Федеральное Государственное образовательное учреждение высшего

профессионального образования

«Дальневосточный государственный университет

путей сообщения»

Кафедра «Информационные технологии и системы»

Лабораторная работа№1

«Наследование»

Вариант 2.

Выполнил:Молдованова,23К

Проверил: Потапов И. И.

Водолажский А.А

Хабаровск

2013

**Требования к работе.**

Реализовать классы:

1. Класс **Person** (описывает персональные данные объекта)
2. Класс **Article** (производит работу с данными о статьях и их авторах)
3. Класс **Edition** (работает с данными об издании)

4. Класс **Magazin**e-производный от **Edition** , включает в себя следующие поля:

* закрытое поле типа **Frequency** с информацией о периодичности выхода журнала;
* закрытое поле типа **System.Collections.ArrayList** со списком редакторов журнала
* (объектов типа Person);
* закрытое поле типа **Article[]** со списком статей в журнале

а также:

* метод void **AddArticles** (params Article[]) для добавления элементов в список статей
* в журнале;
* метод void **AddEditors** (params Person[]) для добавления элементов в список
* редакторов;

5.Класс **Program**: в методе **Main** согласно варианту задания необходимо:

* Создать один объект типа Magazine, преобразовать данные в текстовый вид с

помощью метода ToShortString() и вывести данные.

* Присвоить значения всем определенным в типе Magazine свойствам,преобразовать данные в текстовый вид с помощью метода ToString() и
* вывести данные.
* C помощью метода AddArticles(params Article[]) добавить элементы в список статей и вывести данные объекта Magazine, используя метод ToString().
* Создать два объекта типа Edition с совпадающими данными и проверить, что ссылки на объекты не равны, а объекты равны, вывести значения хэш-кодов для объектов.
* В блоке try/catch присвоить свойству с тиражом издания некорректное значение, в обработчике исключения вывести сообщение, переданное через объект-исключение.
* Создать объект типа Magazine, добавить элементы в редакторов журнала и вывести данные объекта Magazine.
* Вывести значение свойства типа Edition для объекта типа Magazine.
* С помощью метода DeepCopy() создать полную копию объекта Magazine.
* С помощью оператора foreach вывести список всех статей с рейтингом
* больше некоторого заданного значения.
* С помощью оператора foreach вывести список статей, в названии которых есть
* заданная строка.

**Листинг программы.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Data;

namespace ConsoleApplication7

{

enum Frequency

{ Weekly, Monthly, Yearly }

interface IRateAndCopy

{

object DeepCopy();

double Rating { get; set; }

}

class Person:IRateAndCopy

{

public string Family;

public string Name;

private DateTime born;

private double R;

public Person(string F, string N, DateTime b)

{

Family = F;

Name = N;

born = b;

}

public Person()

{

Family = "Молдованова";

Name = "Лариса";

born = new DateTime(1993,11,22);

}

public double Rating

{

get

{

return R;

}

set

{

R = value;

}

}

public string Fam

{

get

{

return Family;

}

set { Family = value; }

}

public string Nam

{

get

{

return Name;

}

set { Name = value; }

}

public DateTime Date

{

get

{

return born;

}

set { born = value; }

}

public int DataRoz

{

get

{

return born.Year;

}

set

{

born = new DateTime(value,born.Month,born.Day) ;

}

}

public virtual string ToShortString()

{

return Name + Family+"\n";

}

public override string ToString()

{

return ("\n\n"+ "Имя: "+ Name +"\n"+"Фамилия: "+Family +"\n" +"Дата рождения: "+born).ToString();

}

public override bool Equals(object obj)

{

Person per = (Person)obj;

if ((per.Family == this.Family) && (per.Name == this.Name) && (per.born == this.born))

return true;

else

return false;

}

public static bool operator ==(Person per1, Person per2)

{

return per1.Equals(per2);

}

public static bool operator !=(Person per1, Person per2)

{

return !per1.Equals(per2);

}

public override int GetHashCode()

{

return Name.GetHashCode() ^ Family.GetHashCode() ^ born.GetHashCode();

}

public object DeepCopy()

{

Person p = new Person();

p.Name = Name;

p.Family = Family;

p.born = new DateTime(born.Year, born.Month, born.Day);

return (object)p;

}

}

}

**Класс ARTICLE**

class Article:IRateAndCopy

{

public Person Author//данные автора статьи

{ get; set;}

public string Title//названия статьи;

{ get; set; }

public double Rating//для рейтинга статьи.

