

Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
Федерального государственного бюджетного образовательного учреждения
высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет _____ ИТР _____

Кафедра _____ ПИИ _____

ЛАБОРАТОРНАЯ РАБОТА №2

По Объектно-ориентированному программированию

Руководитель

Привезенцев Д.Г.

(фамилия, инициалы)

(подпись)

(дата)

Студент ПИИ - 121

(группа)

Ермилов М.В.

(фамилия, инициалы)

(подпись)

(дата)

Муром 2022

Лабораторная работа №2

Тема: Проектирование классов. Включение классов. Инкапсуляция

Ход работы:

Задание: Карточка персоны содержит фамилию и дату рождения.

Реализовать класс ListPerson для работы с картотекой персоналий. Класс должен содержать массив карточек персон. Реализовать методы добавления и удаления карточек персон, а также метод доступа к карточке по фамилии. Фамилии в массиве должны быть уникальны.

Примечание: Задание было изменено преподавателем, до его изменения мной уже были сделаны функции работы со знаками зодиака, которые я так-же приложу в данную работу.

Код:

Класс для хранения знака зодиака:

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

namespace lab2
{
    /// <summary>
    /// Класс описывающий знак зодиака и проверки даты на входимость ее в этот знак
    зодиака
    /// </summary>
    class Zodiac
    {
        #region Публичные свойства
        /// <summary>
        /// Название знака зодиака
        /// </summary>
        public string Name { get => name; }
        /// <summary>
        /// Дата начала знака зодиака
        /// </summary>
        public string Begin { get => _dateTime(begin); }
        /// <summary>
        /// Дата окончания знака зодиака

```

					МИ ВлГУ 09.03.04		
Изм.	Лист	№ докум.	Подпись	Дата	Проектирование классов. Включение классов. Инкапсуляция		
Разраб.	Ермилов М.В.						
Провер.	Привезенцев Д.Г.						
Реценз.							
Н. Контр.							
Утверд.	.				ПИН-121		
					Лит.	Лист	Листов
						2	12

```

    /// </summary>
    public string End { get => _dateTime(end); }
#endregion
#region Приватные переменные
    /// <summary>
    /// Статическая переменная необходимая для работы с текстом
    /// </summary>
    private static CultureInfo cultureInfo = new CultureInfo("ru-RU");
    /// <summary>
    /// Переменная хранящая в себе название знака зодиака
    /// </summary>
    private string name;
    /// <summary>
    /// Переменная хранящая в себе дату начала знака зодиака
    /// </summary>
    private DateTime begin;
    /// <summary>
    /// Переменная хранящая в себе дату окончания знака зодиака
    /// </summary>
    private DateTime end;
#endregion
#region Приватные функции
    /// <summary>
    /// Функция, для приведения даты в стандартный вид удобный для класса
    /// </summary>
    /// <remarks>
    /// Используется 4 год, из-за того что могут попасться високосные числа
    /// </remarks>
    /// <param name="month">Месяц</param>
    /// <param name="day">День</param>
    /// <returns>Дата</returns>
    private DateTime _dateTime(int month, int day) => new DateTime(4, month, day);
    /// <summary>
    /// Функция для преобразования даты в текст
    /// </summary>
    /// <param name="date">Дата которая будет преобразована</param>
    /// <returns>Текстовое представление даты</returns>
    private string _dateTime(DateTime date) => $"{date.Day} {date.ToString("MMMM",
cultureInfo)}";
    /// <summary>
    /// Функция для преобразование названия в нормальный вид
    /// </summary>
    /// <remarks>
    /// Убирает лишнии пробелы и записывает текст с заглавной буквы
    /// </remarks>
    /// <param name="name">Название</param>
    /// <returns>Преобразованное название</returns>
    private string _name(string name) =>
cultureInfo.TextInfo.ToTitleCase(name.Trim());
#endregion
#region Функция проверки входит ли дата в знак зодиака (и его перегрузки)
    /// <summary>
    /// Функция проверки входит ли дата в знак зодиака
    /// </summary>
    /// <param name="Month">проверяемый месяц</param>
    /// <param name="Day">Проверяемый день</param>
    /// <returns>Результат проверки</returns>
    public bool Check(int Month, int Day)
    {
        DateTime DateParams = _dateTime(Month, Day);
        DateTime E = end;
        if (end.Ticks < begin.Ticks)
        {
            E = new DateTime(5, end.Month, end.Day);
            if(DateParams.Ticks < begin.Ticks)

```

					МИ ВлГУ 09.03.04	Лист
						3
Изм.	Лист	№ докум.	Подпись	Дата		

