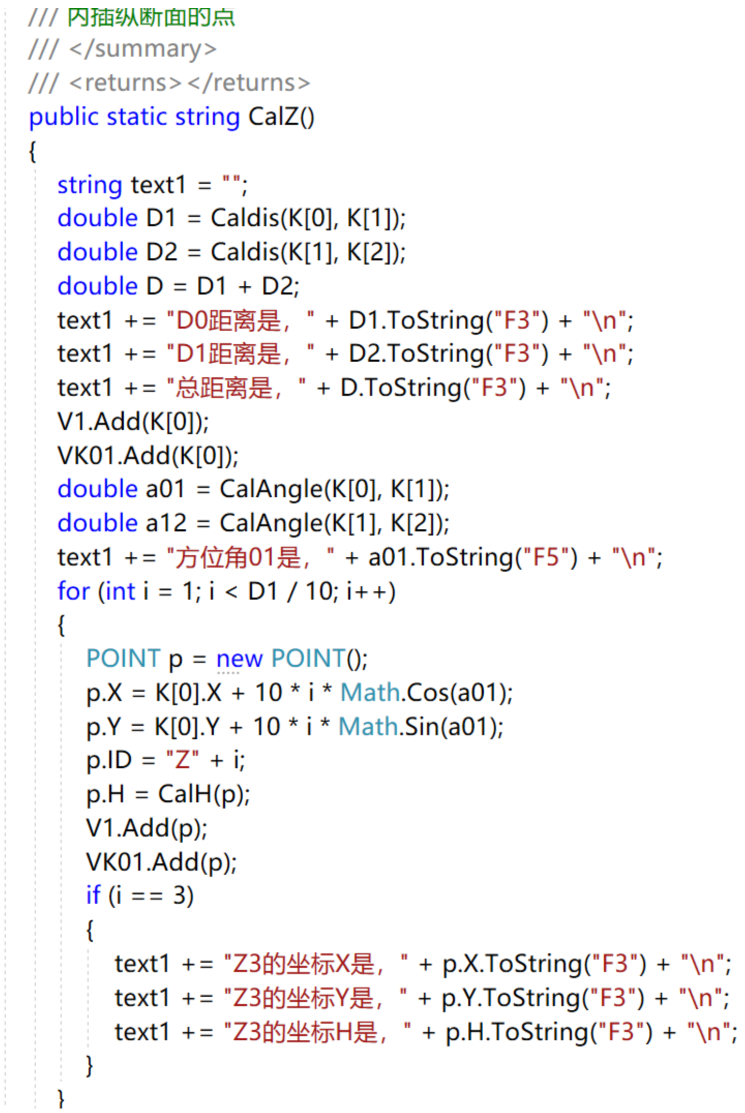


```
namespace Section
{
    class Cal
    {
        public static List<POINT> V1 = new List<POINT>();
        public static List<POINT> VK01 = new List<POINT>();
        public static List<POINT> VK12 = new List<POINT>();
        public static List<POINT> V2 = new List<POINT>();
        public static List<POINT> V3 = new List<POINT>();
        public static List<POINT> p = new List<POINT>();
        public static List<POINT> K = new List<POINT>();
        public static List<POINT> ALL = new List<POINT>();
        public static POINT A = new POINT();
        public static POINT B = new POINT();
        public static double H0;
        public static string text = "";//输出数据
        public static POINT M0 = new POINT();
        public static POINT M1 = new POINT();
```

核心算法源码



```
/// <summary>
/// 计算内插点高程
/// </summary>
/// <param name="a"></param>
/// <returns></returns>
public static double CalH(POINT a)
{
    foreach (POINT p in ALL)
    {
        p.dis = Caldis(a, p);
    }
    ALL.Sort((x, y) => x.dis.CompareTo(y.dis));
    double on = 0;
    double down = 0;
    for (int i = 0; i < 5; i++)
    {
        on += ALL[i].H / ALL[i].dis;
        down += 1 / ALL[i].dis;
    }
    return on / down;
}
```



```
/// 内插纵断面的点
/// </summary>
/// <returns></returns>
public static string CalZ()
{
    string text1 = "";
    double D1 = Caldis(K[0], K[1]);
    double D2 = Caldis(K[1], K[2]);
    double D = D1 + D2;
    text1 += "D0距离是，" + D1.ToString("F3") + "\n";
    text1 += "D1距离是，" + D2.ToString("F3") + "\n";
    text1 += "总距离是，" + D.ToString("F3") + "\n";
    V1.Add(K[0]);
    VK01.Add(K[0]);
    double a01 = CalAngle(K[0], K[1]);
    double a12 = CalAngle(K[1], K[2]);
    text1 += "方位角01是，" + a01.ToString("F5") + "\n";
    for (int i = 1; i < D1 / 10; i++)
    {
        POINT p = new POINT();
        p.X = K[0].X + 10 * i * Math.Cos(a01);
        p.Y = K[0].Y + 10 * i * Math.Sin(a01);
        p.ID = "Z" + i;
        p.H = CalH(p);
        V1.Add(p);
        VK01.Add(p);
        if (i == 3)
        {
            text1 += "Z3的坐标X是，" + p.X.ToString("F3") + "\n";
            text1 += "Z3的坐标Y是，" + p.Y.ToString("F3") + "\n";
            text1 += "Z3的坐标H是，" + p.H.ToString("F3") + "\n";
        }
    }
```

```
/// <summary>
/// 计算横断面
/// </summary>
/// <returns></returns>
public static string CalHENG()
{
    string text2 = "";
    M0.X = (K[0].X + K[1].X) / 2;
    M0.Y = (K[0].Y + K[1].Y) / 2;
    M0.H = CalH(M0);
    M0.ID = "M0";

    M1.X = (K[1].X + K[2].X) / 2;
    M1.Y = (K[1].Y + K[2].Y) / 2;
    M1.H = CalH(M1);
    M1.ID = "M1";
    double M0a = CalAngle(K[0], K[1]) + Math.PI / 2;
    double M1a = CalAngle(K[1], K[2]) + Math.PI / 2;
    int j = 1;
    for (int i = -5; i <= 5; i++)
    {
        POINT p = new POINT();
        if (i == 0)
            V2.Add(M0);
        else
        {
            p.X = M0.X + 5 * i * Math.Cos(M0a);
            p.Y = M0.Y + 5 * i * Math.Sin(M0a);
            p.H = CalH(p);
            p.ID = "Q" + j;
            V2.Add(p);
            j++;
        }
        if (i == -3)
        {
```