{ get; set; }

public Article(Person a, string t, double r)

{

Author=a;

Title=t;

Rating=r;

}

public Article()

{

Author = new Person("Michailov", "Andrei", new DateTime(1980, 03, 22));

Title = "Information society";

Rating = 4.5;

}

public override string ToString()

{

return ("Автор статьи "+ Author+ "\n" + "Название статьи: "+ Title+ "\n" + "Оценка рейтинга: "+ Rating+"\n\n").ToString();

}

public virtual object DeepCopy()

{

Article art = new Article();

art.Author = Author;

art.Title = Title;

art.Rating = Rating;

return (object)art;

}

public override bool Equals(object obj)

{

Article A = (Article)obj;

if ((A.Author == this.Author) && (A.Title == this.Title) && (A.Rating == this.Rating))

return true;

else

return false;

}

public static bool operator ==(Article ar1, Article ar2)

{

return ar1.Equals(ar2);

}

public static bool operator !=(Article ar1, Article ar2)

{

return !ar1.Equals(ar2);

}

public override int GetHashCode()

{

return Author.GetHashCode() ^ Title.GetHashCode() ^ Rating.GetHashCode();

}

}

}

**Класс EDITION**

class Edition:Article

{

protected Frequency FC;

protected string Publication;

protected DateTime Data;

protected int edition;

public Edition(string P, DateTime D, int e)

{

Publication = P;

Data = D;

edition = e;

}

public Edition(string p, DateTime DT, int et, Frequency fr)

{

Publication = p;

Data = DT;

edition = et;

FC = fr;

}

public Edition()

{

Publication = "Аврора";

Data = new DateTime(2013, 10, 10);

edition = 2500;

FC = 0;

}

public string Public

{

get

{

return Publication;

}

set

{

Publication = value;

}

}

public DateTime Date

{

get

{

return Data;

}

set

{

Data = value;

}

}

public int edit

{

get

{

return edition;

}

set

{

if (value < 0)

throw new System.FormatException("Тираж не может иметь данный формат, значения свойства должны быть положительны");

else edition = value;

}

}

new public virtual object DeepCopy()

{

Edition ed= new Edition();

ed.Publication = Publication;

ed.Data = new DateTime(Data.Year, Data.Month, Data.Day);

ed.edition = edition;

return (object)ed;

}

public override bool Equals(object obj)

{

Edition ED = (Edition)obj;

if ((ED.Publication == this.Publication) && (ED.edition == this.edition) && (ED.Data == this.Data))

return true;

else

return false;

}

public static bool operator ==(Edition e1, Edition e2)

{

return e1.Equals(e2);

}

public static bool operator !=(Edition e1, Edition e2)

{

return !e1.Equals(e2);

}

public override int GetHashCode()

{

return (int)Publication.GetHashCode() ^ edition.GetHashCode() ^ Data.GetHashCode();

}

public override string ToString()

{

return ("Издание: "+ Publication + "\n" + "Тираж издания - "+ edition + " шт" + "\n"+ "Дата выхода издания:"+ Data).ToString();

}

}

}

**Класс MAGAZINE**

class Magazine:Edition

{

private Frequency period;

private ArrayList List = new ArrayList();

private Article[] art = new Article[0];

public Magazine(string str, Frequency fr, DateTime dt, int t)

{

period = fr;

Publication = str;

Data = dt;

edition = t;

}

public Magazine()

{

period = (Frequency)1;

Publication = "Star";

Data = new DateTime(2013, 02, 03);

edition = 200;

}

public ArrayList AR

{

get

{

return List;

}

set{List=value;}

}

public Article[] AL

{

get

{

return art;

}

set

{

art = value;

}

}

public Edition EDIT

{

get

{

Edition E = new Edition();

E.Public = Public;

E.Date = new DateTime(Data.Year,Data.Month,Data.Day);

E.edit = edit;

return E;

}

set

{

Edition d = new Edition();

Public = d.Public;

Data = d.Date;

edit = d.edit;

}

}

public double SR

{

get

{

double ball = 0;

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

{

ball = ball + art[i].Rating;

}

ball = ball / art.Length;

return ball;

}

}

public void AddArticles(params Article[] list)

{

Article[] tmp = new Article[art.Length+list.Length];

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

tmp[i]=art[i];

tmp[art.Length] = list[0];

art = tmp;

}

public void AddEditors(params Person[] list1)

{

for (int i = 0; i < list1.Length; i++)

List.Add(list1[i]);

}

public override string ToString()

{

string redact = "";

string article = "";

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

redact = redact + art[i].ToString();

for (int i = 0; i < List.Count; i++)

article = article + List[i].ToString();

return redact + article.ToString();

}

public override int GetHashCode()

{

return period.GetHashCode() ^ Rating.GetHashCode();

}

new public object DeepCopy()

{

Magazine a = new Magazine();

Edition c = new Edition();

a.period = period;

c.Date = Date;

c.edit = edit;

c.Public = Public;

for (int i = 0; i < List.Count; i++)

{

a.List.Add(((Article)List[i].DeepCopy());

}

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

{

a.AddArticles((Edition)c.DeepCopy());

