```
using System;
using System.Collections.Generic;
using System.IO;
using System.Text;

namespace kektura_VL
{
    struct Szakasz
    {
        public String Kiindulo_VL;
        public String Vegpont_VL;
        public Double Hossz_VL;
        public int Emelkedes_VL;
        public int Lejtes_VL;
        public bool Pecset_VL;


        public Szakasz(string[] m)
        {
            Kiindulo_VL = m[0];
            Vegpont_VL = m[1];
            Hossz_VL = double.Parse(m[2]);
            Emelkedes_VL = int.Parse(m[3]);
            Lejtes_VL = int.Parse(m[4]);
            Pecset_VL = m[5] == "i";
        }

        //6. feladat:
        public bool HianyosNev
        {
            get
            {
                if (Pecset_VL)
                {
                    if (Vegpont_VL.Contains("pecsetelohely")) return false;
                    else return true;
                }
                return false;
            }
        }

        public override string ToString()
        {
            string NemHiányosNév = Vegpont_VL;
            if (HianyosNev) NemHiányosNév += " pecsetelohely";
            return String.Format("{0};{1};{2};{3};{4};{5}", Kiindulo_VL, NemHiányosNév,
Hossz_VL, Emelkedes_VL, Lejtes_VL, Pecset_VL ? 'i' : 'n');
        }
    }
    class kektura
    {
        static void Main()
        {
            List<Szakasz> sz = new List<Szakasz>();
            string[] forrás = File.ReadAllLines("kektura.csv");
            int tszfm = int.Parse(forrás[0]); //kiindulópont tengerszint feletti magassága
            for (int i = 1; i < forrás.Length; i++)
            {
```

```
            sz.Add(new Szakasz(forrás[i].Split(';')));
        }

        Console.WriteLine("3. feladat: Szakaszok száma: {0} db", sz.Count);

        // 4. feladat: A túra teljes hossza
        double hossz = 0;
        foreach (var i in sz)
        {
            hossz += i.Hossz_VL;
        }
        Console.WriteLine("4. feladat: A túra teljes hossza: {0} km", hossz);

        // 5. feladat: A legrövidebb szakasz adatai
        int mini = 0;
        for (int i = 1; i < sz.Count; i++)
        {
            if (sz[i].Hossz_VL < sz[mini].Hossz_VL) mini = i;
        }
        Console.WriteLine("5. feladat: A legrövidebb szakasz adatai:");
        Console.WriteLine("\tKezdete: {0}", sz[mini].Kiindulo_VL);
        Console.WriteLine("\tVége: {0}", sz[mini].Vegpont_VL);
        Console.WriteLine("\tTávolság: {0} km", sz[mini].Hossz_VL);

        // 7. feladat: Hiányos állomásnevek
        Console.WriteLine("7. feladat: Hiányos állomásnevek:");
        bool voltHiányos = false;
        for (int i = 0; i < sz.Count; i++)
        {
            if (sz[i].HianyosNev)
            {
                Console.WriteLine("\t{0}", sz[i].Vegpont_VL);
                voltHiányos = true;
            }
        }
        if (!voltHiányos) Console.WriteLine("Nincs hiányos állomásnév!");


        //8. feladat: A túra legmagasabban fekvő végpontja
        int aktMagasság = tszfm + sz[0].Emelkedes_VL - sz[0].Lejtes_VL;
        int maxMagasság = aktMagasság;
        int maxi = 0;
        for (int i = 1; i < sz.Count; i++)
        {
            aktMagasság += sz[i].Emelkedes_VL - sz[i].Lejtes_VL;
            if (aktMagasság > maxMagasság)
            {
                maxMagasság = aktMagasság;
                maxi = i;
            }
        }
        Console.WriteLine("8. feladat: A túra legmagasabban fekvő végpontja:");
        Console.WriteLine("\tA végpont neve: {0}", sz[maxi].Vegpont_VL);
        Console.WriteLine("\tA végpont tengerszint feletti magassága: {0} m",
maxMagasság);
```

```
//9. feladat: kektura2.csv állomány
            List<string> sorok = new List<string>();
            sorok.Add(tszfm.ToString());
            foreach (var i in sz)
            {
                sorok.Add(i.ToString());
            }
            File.WriteAllLines("kektura2.csv", sorok);

            Console.ReadKey();
        }
    }
}
```