Умова

Скористатись описами абстракцій, визначеними у лабораторній роботі No3. Клас Date не створювати.
Описи типів користувача переписати відповідно синтаксису мови C#.
Врахувати наступне:
- у тих класах, де використовувалось поле типу Date, скористатись типом System.DateTime;
- для реалізації глибокого копіювання скористатись функцією Clone() (реалізація інтерфейсу ICloneable);
- функції-селектори та функції-модифікатори для доступу до закритих полів реалізувати як властивості;
- там, де це доцільно, скористатися специфікаторами readonly, static, protected;
- визначити перевантажені версії віртуального методу ToString() для кожного класу (за винятком класів-колекцій) з формуванням рядка зі значеннями усіх полів;
- для класу, у якому був визначений оператор індексації, визначити індексатор з одним параметром;
- там, де це доцільно, скористатись можливістю генерування виключної ситуації;
- арифметичні оператори перевизначити як статичні методи відповідно правилам C#;
- для реалізації логічних операторів “==” та “!=” потрібно:
= перевизначити метод Equals() так, щоб відбувалась перевірка об’єктів, а не посилань на них;
= перевизначити віртуальний метод int GetHashCode(), для чого скористатись будь-яким рядком, що може ідентифікувати об’єкт у разі його використання в колекціях;
= скористатись методом Equals() в операторних функціях для “==” та “!=” ;
- для реалізації логічних операторів “<” та “>” або “<=” та ”>=” потрібно скористатись функцією CompareTo() (реалізація інтерфейсу IComparable);
- оператори “<<” та ”>>” не перевизначати, виведення повної інформації на консоль реалізувати як віртуальну функцію з використанням методу ToString();
- виведення скороченої інформації (наприклад, значення одного з полів) реалізувати як невіртуальні функції з однаковими іменами у базовому та похідному класах.

Реалізувати тестовий приклад, у якому продемонструвати:
- створення об’єкта та його копії;
- модифікацію одного з об’єктів;
- порівняння об’єктів;
створення третього об’єкта з використанням конструктора умовчання;
- виконання арифметичної операції з присвоюванням третьому об’єкту та з

присвоюванням собі для одного з об’єктів (наприклад, відповідно “+” і “+=”);
- використання індексатора;
- роботу віртуальної та невіртуальної функцій.

Визначено:
- в класі Student визначено конструктор за замовчуванням, конструктор з параметрами, визначено абстрактну функцію object Clone() з інтерфейсу Icloneable для реалізації глибокого копіювання, перевизначено віртуальні функцію bool Equals(object o) та метод int GetHashCode(), перевантажили оператори == та !=, створили функції для виведення інформації про об’єкт даного класу void ShortOut() та віртуальну FullOut(), де використали перевизначений метод string ToString();
- в інших класах виконали аналогічні дії згідно з завданням;
- в тілі програми (static void Main(string[] args)) ініціалізували об’єкти створених нами класів (Student, Customer, Publication, Research), задіяли методи/властивості/etc., визначені в них.

Текст програми

Student.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5
{
   class Student:ICloneable
   {
      private string name, surname;
      private int enroll_year;
      public string Name {get; set;}
      public string Surname {get; set;}
      public int Enroll_year {
         get{return enroll_year;}
         set{if (value>1000 & value<DateTime.Today.Year) enroll_year=value;}
      }
      public Student(){
         Name = "Name";
         Surname = "Surname";
         Enroll_year = DateTime.Today.Year;
      }
      public Student(string nam, string surnam, int year){
         try{
```

