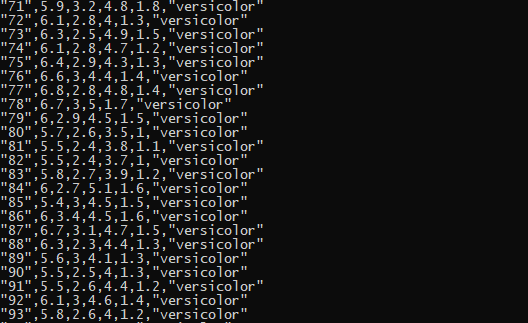
***Assignment 5***



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Runtime.Serialization.Formatters.Binary;

using System.Runtime.Serialization;

using static System.Console;

namespace ConsoleApplication6

{

class Program

{

static BinaryFormatter bf = new BinaryFormatter();

static void Main(string[] args)

{

const string FILENAME = @"iris.csv";

const string serFile = @"iris.txt";

const string planeTxt = @"planeiris.txt";

Species spe = new Species();

FileStream fs = new FileStream(serFile, FileMode.Create, FileAccess.Write);

Read(FILENAME, spe);

bf.Serialize(fs, spe);

fs.Close();

FileStream fileStream = new FileStream(serFile, FileMode.Open, FileAccess.Read);

Write(planeTxt, planeTxt, fileStream);

ReadLine();

}

static void Read(string file, Species species)

{

int i = 0;

Species spe = new Species();

FileStream inFile = new FileStream(file, FileMode.Open, FileAccess.Read);

StreamReader reader = new StreamReader(inFile);

try

{

while (!reader.EndOfStream)

{

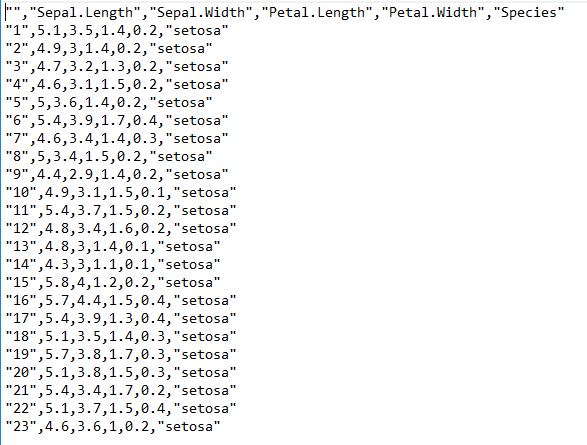
species.Iris.Add(reader.ReadLine());

//Console.WriteLine(species.Iris[i]);

i++;

}

}



catch(Exception)

{

Console.Write("An error occured in Read()");

}

finally

{

reader.Close();

inFile.Close();

}

}

static void Write(string file, string file2, FileStream fs)

{

//Species spe = new Species();

FileStream fileStream = new FileStream(file2, FileMode.Create, FileAccess.Write);

StreamWriter writer = new StreamWriter(fileStream);

FileStream files = new FileStream("versicolor.csv", FileMode.Create, FileAccess.Write);

StreamWriter writer2 = new StreamWriter(files);

try

{

Species spe = (Species)bf.Deserialize(fs);

fs.Close();

if (fileStream.CanWrite)

{

foreach (string s in spe.Iris)

{

writer.WriteLine(s);

if (s.Contains("versicolor"))

{

writer2.WriteLine(s);

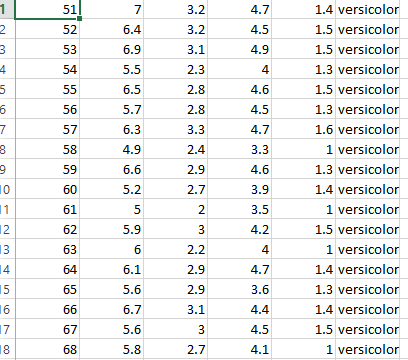
Console.WriteLine(s);

}

}

}

}



catch(Exception)

{

Console.Write("An Error in Write()");

}

finally

{

files.Close();

fileStream.Close();

}

}

}

class Sepal

{

float length;

float width;

public float Length

{

get

{

return length;

}

set

{

length = value;

}

}

public float Width

{

get

{

return width;

}

set

{

width = value;

}

}

}

class Petal

{

float length;

float width;

public float Length

{

get

{

return length;

}

set

{

length = value;

}

}

public float Width

{

get

{

return width;

}

set

{

width = value;

}

}

}

[Serializable]

class Species

{

float length;

float width;

public List<string> Iris = new List<string>();

public float Length

{

get

{

return length;

}

set

{

length = value;

}

}

public float Width

{

get

{

return width;

}

set

{

width = value;

}

}

}

[Serializable]

class Iris

{

public double SepalLength { get; set; }

public double SepalWidth { get; set; }

public double PetalLength { get; set; }

public double PetalWidth { get; set; }

public string species { get; set; }

public string RecordNumber { get; set; }

}

}