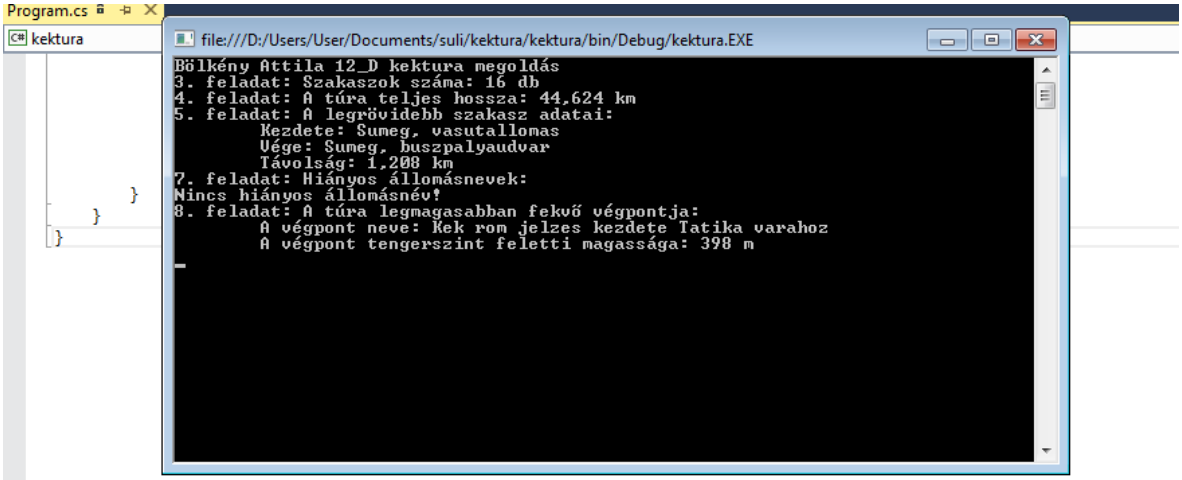
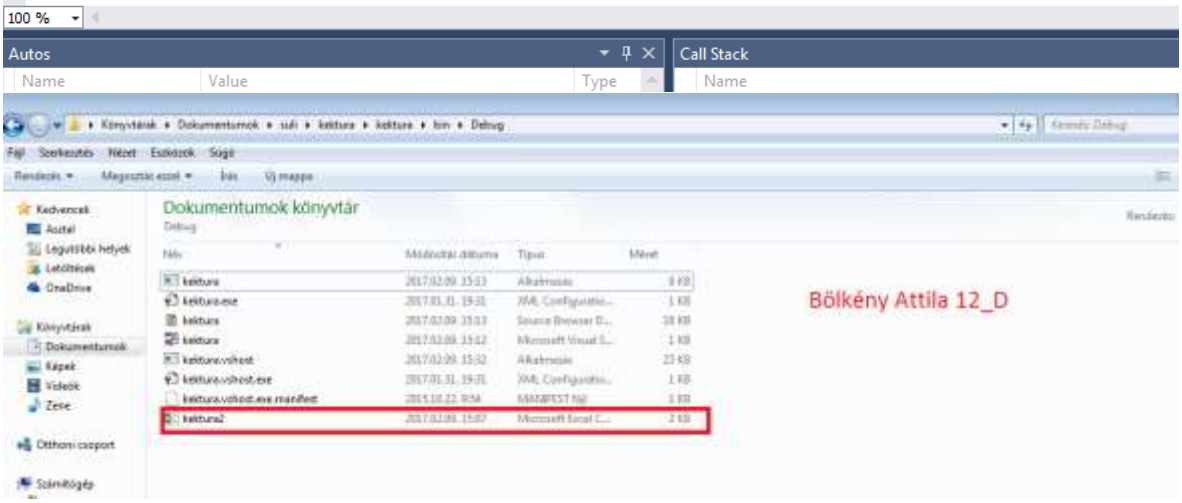
Bölkény Attila 12.D – kektura programozás feladat megoldás






```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
namespace kektura
{
 class Program
 {
 public static bool HianyosNev(string nev, bool pecset)
 {
 Console.WriteLine("Bölkény Attila 12_D kektura megoldas");
 //Feladat 6:
 if (nev.Contains("pecsetelohely") ^ (pecset == true))
 {
 return true;
 }
 else
 {
 return false;
 }
```

```
}
static void Main(string[] args)
{
// Az adatok tárolásához előkészített tömbök:
const int tengerszint = 192;
string[] kiindulas_BA = new string[20];
string[] vegpont_BA = new string[20];
double[] hossz_BA = new double[20];
int[] emel = new int[20];
int[] lejt = new int[20];
bool[] pecset = new bool[20];
int szakasz = 0;
int szamlalo = 0;
int szamlalo2 = 0;
double osszeg_BA = 0;
double legkisebbhossz = 0;
string legkisebbki;
string legkisebbveg;
int legmagasabb;
int legmagasabbindex = 0;
// Fájl megnyitássa.