```
            Name = String.Copy(nam);
            Surname = String.Copy(surnam);
            Enroll_year=year;
         }
         catch{}
      }
      public object Clone(){
         return new Student(this.Name, this.Surname, this.Enroll_year);
      }
      public override bool Equals(object o)
      {
         if (o != null && o is Student)
         {
            Student temp = o as Student;
            return ((Name.Equals(temp.Name)) &
(Surname.Equals(temp.Surname)) & (Enroll_year == temp.Enroll_year));
         }
         else {
            throw new ArgumentException("Error in Equals.");
         }
      }
      public override int GetHashCode()
      {
         //returns any one HashCode
         return (enroll_year.GetHashCode() | name.GetHashCode() |
surname.GetHashCode() );
      }
      public static bool operator==(Student stud1, Student stud2){
         return stud1.Equals(stud2);
      }
      public static bool operator!=(Student stud1, Student stud2){
         return !stud1.Equals(stud2);
      }
      public void ShortOut() {
         Console.WriteLine("{0} {1}", Surname, Enroll_year);
      }
      public override string ToString(){
         return "\nName : " + Name + "\nSurname : " + Surname + "\nEnroll
year : " + Enroll_year.ToString();
      }
      public void FullOut()
      {
         Console.WriteLine(ToString());
      }
   }
}
```

Customer.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5
{
   class Customer:ICloneable
   {
      protected string name, theme;
      protected int price;
      public Customer(){
         Name = "Name";
         Theme = "Theme";
         Price = 0;
      }
      public Customer(string nam, string them, int pric){
         try{
            Name = String.Copy(nam);
            Theme = String.Copy(them);
            Price = pric;
         }
         catch{}
      }
      public object Clone(){
         return new Customer(Name, Theme, Price);
      }
      public string Name{get; set;}
      public string Theme{get; set;}
      public int Price{
         get{return price;}
         set{if (value>0) price = value;}
      }
      public override string ToString()
      {
         return "\nName : " + name + "\nTheme : " + theme + "\nPrice : " +
price.ToString();
      }
      public virtual void FullOut(){
         Console.WriteLine(ToString());
      }
      public void ShortOut(){
         Console.WriteLine("{0} {1}", Name, Price);
      }
   }
}
```

Publication.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5
{
   class Publication:ICloneable
   {
      private Student author;
      private SciAchivment pub_type;
      public Student Author{
         get{return author;}
         set{if (value is Student) author=value;}
      }
      public SciAchivment Pub_type{get; set;}
      public Publication(){
         Author = new Student();
         Pub_type = (int)SciAchivment.THESIS;
      }
      public Publication(Student stu, SciAchivment ach){
         Author=(Student)stu.Clone();
         pub_type=ach;
      }
      public object Clone(){
         return new Publication(Author, Pub_type);
      }
      public override string ToString()
      {
         return "\nAuthor : " + author.ToString()+ "\nPublication type : " +
pub_type.ToString();
      }
      public virtual void FullOut()
      {
         Console.WriteLine(ToString());
      }
      public void ShortOut()
      {
         Console.WriteLine("{0}", Pub_type);
      }
   }
}
```

Research.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;

namespace Lab5
{
   class Research:Customer, ICloneable, IComparable
   {
      private DateTime sign_date;
      private Publication[] publications;
      private int num_of_publications;
      public DateTime Sign_date{get; set; }
      public Publication[] Publications{
         get{return publications;}
         set{if (value is Publication[]) publications = (Publication[])value;}
      }
      public int Num_of_publications{
         get{return num_of_publications;}
         set{if (value>0) num_of_publications = value;}
      }

      public Research():base(){
         Sign_date=DateTime.Today;
         Num_of_publications = 0;
         this.publications = new Publication[1];
      }
      public Research(Customer cust, Publication[] pubs):base(cust.Name,
cust.Theme, cust.Price){
         Sign_date = DateTime.Today;
         if (pubs != null && pubs.Length != 0)
         {
            Num_of_publications = pubs.Length;
            publications = new Publication[Num_of_publications];
            for (int i = 0; i < Num_of_publications; i++) publications[i] =
(Publication)pubs[i].Clone();
         }
         else
         {
            Num_of_publications = 0;
            this.publications = new Publication[1];
         }
      }
      public new object Clone(){
         Publication[] temp = new Publication[this.Num_of_publications];
         for (int i = 0; i < this.Num_of_publications; i++) temp[i] =
(Publication)this.publications[i].Clone();
         return new Research(new Customer(Name, Theme, Price), temp);
      }

      public void AddPublication(Publication pub){
         int oldsize = Num_of_publications;
         if (oldsize > 0)
         {
```

```
            Publication[] temp = new Publication[oldsize];
            for (int i = 0; i < oldsize; i++) temp[i] = publications[i];
            publications = new Publication[oldsize + 1];
            for (int i = 0; i < oldsize; i++) publications[i] = temp[i];
            publications[oldsize] = pub;
         }
         else
         {
            publications = new Publication[1];
            publications[oldsize] = pub;

         }
         Num_of_publications = publications.Length;
      }

      public int CompareTo(object o)
      {
         if(o != null && o is Research){
            Research temp = (Research)o;
            if (Price < temp.Price) return -1;
            if (Price > temp.Price) return 1;
            return 0;
         }
         else throw new ArgumentException("Error in CompareTo.");
      }
      public static bool operator>(Research lhs, Research rhs){
         return (lhs.CompareTo(rhs)>0 | lhs.CompareTo(rhs)==0);
      }
      public static bool operator<(Research lhs, Research rhs){
         return lhs.CompareTo(rhs)<0;
      }

      public string this[int index]{
         get{ return publications[index].ToString();}
         set{Console.WriteLine("Impossible to write string to Publication.");}
      }

      public override string ToString()
      {
         string temp = "\nName : " + Name + "\nPrice : " + Price.ToString() +
"\nResearch date : " + Sign_date.ToString()
             + "\nTheme of research : " + Theme + "\nNumber of publications : "
+ Num_of_publications.ToString();
         for (int i = 0; i < Num_of_publications; i++) temp +=
publications[i].ToString();
         return temp;
      }
      public override void FullOut()
      {
         Console.WriteLine(ToString());
      }
```

```
        public new void ShortOut()
        {
            Console.WriteLine("Number of publications : {0}",
Num_of_publications);
        }

    }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Lab5
{
    public enum SciAchivment { THESIS, ARTICLE, REPORT, INTARTICLE };

    class Program
    {
        static void Main(string[] args)
        {

            Student stud1 = new Student("Devi", "Jones", 1678);
            Student stud2 = stud1.Clone() as Student;
            Console.WriteLine("Student 1: {0}\n", stud1);
            Console.WriteLine("Student 2: {0}\n", stud2);
            Console.WriteLine("Student 1==Student 2: {0}\n", stud1==stud2);
            stud2.Enroll_year = 1976;
            Console.WriteLine("Student 1==Student 2: {0}\n", stud1 == stud2);
            Console.WriteLine("\n------------------");

            Customer cust1 = new Customer("Yanina", "Religion", 115);
            Publication pub1 = new Publication(stud1, SciAchivment.ARTICLE);
            Publication[] pubarray = new Publication[3];
            pubarray[0]=new Publication();
            pubarray[1]=new Publication();
            pubarray[2] = (Publication)pub1.Clone();

            Research ob1 = new Research(cust1, null);
            Research ob2 = ob1.Clone() as Research;
            Console.WriteLine("Research 1: {0}\n", ob1);
            Console.WriteLine("\n------------------");
            ob2.AddPublication(pub1);
            Console.WriteLine("Research 2: {0}\n", ob2);
            Customer ob3 = new Research(cust1, pubarray);
            Console.WriteLine("Research 3: {0}\n", ob3);
            Console.WriteLine("\n------------------");
```

```
            Console.WriteLine("Research 3 < Research 2: {0}\n", ob2<ob1);
            Console.WriteLine("\n------------------");

            Console.WriteLine(ob2[0]);
            Console.WriteLine("\n------------------");

            cust1.FullOut();
            Console.WriteLine("\n------------------");
            cust1.ShortOut();
            Console.WriteLine("\n------------------");
            ob3.FullOut();
            Console.WriteLine("\n------------------");
            ob3.ShortOut();
            Console.WriteLine("\n------------------");

            Console.ReadKey();
        }
    }
}
```

