using System;  
using System.Collections.Generic;  
using System.IO;  
  
namespace ConsoleProjektH1  
{  
 class Program  
 {  
 private static void Main(string[] args)  
 {  
 var people = new List<Person>  
 {  
 new Person  
 {  
 Age = 33,  
 Name = "Ole",  
 Balance = 1000.11  
 },  
 new Person  
 {  
 Age = 61,  
 Name = "Peter",  
 Balance = 1400.21  
 },  
 new Person  
 {  
 Age = 23,  
 Name = "Jonna",  
 Balance = 800.00  
 },  
 new Person  
 {  
 Age = 19,  
 Name = "Finn",  
 Balance = 1200.50  
 },  
 new Person  
 {  
 Age = 22,  
 Name = "Heidi",  
 Balance = 3000.24  
 },  
 };  
  
 bool isRunning;  
   
 Functions functions = new Functions();  
   
 List<string> inputList;  
   
 try  
 {  
 Console.WriteLine("Hello, welcome to this list of people - Type \"help\" to " +  
 "receive a list of commands");  
 isRunning = true;  
 while (isRunning)  
 {  
 Console.Write(":>");  
 inputList = functions.FilterInput(Console.ReadLine().ToLower());  
 try  
 {  
 functions.HandleCommands(people, inputList, functions, isRunning);  
 }  
 catch (Exception e)  
 {  
 if (inputList[0] == "changeperson" || inputList[0] == "changeage" ||  
 inputList[0] == "changebalance" || inputList[0] == "deleteperson" ||   
 inputList[0] == "addperson")  
 {  
 Console.WriteLine("That person is not on the list, or you entered an incorrect value");  
 }  
 else  
 {  
 Console.WriteLine("Please enter a name");  
 //Console.WriteLine(e);  
 }  
 }  
 }  
 }  
 catch (Exception nfe)  
 {  
 Console.WriteLine(nfe);   
 }   
 }  
 }  
  
 class Functions  
 {  
 // Show the entire list  
 private void ShowAll(List<Person> people)  
 {  
 //var stringList = AlterNameList();  
  
 int pA = 6;  
   
 Console.WriteLine("Name".PadRight(pA) + "Age".PadRight(pA-2) + "Balance".PadRight(pA));  
 foreach (var person in people)  
 {  
 Console.WriteLine(person.Name.PadRight(pA) + person.Age.ToString().PadRight(pA-2) +   
 person.Balance.ToString().PadRight(pA));  
 }  
 }  
  
 // Add a person at the end of the list  
 private void AddPerson(string b)  
 {  
 File.AppendAllText(Environment.CurrentDirectory + "\\NameList.txt", Capitalize(b) + ";");  
 Console.WriteLine("Person added\n");  
 }  
  
 // Remove a person with a specific name  
 private void DeletePerson(string b)  
 {  
 var stringList = AlterNameList();  
 stringList.Remove(Capitalize(b));  
 AppendNames(stringList);  
 Console.WriteLine("Person deleted\n");  
 }  
   
 // Change the person with a specific name, to another name  
 private void Change(List<Person> persons, string a, string b)  
 {  
 var stringList = AlterNameList();  
 stringList[stringList.IndexOf(Capitalize(a))] = Capitalize(b);  
 AppendNames(stringList);  
  
 foreach (var person in persons)  
 {  
 if (person.Name == Capitalize(a))  
 {  
 person.Name = Capitalize(b);  
 }  
 }  
   
 Console.WriteLine("Person changed\n");  
 }  
  
 // Change the person with a specific name, to a different age  
 private void Change(List<Person> persons, string a, int b)  
 {  
 foreach (var person in persons)  
 {  
 if (person.Name == Capitalize(a))  
 {  
 person.Age = b;  
 }  
 }  
   
 Console.WriteLine("Age changed\n");  
 }  
  
 // Change the person with a specific name, to a different balance  
 private void Change(List<Person> persons, string a, double b)  
 {  
 foreach (var person in persons)  
 {  
 if (person.Name == Capitalize(a))  
 {  
 person.Balance = b;  
 }  
 }  
   
 Console.WriteLine("Balance changed\n");  
 }  
   
 // Append the names from the .txt-file  
 private void AppendNames(List<string> stringList)  
 {  
 File.WriteAllText(Environment.CurrentDirectory + "\\NameList.txt", "");  
  
 foreach (var name in stringList)  
 {  
 File.AppendAllText(Environment.CurrentDirectory + "\\NameList.txt", Capitalize(name) + ";");  
 }  
 }  
  
 // Capitalize the first letter in a string / char array  
 private string Capitalize(string a)  
 {  
 if (a[0] != char.ToUpper(a[0]))  
 {  
 var newCharArray = a.ToCharArray();  
 if (a != "")  
 {  
 newCharArray[0] = char.ToUpper(a[0]);  
 }  
 return new string(newCharArray).Replace(" ", "");  
 }  
 else  
 {  
 return a.Replace(" ", "");  
 }  
   
 }  
  
 // Fetch the list of names for alteration  
 private List<string> AlterNameList ()  
 {  
 var content = File.ReadAllText(Environment.CurrentDirectory + "\\NameList.txt");  
 var stringList = new List<string>();  
  
 foreach (var name in content.Split(';'))  
 {  
 if (name != "")  
 {  
 stringList.Add(Capitalize(name));  
 }  
 }  
  
 return stringList;  
 }  
  
 public List<string> FilterInput(string input)  
 {  
 return new List<string>(input.Split(new[] {" "}, StringSplitOptions.RemoveEmptyEntries));  
 }  
   
 public void HandleCommands(List<Person> people, List<string> inputList, Functions functions, bool isRunning)  
 {  
 switch (inputList[0])  
 {  
 case "showall":  
 functions.ShowAll(people);  
 break;  
 case "addperson":  
 functions.AddPerson(inputList[1]);  
 break;  
 case "deleteperson":  
 functions.DeletePerson(inputList[1]);  
 break;  
 case "changeperson":  
 functions.Change(people, inputList[1], inputList[2]);  
 break;  
 case "changeage":  
 functions.Change(people, inputList[1], int.Parse(inputList[2]));  
 break;  
 case "changebalance":  
 functions.Change(people, inputList[1], double.Parse(inputList[2]));  
 break;  
 case "clear":  
 Console.Clear();  
 Console.WriteLine("Hello, welcome to this list of people - Type \"help\" to " +  
 "receive a list of commands");  
 break;  
 case "quit":  
 isRunning = false;  
 break;  
 case "help":  
 Console.WriteLine("These are the available commands:");  
 Console.WriteLine("\"showall\" - Shows the current list of people");  
 Console.WriteLine("\"addperson\" <name> - Adds a person to the current list of people");  
 Console.WriteLine("\"deleteperson\" <name> - Deletes a person from the current list of people");  
 Console.WriteLine("\"changeperson\" <name1> <name2> - changes the name of a person from the " +  
 "current list of people");  
 Console.WriteLine("\"clear\" - Clears the console");  
 Console.WriteLine("\"quit\" - Quits the console");  
 Console.WriteLine("\"help\" - Shows this list of available commands");  
 break;  
 default:  
 Console.WriteLine("That is not a command");  
 break;  
 }  
 }  
 }  
  
 class Person  
 {  
 public int Age { get; set; }  
 public double Balance { get; set; }  
 public string Name { get; set; }  
 }  
}