Государственное общеобразовательное учреждение

Высшего профессионального образования

«Дальневосточный государственный университет путей сообщения»

Кафедра: «Информационные технологии

и системы»

Лабораторная работа №1

Выполнил: Суетин А.А.

Группа: 23К

Проверил: Водолажский А.А.

Потапов И.И.

Г. Хабаровск

2013

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Лабораторная\_работа\_\_1

{

interface IRateAndCopy

{

object DeepCopy();

int Rating { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Лабораторная\_работа\_\_1

{

class Person : IRateAndCopy

{

private string Family;

public string family

{

get { return Family; }

set { Family = value; }

}

private string Name;

public string name

{

get { return Name; }

set { Name = value; }

}

private DateTime Data;

public DateTime data

{

get { return Data; }

set { Data = value; }

}

public int ChangeYear

{

get { return Data.Year; }

set { Data = new DateTime(value, Data.Month, Data.Day); }

}

public Person(string Family, string Name, DateTime Data)

{

this.Family = Family;

this.Name = Name;

this.Data = Data;

}

public Person()

{

Family = "";

Name = "";

Data = DateTime.Now;

}

public override string ToString()

{

return "Фамилия:" + Family + " Имя:" + Name + " День рождения:" + Data.ToShortDateString();

}

public virtual string ToShortString()

{

return "Фамилия:" + Family + " Имя:" + Name;

}

public override bool Equals(object obj)

{

if ((obj == null) || (GetType() != obj.GetType()))

return false;

else

{

Person person = (Person)obj;

return (person.Family == Family) && (person.Name == Name) && (person.Data == Data);

}

}

public static bool operator == (Person person1, Person person2)

{

return person1.Equals(person2);

}

public static bool operator != (Person person1, Person person2)

{

return !person1.Equals(person2);

}

public override int GetHashCode()

{

return this.Family.GetHashCode() ^ this.Name.GetHashCode() ^ this.Data.GetHashCode();

}

public virtual object DeepCopy()

{

Person PersonCopy = new Person();

PersonCopy.Family = Family;

PersonCopy.Name = Name;

PersonCopy.Data = Data;

return (object)PersonCopy;

}

public int Rating { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Лабораторная\_работа\_\_1

{

class Article : IRateAndCopy

{

public Person person;

public string ArticleName;

public double rat;

public Article()

{

person = new Person();

ArticleName = "";

rat = 0;

}

public Article(Person author, string name, double rating)

{

person = author;

ArticleName = name;

rat = rating;

}

public override string ToString()

{

return "Название статьи:" + ArticleName + " Рейтинг:" + rat + " " + person.ToString();

}

public int Rating { get; set; }

public virtual object DeepCopy()

{

Article art = new Article();

art.person = person;

art.rat = rat;

art.ArticleName = ArticleName;

return (object)art;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Лабораторная\_работа\_\_1

{

class Edition

{

protected string publication;

public string Publication

{

get { return publication; }

set { publication = value; }

}

protected DateTime DateWrite;

public DateTime datewrite

{

get { return DateWrite; }

set { DateWrite = value; }

}

protected int circulation;

public int Circulation

{

get { return circulation; }

set

{

if (value < 0)

throw new ArgumentOutOfRangeException("Введено ошибочное значение тиража. Тираж не может быть отрицательным числом");

else

circulation = value;

}

}

protected Frequency freq;

public Frequency FR

{

get { return freq; }

set { freq = value; }

}

public Edition()

{

publication = "";

DateWrite = DateTime.Now;

circulation = 0;

}

public Edition(string publicat, DateTime write, int circulat)

{

publication = publicat;

DateWrite = write;

circulation = circulat;

}

public Edition(string publicat, Frequency Freq, DateTime write, int circulat)

{

publication = publicat;

freq = Freq;

DateWrite = write;

circulation = circulat;

}

public virtual object DeepCopy()

{

Edition edition = new Edition();

edition.circulation = circulation;

edition.DateWrite = DateWrite;

edition.publication = publication;

return (object) edition;

}

public override bool Equals(object obj)

{

if ((obj == null) || (GetType() != obj.GetType()))

return false;

else

{

Edition edition = (Edition)obj;

return (edition.publication==publication)&&(edition.DateWrite==DateWrite)&&(edition.circulation==circulation);

}

}

public static bool operator == (Edition edition1,Edition edition2)

{

return edition1.Equals(edition2);

}

public static bool operator !=(Edition edition1, Edition edition2)

{

return !edition1.Equals(edition2);

}

public override int GetHashCode()

{

return this.publication.GetHashCode() ^ this.DateWrite.GetHashCode() ^ this.circulation.GetHashCode();

}

public override string ToString()