StreamReader sr = new StreamReader("kektura.csv");
string adat = sr.ReadLine();
// Fájl beolvasása, és adatok tárolása.
while (!sr.EndOfStream)
{
adat = sr.ReadLine();
string[] ertekek = adat.Split(';');
kiindulas_BA[szamlalo] = ertekek[0];
vegpont_BA[szamlalo] = ertekek[1];
hossz_BA[szamlalo] = double.Parse(ertekek[2]);
emel[szamlalo] = Int32.Parse(ertekek[3]);
lejt[szamlalo] = Int32.Parse(ertekek[4]);
if (ertekek[5] == "i")
{
pecset[szamlalo] = true;
}
else
{
pecset[szamlalo] = false;
}
szamlalo++;
}
sr.Close();
//Feladat 3:
while (kiindulas_BA[szamlalo2] != null)
{
szamlalo2++;
szakasz++;
}
Console.WriteLine("3. Feladat: A szakaszok száma: {0} db", szakasz);
//Feladat 4:
for (int i = 0; i < szakasz; i++)
{
osszeg_BA += hossz_BA[i];
}
Console.WriteLine("4. Feladat: A túra teljes hossza: {0} km", osszeg_BA);
//Feladat 5:
legkisebbki = kiindulas_BA[0];
legkisebbveg = vegpont_BA[0];
legkisebbhossz = hossz_BA[0];
```

```
 for (int i = 0; i < szakasz; i++)
 {
 if (hossz_BA[i] < legkisebbhossz)
 {
 legkisebbhossz = hossz_BA[i];
 legkisebbki = kiindulas_BA[i];
legkisebbveg = vegpont_BA[i];
 }
 }
 Console.WriteLine("5. feladat: a legrovidebb szakasz adatai: \n\tkezdete:  \tvege:
{1} \n\tHossza: {2} km", legkisebbki, legkisebbveg, legkisebbhossz);
 //Feladat 7:
 Console.WriteLine("7. Feladat: Hiányos állomásnevek:");
 for (int i = 0; i < szakasz; i++)
 {
 if (HianyosNev(vegpont_BA[i], pecset[i]) == true)
 {
 Console.WriteLine("\t{0}", vegpont_BA[i]);
 }
 else
 {
 if (i == szakasz - 1)
 {
 Console.WriteLine("Nincs hiányos állomásnév!");
 }
else
{
 }
 }
 }
 //Feladat 8:
 Console.WriteLine("8. Feladat: A túra legmagassabban fekvő végpontja:");
 legmagasabb = emel[0];
 for (int i = 0; i < szakasz; i++)
 {
 if (emel[i] > legmagasabb)
 {
 legmagasabbindex = i;
legmagasabb = emel[i];
 }
 }
 Console.WriteLine("\tA végpont neve: {0}", vegpont_BA[legmagasabbindex]);
 legmagasabb = tengerszint;
 for (int i = 0; i < legmagasabbindex + 1; i++)
 {
 legmagasabb += (emel[i] - lejt[i]);
 }
 Console.WriteLine("\tA végpont tengerszint feletti magassága: {0} m",
legmagasabb);
 //Feladat 9:
 StreamWriter sw = new StreamWriter("kektura2.txt");
 for (int i = -1; i < szakasz; i++)
 {
 if (i == -1)
 {
 continue;
 }
 else
 {
 if ((HianyosNev(vegpont_BA[i], pecset[i])) == true)
 {
```

```
 vegpont_BA[i] = vegpont_BA[i] + " " + "pecsetelohely";
 }
 }
 if (i == 0)
 {
 sw.WriteLine(tengerszint);
 }
 else
 {
 }
 if (pecset[i] == true)
 {
 sw.WriteLine(kiindulas_BA[i] + ";" + vegpont_BA[i] + ";" + hossz_BA[i]
+ ";" + emel[i] + ";" + lejt[i] + ";" + "i");
 }
 else
 {
 sw.WriteLine(kiindulas_BA[i] + ";" + vegpont_BA[i] + ";" + hossz_BA[i]
+ ";" + emel[i] + ";" + lejt[i] + ";" + "n");
 }
 }
 sw.Close();
 }
 }
}
```