```
/*
    FILE: Hanno Haversine.cs

    B - Latitude +N/-S
    L - Longitude +E/-W
    Dec - Declination +N/-S
    GHA - Greenwich hour angle

    LHA - Local hour angle

    Hc - Altitude (calculated)
    Zn - Azimut

    This file contains proprietary information of Andrés Ruiz Gonzalez
    Copying or reproduction without prior written approval is prohibited.
    Andrés Ruiz. San Sebastian - Donostia. Gipuzkoa
    Navigational Algorithms
    Copyright (c) 2015
*/

namespace NavigationalAlgorithms
{
    public class HannoHaversine : Sexagesimal
    {
        public static double Hc( double B, double Dec, double LHA )
        {
            double m, n, a;

            double aB = Math.Abs(B);
            double aDec = Math.Abs(Dec);

            if (Math.Sign(B) == Math.Sign(Dec)) // Same Name
            {
                n = Haversine(aB - aDec);
                m = Haversine(aB + aDec);
            }
            else // Contrary Name
            {
                n = Haversine(aB + aDec);
                m = Haversine(aB - aDec);
            }

            a = Haversine(LHA);

            double hv_ZD = n + (1 - (n + m)) * a;
            double ZD = AHaversine(hv_ZD);

            return (90.0 - ZD);
        }
        public static double Zn( double B, double Dec, double HC, double LHA )
        {
            double m, n, a;

            double aB = Math.Abs(B);
            double aDec = Math.Abs(Dec);

            if (Math.Sign(B) == Math.Sign(Dec)) // Same Name
            {
                a = Haversine(90.0 - aDec);
            }
            else // Contrary Name
            {
                a = Haversine(90.0 + aDec);
            }

            m = Haversine(aB + HC);
            n = Haversine(aB - HC);

            double hv_Z = (a - n) / (1 - (n + m));
            double Z = AHaversine( hv_Z );

            double Zn;

            if (Math.Sign(B) >= 0) // N Latitude
            {
                if (LHA <= 180.0) Zn = 360.0 - Z;
                else Zn = Z; // LHA > 180°
            }
            else // S Latitude
            {
                if (LHA <= 180.0) Zn = 180.0 + Z;
                else Zn = 180.0 - Z; // LHA > 180°
            }

            return ( Zn );
        }
    }
}
```

```
namespace NavigationalAlgorithms
{
    public class HannoHaversine : Sexagesimal
    {
        #region Haversine

        public static double Haversine(double x)
        {
            double sx2 = SIN(x / 2.0);
            return (sx2 * sx2);
        }

        public static double Haversinec(double x)
        {
            return ((1.0 - COS(x)) / 2.0);
        }

        public static double AHaversine(double x)
        {
            return (2.0 * ASIN(Math.Sqrt(x)));
        }

        public static string  HaversineTable()
        {
            string table = "";
            double x;

            //            for (double xd = 0.0; xd < 180.0; xd++)
            for (double xd = 0.0; xd <= 360.0; xd++)
                {
                    for (double xm = 0.0; xm <= 60.0; xm++)
                {
                    x = xd + xm / 60.0;
                    table += xd + "º " + xm + "'\t" + x + "\t" + Haversine(x) + "\r\n";
                }
            }

            return table;
        }

        #endregion

        public static string log = "";
    }
}
```