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

namespace kektura_Bakos_Norbert
{
    struct Szakasz
    {         public String
Kezdet;         public String Vég;
public Double Táv;         public
int Emelkedés;         public int
Lejtés;         public bool
PecsételőHely;

        public Szakasz(string[] m)
        {
            Kezdet = m[0];
            Vég = m[1];
            Táv = double.Parse(m[2]);
            Emelkedés = int.Parse(m[3]);
Lejtés = int.Parse(m[4]);
            PecsételőHely = m[5] == "i";
        }

        //6. feladat:
public bool HianyosNev
        {
get
{
                if (PecsételőHely)
                {
                    if (Vég.Contains("pecsetelohely")) return false;
else return true;
                }
                return false;
            }
        }
        public override string ToString()
        {
            string NemHiányosNév = Vég;
            if (HianyosNev) NemHiányosNév += " pecsetelohely";
            return String.Format("{0};{1};{2};{3};{4};{5}", Kezdet, NemHiányosNév, Táv,
Emelkedés, Lejtés, PecsételőHely ? '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.Táv;
            }
            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].Táv < sz[mini].Táv) mini = i;
            }
            Console.WriteLine("5. feladat: A legrövidebb szakasz adatai:");
            Console.WriteLine("\tKezdete: {0}", sz[mini].Kezdet);
            Console.WriteLine("\tVége: {0}", sz[mini].Vég);
            Console.WriteLine("\tTávolság: {0} km", sz[mini].Táv);

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


            //8. feladat: A túra legmagasabban fekvő végpontja
int aktMagasság = tszfm + sz[0].Emelkedés - sz[0].Lejtés;
int maxMagasság = aktMagasság;             int maxi = 0;
            for (int i = 1; i < sz.Count; i++)
            {
                aktMagasság += sz[i].Emelkedés - sz[i].Lejtés;
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);
            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();
        }
    }
}
```