Листинг лабораторной работы №8

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using System.Runtime.Serialization.Formatters.Binary;

namespace struct\_files

{

[Serializable]

struct Student

{

public string name;

public int kurs;

public double rating;

public Student(string s, int k, double r)

{

name = s;

kurs = k;

rating = r;

}

public void PrintStudent()

{

Console.WriteLine("{0}, курс {1}, рейтинг {2}", name, kurs, rating);

}

public void MakeStudent()

{

Console.Write("Введите имя: ");

name = Console.ReadLine();

bool ok = false;

do

{

Console.Write("Введите курс: ");

ok = Int32.TryParse(Console.ReadLine(), out kurs);

} while (!ok);

do

{

Console.Write("Введите рейтинг: ");

ok = Double.TryParse(Console.ReadLine(), out rating);

} while (!ok);

}

}

class Program

{

const int maxSize=10;

static int count=0;

static void SaveFile(Student[] arr)

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Append);

BinaryFormatter bf = new BinaryFormatter();

for (int i = 0; i < count; i++)

bf.Serialize(f1, arr[i]);

f1.Close();

Console.WriteLine(" Save....");

}

static void PrintFile()

{

int count=0;

FileStream f1 = new FileStream("StudBD.dat", FileMode.Open);

BinaryFormatter bf = new BinaryFormatter();

try

{

while (true)

{

Student s = (Student)bf.Deserialize(f1);

s.PrintStudent();

count++;

}

}

catch (Exception e)

{

if (count == 0) Console.WriteLine("Файл не создан!");

else Console.WriteLine("======================================================================");

f1.Close();

}

}

static void AddToEndOfFile( Student s)

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Append);

BinaryFormatter bf = new BinaryFormatter();

bf.Serialize(f1,s);

f1.Close();

Console.WriteLine(" Save....");

}

static void AddFile( Student s,int nom=0)

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Open);

BinaryFormatter bf = new BinaryFormatter();

FileStream temp = new FileStream("temp", FileMode.Create);

BinaryFormatter bfTemp = new BinaryFormatter();

bool add=false;

if (nom == 0)

{

bfTemp.Serialize(temp, s);

add = true;

}

int count = 1;

try

{

while (true)

{

Student t = (Student)bf.Deserialize(f1);

if(!add)

if (count == nom)

{

bfTemp.Serialize(temp, s);

add = true;

}

bfTemp.Serialize(temp, t);

count++;

}

}

catch (Exception e)

{

if(add)

Console.WriteLine("Add and Save...");

else

Console.WriteLine("There is not record with such number");

f1.Close();

temp.Close();

File.Delete("StudBD.dat");

File.Move("temp", "StudBD.dat");

}

}

static void ClearFile()

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Create);

f1.Close();

}

static void DelByName(string name)

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Open);

BinaryFormatter bf = new BinaryFormatter();

FileStream temp = new FileStream("temp", FileMode.Create);

BinaryFormatter bfTemp = new BinaryFormatter();

bool del = false;

try

{

while (true)

{

Student t = (Student)bf.Deserialize(f1);

if (!t.name.Equals(name))

{

bfTemp.Serialize(temp, t);

}

else

{

t.PrintStudent();

Console.WriteLine(" is deleted...");

del = true;

}

}

}

catch (Exception e)

{

if (!del)

Console.WriteLine("There is not record with such name");

f1.Close();

temp.Close();

File.Delete("StudBD.dat");

File.Move("temp", "StudBD.dat");

}

}

static void DelByNumber(int number)

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Open);

BinaryFormatter bf = new BinaryFormatter();

FileStream temp = new FileStream("temp", FileMode.Create);

BinaryFormatter bfTemp = new BinaryFormatter();

bool del = false;

int count = 1;

try

{

while (true)

{

Student t = (Student)bf.Deserialize(f1);

if (count!=number)

{

bfTemp.Serialize(temp, t);

}

else

{

t.PrintStudent();

Console.WriteLine(" is deleted...");

del = true;

}

count++;

}

}

catch (Exception e)

{

if (!del)