}

return (object)a;

}

public override bool Equals(object obj)

{

Magazine p = (Magazine) obj;

if ((p.period == this.period) &&

(p.List.Equals(this.List)) &&

(p.art.Equals(this.art)))

return true;

else return false;

}

public static bool operator ==(Magazine a, Magazine b)

{

return a.Equals(b);

}

public static bool operator !=(Magazine a, Magazine b)

{

return !(a.Equals(b));

}

public virtual string ToShortString()

{

return "Период публикации журнала - "+ period.ToString() + "\n" + "Оценка рейтинга = "+ Rating.ToString();

}

}

}

**Класс PROGRAM**

class Program

{

static void Main(string[] args)

{

Magazine a = new Magazine();

Magazine b = new Magazine();

Console.WriteLine(a.ToShortString());

Console.ReadLine();

a.Public = "Байкал";

a.Date = new DateTime(2013, 10, 05);

a.edit = 2300;

Console.WriteLine(a.EDIT.ToString());

Console.ReadLine();

Article e = new Article(new Person("Бородина","Наталья",new DateTime(1990,02,12)),"Апостол",5);

Article e1 = new Article(new Person("Молдованова", "Лариса",new DateTime(1990, 10, 11)),"Вести",4.5);

a.AddArticles(e);

a.AddArticles(e1);

Console.WriteLine("СТАТЬИ:");

Console.WriteLine(a.ToString());

Console.ReadLine();

Person et = new Person("Никитенко", "Андрей", new DateTime(1993, 08, 04));

Person et1 = new Person("Никитин", "Михаил", new DateTime(1980, 03, 04));

b.AddEditors(et);

b.AddEditors(et1);

Console.WriteLine("РЕДАКТОРЫ:");

Console.WriteLine(b.ToString()+"\n\n");

Console.WriteLine("КОПИЯ ОБЪЕКТА");

Console.WriteLine(a.DeepCopy());

Console.WriteLine("------------------------------------------------------");

Console.ReadLine();

Console.WriteLine("CРАВНЕНИЕ ССЫЛОК СЛЕДУЮЩИХ ОБЪЕКТОВ:");

Console.WriteLine();

Edition ed1 = new Edition("Домострой",new DateTime(2009, 10, 07),1270);

Edition ed2 = new Edition("Домострой",new DateTime(2009, 10, 07),1270);

Console.WriteLine(ed1.ToString()+"\n");

Console.WriteLine(ed2.ToString()+"\n\n");

ReferenceEquals((object)ed1, (object)ed2);

if (ed1 == ed2)

Console.WriteLine("ВЫВОД: Объекты идентичны"+"\n");

else

Console.WriteLine("ВЫВОД: Объекты не идентичны"+"\n");

Console.WriteLine("Хэш-код первого объекта: " +"\n"+ ed1.GetHashCode() + "\n"+ "Хэш-код второго объекта: " +"\n"+ ed2.GetHashCode()+"\n\n");

Console.WriteLine("Введите тираж издания");

try

{

ed1.edit = int.Parse(Console.ReadLine());

}

catch (FormatException nonedition)

{

Console.WriteLine(nonedition.Message);

}

Console.ReadLine();

Console.WriteLine("----------------------------------------------------");

Console.WriteLine("ВЫВОД СПИСКА РЕДАКТОРОВ И СТАТЕЙ ЖУРНАЛА НЕКОТОРЫХ ОБЪЕКТОВ" + "\n\n");

Article[] artic = new Article[2];

Article ed = new Article(new Person("Никитенко", "Андрей", new DateTime(1993, 08, 04)), "Вечность", 3);

Article edd = new Article(new Person("Антонов", "Андрей", new DateTime(1990, 07, 03)), "Жизнь", 4.5);

artic[0] = ed;

artic[1] = edd;

ArrayList al = new ArrayList();

Person etW = new Person("Никитенко","Андрей",new DateTime(1993,08,04));

Person et1W = new Person("Федоренко","Степан", new DateTime(1980, 03, 04));

al.Add(etW);

al.Add(et1W);

Console.WriteLine(ed.ToString());

Console.WriteLine(edd.ToString());

Console.WriteLine("Редактировали");

Console.WriteLine(etW.ToString());

Console.WriteLine(et1W.ToString());

Console.WriteLine("--------------------------------------------------------");

Console.WriteLine("Список статей с рейтингом выше 4");

foreach (Article i in artic)

if (i.Rating > 4)

Console.WriteLine(i.Title.ToString());

Console.WriteLine();

Console.WriteLine("Список статей, в названиях которых есть сочетание 'Жи'");

foreach (Article i in artic)

{

if (i.Title.Contains("Жи"))

Console.WriteLine(i.Title.ToString());

}

Console.ReadLine();

**Результат работы программы:**









