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

namespace kektura_TR
{
    struct Szakasz
    {
        public String Kezdet_TR;
        public String Vég_TR;
        public Double Táv_TR;
        public int Emelkedés_TR;
        public int Lejtés_TR;
        public bool PecsételőHely_TR;


        public Szakasz(string[] m)
        {
            Kezdet_TR = m[0];
            Vég_TR = m[1];
            Táv_TR = double.Parse(m[2]);
            Emelkedés_TR = int.Parse(m[3]);
            Lejtés_TR = int.Parse(m[4]);
            PecsételőHely_TR = m[5] == "i";
        }


        public bool HianyosNev
        {
            get
            {
                if (PecsételőHely_TR)
                {
                    if (Vég_TR.Contains("pecsetelohely")) return false;
                    else return true;
                }
                return false;
            }
        }

        public override string ToString()
        {
            string NemHiányosNév = Vég_TR;
            if (HianyosNev) NemHiányosNév += " pecsetelohely";
            return String.Format("{0};{1};{2};{3};{4};{5}", Kezdet_TR, NemHiányosNév,
Táv_TR, Emelkedés_TR, Lejtés_TR, PecsételőHely_TR ? 'i' : 'n');
        }
    }
    class kektura_TR
    {
        static void Main()
        {
            List<Szakasz> sz = new List<Szakasz>();
            string[] forrás = File.ReadAllLines("kektura_TR.csv");
            int tszfm = int.Parse(forrás[0]);
            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);


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


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


        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].Vég_TR);
                voltHiányos = true;
            }
        }
        if (!voltHiányos) Console.WriteLine("Nincs hiányos állomásnév!");



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

```
            List<string> sorok = new List<string>();
            sorok.Add(tszfm.ToString());
            foreach (var i in sz)
            {
                sorok.Add(i.ToString());
            }
            File.WriteAllLines("kektura_TR.csv", sorok);

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