Console.WriteLine("There is not record with such number");

f1.Close();

temp.Close();

File.Delete("StudBD.dat");

File.Move("temp", "StudBD.dat");

}

}

static void CorrectFile(string name)

{

FileStream f1 = new FileStream("StudBD.dat", FileMode.Open);

BinaryFormatter bf = new BinaryFormatter();

FileStream temp = new FileStream("temp", FileMode.Create);

BinaryFormatter bfTemp = new BinaryFormatter();

bool correct = false;

try

{

while (true)

{

Student t = (Student)bf.Deserialize(f1);

if (!t.name.Equals(name))

{

bfTemp.Serialize(temp, t);

}

else

{

Console.WriteLine("Enter new inforation, please:");

Student s = new Student();

s.MakeStudent();

bfTemp.Serialize(temp, s);

correct = true;

}

}

}

catch (Exception e)

{

if (!correct)

Console.WriteLine("There is not record with such name");

f1.Close();

temp.Close();

File.Delete("StudBD.dat");

File.Move("temp", "StudBD.dat");

}

}

static void PrintMenu()

{

Console.WriteLine("1. Ввести информацию о Студенте");

Console.WriteLine("2. Вывести информацию о Студенте");

Console.WriteLine("3. Сохранить записи в файл");

Console.WriteLine("4. Печать файла");

Console.WriteLine("5. Добавить запись");

Console.WriteLine("6. Удалить запись");

Console.WriteLine("7. Корректировать запись");

Console.WriteLine("8. Очистить файл");

Console.WriteLine("9. Выход");

}

static void PrintSaveMenu()

{

Console.WriteLine("1. Сохранить в начало файла");

Console.WriteLine("2. Сохранить в конец файла");

Console.WriteLine("3. Сохранить запись с заданным номером");

Console.WriteLine("4. Назад");

}

static void PrintDeleteMenu()

{

Console.WriteLine("1.Удалить с заданным номером");

Console.WriteLine("2. Удалить с заданным ключом");

Console.WriteLine("3. Назад");

}

static void Main(string[] args)

{

int answer=0;

Student []arr=new Student[maxSize];

do

{

PrintMenu();

bool ok=Int32.TryParse(Console.ReadLine(), out answer);

if(!ok)answer=9;

switch(answer)

{

case 1:

{

arr[count++].MakeStudent();

if(count==maxSize)

{

SaveFile(arr);

count=0;

}

break;

}

case 2:

{

foreach(Student s in arr)

s.PrintStudent();

break;

}

case 3:

{

SaveFile(arr);

count = 0;

break;

}

case 4:

{

PrintFile();

break;

}

case 5:

{

int check;

do

{

PrintSaveMenu();

check=Convert.ToInt32(Console.ReadLine());

switch (check)

{

case 1:

{

Student s = new Student();

s.MakeStudent();

AddFile(s);

break;

}

case 2:

{

Student s = new Student();

s.MakeStudent();

AddToEndOfFile(s);

break;

}

case 3:

{

Student s = new Student();

s.MakeStudent();

int number = 0;

do

{

Console.Write("Введите номер: ");

ok = Int32.TryParse(Console.ReadLine(), out number);

} while (!ok);

AddFile(s, number);

break;

}

}

}while(check<4);

break;

}

case 6://удаление

{

int check;

do

{

PrintDeleteMenu();

check = Convert.ToInt32(Console.ReadLine());

switch (check)

{

case 1:

{

int number = 0;

do

{

Console.Write("Введите номер: ");

ok = Int32.TryParse(Console.ReadLine(), out number);

} while (!ok);

DelByNumber(number);

break;

}

case 2:

{

string name="";

Console.Write("Введите имя: ");

name = Console.ReadLine();

DelByName(name);

break;

}

}

} while (check < 3);

break;

}

case 7: //correct

{

string name="";

Console.Write("Введите имя: ");

name = Console.ReadLine();

CorrectFile(name);

break;

}

case 8:

{

ClearFile();

Console.WriteLine("File is empty");

break;

}

}

} while(answer<9);

}

}

}