{

return "Название:" + publication + " Дата публикации:" + DateWrite.ToShortDateString() + " Тираж:" + circulation;

}

}

}

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Лабораторная\_работа\_\_1

{

class Magazine:Edition, IRateAndCopy

{

private ArrayList editor = new ArrayList();

public ArrayList Editor

{

get

{

ArrayList al = new ArrayList();

foreach (object i in editor)

al.Add(((Person)i).DeepCopy());

return al;

}

set

{

ArrayList al = new ArrayList();

foreach (object i in value)

al.Add(((Person)i).DeepCopy());

editor = al;

}

}

Article[] articles = new Article[0];

public Article[] article

{

get

{

Article[] art = new Article[articles.Length];

for (int i = 0; i < art.Length; i++)

art[i] = (Article)articles[i].DeepCopy();

return art;

}

set

{

Article[] art = new Article[value.Length];

for (int i = 0; i < art.Length; i++)

art[i] = (Article)value[i].DeepCopy();

articles = art;

}

}

public double SredRating

{

get

{

double result = 0;

for (int i = 0; i < articles.Length; i++)

result = result + articles[i].rat;

result = result / articles.Length;

return result;

}

}

public Magazine(string publicat, Frequency Freq, DateTime dw, int circul)

{

publication = publicat;

freq = Freq;

DateWrite = dw;

circulation = circul;

}

public Magazine():base()

{

freq = 0;

}

public void AddArticles(params Article[] art)

{

int q = articles.Length;

Array.Resize<Article>(ref articles, q + art.Length);

for (int i = q, j = 0; i < articles.Length; i++, j++)

articles[i] = art[j];

}

public void AddEditors(params Person[] per)

{

for (int i = 0; i < per.Length; i++)

editor.Add(per[i]);

}

public override string ToString()

{

string str = base.ToString() + " " + freq + " " + "Cписок статей:\n";

foreach (var i in articles)

str += ((Article)i).ToString() + "\n";

str += "Редакторы\n";

foreach (var i in editor)

str+=((Person)i).ToString()+"\n";

str += "Средний рейтинг: " + SredRating;

return str;

}

public virtual string ToShortString()

{

return base.ToString() + " " + freq + " " + "Средний рейтинг:" + SredRating;

}

public override object DeepCopy()

{

Magazine mag = new Magazine(publication, freq, DateWrite, circulation);

mag.editor = Editor;

mag.articles = article;

return (object)mag;

}

public int Rating { get; set; }

public Edition edition

{

get { return new Edition(publication, freq, DateWrite, circulation); }

set

{

publication = value.Publication;

freq = value.FR;

DateWrite = value.datewrite;

circulation = Circulation;

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Лабораторная\_работа\_\_1

{

enum Frequency { Weekly, Monthly, Yearly };

class Program

{

static void Main(string[] args)

{

Edition ed = new Edition("Эхо Сочи", DateTime.Now, 100);

Magazine magaz = new Magazine(ed.Publication, Frequency.Weekly, ed.datewrite, ed.Circulation);

Console.WriteLine(magaz.ToShortString());

magaz.AddArticles(new Article(new Person("Suetin", "Artyom", new DateTime(1994, 8, 26)),"Дальневосточный потоп",5));

magaz.AddArticles(new Article(new Person("Potapov", "Igor", new DateTime(1966, 6, 6)),"Исследование русла реки Амур",3));

magaz.AddEditors(new Person("Ronaldo", "Cristiano", new DateTime(1985, 2, 5)));

Console.WriteLine(magaz.ToString());

Edition edition1 = new Edition("Сочи Олимпийский",new DateTime(1994,8,26),5);

Edition edition2 = new Edition("Сочи Олимпийский", new DateTime(1994, 8, 26), 5);

if ((!object.ReferenceEquals(edition1, edition2)) && (edition1 == edition2))

Console.WriteLine(edition1.GetHashCode() + "\t" + edition2.GetHashCode());

try

{

ed.Circulation = -10;

}

catch (ArgumentOutOfRangeException aoore)

{

Console.WriteLine(aoore.Message);

}

Magazine magaz1 = (Magazine)magaz.DeepCopy();

magaz.Publication = "Эхо Москвы";

Console.WriteLine(magaz.ToString());

Console.WriteLine(magaz1.ToString());

Console.Write("Введите параметр рейтинга: ");

int raiting=int.Parse(Console.ReadLine());

foreach (var i in magaz.article)

{

if (i.rat>raiting)

Console.WriteLine(i.ToString());

}

Console.WriteLine(magaz.edition);

Console.Write("Введите подстроку: ");

string find=Console.ReadLine();

foreach (var i in magaz.article)

{

if (i.ArticleName.Contains(find))

Console.WriteLine(i.ToString());

}

}

}

}