Лабораторная работа по теме

«Элементы управления Windows Forms, рефлексия, xml-сериализация»

1. Исследуйте элементы управления (ЭУ) Windows Forms. Выявите основные особенности и отличительные черты каждого ЭУ.
2. Проанализируйте исходный код примера сериализации и десериализации объектов в формате xml.
3. Изучите основные особенности использования рефлексии. Проанализируйте информацию из следующего источника.
4. Изучите основные возможности использования рефлексии для валидации данных (используйте проект из директории "***/sample/Validator***"). «Проиграйтесь» с ним.
5. Создайте приложение по варианту:

a) разрабатываемое приложение должно состоять из нескольких сборок;
b) основная сборка – Windows Forms приложение, состоящее из нескольких форм. Главная форма должна содержать список всех объектов основного класса (из xml файла). При нажатии на выбранный объект открывается форма с подробным описанием всех полей объекта с возможностью редактировать, а также удалять объект. Необходимо также реализовать логику для создания, редактирования, удаления агрегируемого объекта;
c) используйте различные ЭУ (радиокнопки, списки, поля ввода, метки, кнопки, слайдеры и т.д.) для ввода/вывода информации об объектах, указанных в вариантах.
d) разработайте собственные атрибуты для валидации моделей. Реализуйте необходимую логику с использованием рефлексии для валидации объектов различных типов;
e) добавьте функционал сериализации и десериализации данных в формате xml;
f) при сохранении любой из моделей сериализованный объект должен дописывается к уже существующим объектам в xml файл;
g) функционал по созданию новых записей следует вынести в виде отдельных пунктов в меню формы.
h) при работе с наследуемыми классами используйте паттерн «Абстрактная фабрика» либо «Фабричный метод».

Вариант	**Задание**
1, 8	*Основной класс* – «Рабочий». *Наследуемые классы*: рабочий индустриального предприятия (завод, фабрика), рабочий транспортного предприятия (железная дорога, аэродромная служба, др.). Возможные поля: ФИО, возраст, специальность, стаж, зарплата, пол, место работы и др.

	Агрегируемый объект – «Место работы». Возможные поля: год принятия на работу, год увольнения, причина увольнение (по истечению контракта, по статье…), компания, должность и т.д.
2, 9	*Основной класс* – «Счет». *Наследуемые классы*: расчетный счет, накопительный счет. Возможные поля: номер, тип вклада, PIN, баланс, дата создания счета, пользователь, история изменений счета (список операций пополнения/снятия денег со счета), и т.д. *Агрегируемый объект* – «Пользователь». Возможные поля: ФИО, дата рождения, тип пользователя (активный, заблокированный …) и т.д.
3, 10	*Основной класс* – «Дисциплина». Наследуемые классы: экономическая дисциплина, химическая дисциплина. Возможные поля: название, количество лекций, количество лабораторных, наличие курсового проекта, вид контроля, количество слушателей, лектор и т.д. *Агрегируемый объект*: «Лектор». Возможные поля: факультет, кафедра, наличие ученой степени, ФИО и т.д.
4, 11	*Основной класс* – «Книга». *Наследуемые классы*: аудиокнига, учебник. Возможные поля: название, область науки, количество страниц, издательство, тип переплета, наличие CD, DVD, автор и т.д. *Агрегируемый объект* – «Автор». Возможные поля: ФИО, страна, город, пол и т.д.
5, 12	*Основной класс* – «Растение». *Наследуемые классы*: папоротник, ель обыкновенная. Возможные поля: название, описание, вид, возраст, класс, размер соцветия, класс опасности, ареал произрастания, область применения (в пищевой промышленности, в получении лекарственных препаратов, в сельском хозяйстве…), первооткрыватель (ученый) и т.д. *Агрегируемый объект* – «Место произрастание». Возможные поля: страна, область (регион), район, площадь ($км^2$) и т.д.
6, 13, 15	*Основной объект* – «Компьютер». *Наследуемые классы*: ультрабук, суперкомпьютер. Возможные поля: количество ядер процессора, тип процессора, частота, наличие технологии Hyper-Threading, разрядность архитектуры, производитель, видеокарта, размер и тип ОЗУ, размер и тип жесткого диска, и т.д. *Агрегируемый объект* – «Производитель». Возможные поля: имя, страна, год основания, типы производимых компьютеров и т.д.
7, 14, 16	Основной объект – «Самолет». *Наследуемые классы*: военный самолет, гражданский самолет. Обязательные поля: номер, класс, авиакомпания, члены экипажа

	(список), количество мест, рейсы, год выпуска, производитель и т.п. *Агрегируемый объект* – «Производитель». Возможные поля: имя, страна, год основания, типы производимых самолетов и т.д.

Исходный код примера (сериализация и десериализация):

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Xml.Serialization;

namespace Serialization
{
    class Program
    {
        static void Main()
        {
            var role = new List<Role>
            {
                new Role
                {
                    Id = Guid.NewGuid(),
                    Name = "User"
                }
            };

            var users = new List<User>
            {
                new User("Renee", "Miller", 24)
                {
                    Roles = role,
                    Type = UserType.New,
                    Sex = 'W'
                },
                new User("Angel", "Bates", 56)
                {
                    Roles = role,
                    Type = UserType.Locked,
                    Sex = 'M'
                },
            };

            XmlSerializeWrapper.Serialize(users, "users.xml");
            var deserializeUsers =
XmlSerializeWrapper.Deserialize<List<User>>("users.xml");

            XmlSerializeWrapper.Serialize(users.First(), "user.xml");
            var deserializeUser = XmlSerializeWrapper.Deserialize<User>("user.xml");
        }
    }

    [Serializable]
    [XmlRoot(Namespace = "example")]
    [XmlType("user")]
    public class User
    {
        #region Properties

        [XmlElement(ElementName = "id")]
```

```
        public Guid Id { get; set; }

        [XmlElement(ElementName = "name")]
        public string FirstName { get; set; }

        [XmlElement(ElementName = "surname")]
        public string LastName { get; set; }

        [XmlElement(ElementName = "age")]
        public int Age { get; set; }

        [XmlIgnore]
        public char Sex { get; set; }

        [XmlElement(ElementName = "type")]
        public UserType Type { get; set; }

        [XmlArray("roles")]
        [XmlArrayItem("role")]
        public List<Role> Roles { get; set; }

        #endregion

        #region Constructors

        public User()
        {
            Id = Guid.NewGuid();
        }

        public User(string firstName, string lastName, int age) : this()
        {
            FirstName = firstName;
            LastName = lastName;
            Age = age;
        }

        #endregion
    }

    [Serializable]
    public class Role
    {
        public Guid Id { get; set; }

        public string Name { get; set; }
    }

    [Serializable]
    public enum UserType
    {
        [XmlEnum("L")]
        Locked,

        [XmlEnum("N")]
        New
    }

    public static class XmlSerializeWrapper
    {
        public static void Serialize<T>(T obj, string filename)
        {
            XmlSerializer formatter = new XmlSerializer(typeof(T));

            using (FileStream fs = new FileStream(filename, FileMode.OpenOrCreate))
```

```
            {
                formatter.Serialize(fs, obj);
            }
        }

        public static T Deserialize<T>(string filename)
        {
            T obj;
            using (FileStream fs = new FileStream(filename, FileMode.OpenOrCreate))
            {
                XmlSerializer formatter = new XmlSerializer(typeof(T));
                obj = (T)formatter.Deserialize(fs);
            }

            return obj;
        }
    }
}
```