Результати роботи програми

```
Student 1:
Name : Devi
Surname : Jones
Enroll year : 1678

Student 2:
Name : Devi
Surname : Jones
Enroll year : 1678

Student 1==Student 2: True

Info in Student 2 is changed.
Student 1==Student 2: False


------------------
Research 1:
Name : Yanina
Price : 115
Research date : 12/19/2016 12:00:00 AM
Theme of research : Religion
Number of publications : 0


------------------
Research 2:
Name : Yanina
Price : 115
Research date : 12/19/2016 12:00:00 AM
```

Theme of research : Religion
Number of publications : 1
Author :
Name : Devi
Surname : Jones
Enroll year : 1678
Publication type : THESIS

Research 3:
Name : Yanina
Price : 115
Research date : 12/19/2016 12:00:00 AM
Theme of research : Religion
Number of publications : 3
Author :
Name : Name
Surname : Surname
Enroll year : 0
Publication type : THESIS
Author :
Name : Name
Surname : Surname
Enroll year : 0
Publication type : THESIS
Author :
Name : Devi
Surname : Jones
Enroll year : 1678
Publication type : THESIS

Research 3 < Research 2: False

Author :
Name : Devi
Surname : Jones
Enroll year : 1678
Publication type : THESIS

Name : Yanina
Theme : Religion
Price : 115

Yanina 115

Name : Yanina
Price : 115

Research date : 12/19/2016 12:00:00 AM
Theme of research : Religion
Number of publications : 3
Author :
Name : Name
Surname : Surname
Enroll year : 0
Publication type : THESIS
Author :
Name : Name
Surname : Surname
Enroll year : 0
Publication type : THESIS
Author :
Name : Devi
Surname : Jones
Enroll year : 1678
Publication type : THESIS

Yanina 115