```

        DateParams = new DateTime(5, Month, Day);
    }

    if (begin.Ticks <= DateParams.Ticks && DateParams.Ticks <= E.Ticks)

        return true;
    return false;
}
/// <summary>
/// Функция проверки входит ли дата в знак зодиака
/// </summary>
/// <param name="Date">Проверяемая дата</param>
/// <returns>Результат проверки</returns>
public bool Check(DateTime Date) => Check(Date.Month, Date.Day);
/// <summary>
/// Функция проверки входит ли дата в знак зодиака
/// </summary>
/// <param name="Date">Проверяемая дата в тиках</param>
/// <returns>Результат проверки</returns>
public bool Check(long Ticks) => Check(new DateTime(Ticks));
#endregion
#region Конструктор (и его перегрузки)
/// <summary>
/// Конструктор
/// </summary>
/// <param name="Name">Название знака зодиака</param>
/// <param name="BeginMonth">Месяц начала знака зодиака</param>
/// <param name="BeginDay">День в месяце начала знака зодиака</param>
/// <param name="EndMonth">Месяц окончания знака зодиака</param>
/// <param name="EndDay">День в месяце окончания знака зодиака</param>
public Zodiac(string Name, int BeginMonth, int BeginDay, int EndMonth, int
EndDay)
{
    begin = _dateTime(BeginMonth, BeginDay);
    end = _dateTime(EndMonth, EndDay);
    name = _name(Name);
}
/// <summary>
/// Конструктор
/// </summary>
/// <param name="Name">Название знака зодиака</param>
/// <param name="Begin">Начало знака зодиака</param>
/// <param name="End">Окончание знака зодиака</param>
public Zodiac(string Name, DateTime Begin, DateTime End) : this(Name,
Begin.Month, Begin.Day, End.Month, End.Day) { }
/// <summary>
/// Конструктор
/// </summary>
/// <param name="Name">Название знака зодиака</param>
/// <param name="BeginTicks">Начало знака зодиака в тиках</param>
/// <param name="EndTicks">Окончание знака зодиака в тиках</param>
public Zodiac(string Name, long BeginTicks, long EndTicks) : this(Name, new
DateTime(BeginTicks), new DateTime(EndTicks)) { }
#endregion
#region Переопределенные методы и функции
public override string ToString() => $"Знак зодиака: {Name};\nНачало:
{Begin};\nКонец: {End};";
#endregion
}
}

```

					МИ ВлГУ 09.03.04	Лист
Изм.	Лист	№ докум.	Подпись	Дата		4

Enum список знаков зодиака:

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

namespace lab2
{
    /// <summary>
    /// Список знаков зодиака
    /// </summary>
    enum ZodiacEnum
    {
        /// <summary>
        /// Овен
        /// </summary>
        Aries = 0,
        /// <summary>
        /// Телец
        /// </summary>
        Taurus = 1,
        /// <summary>
        /// Близнецы
        /// </summary>
        Gemini = 2,
        /// <summary>
        /// Рак
        /// </summary>
        Cancer = 3,
        /// <summary>
        /// Лев
        /// </summary>
        Leo = 4,
        /// <summary>
        /// Дева
        /// </summary>
        Virgo = 5,
        /// <summary>
        /// Весы
        /// </summary>
        Libra = 6,
        /// <summary>
        /// Скорпион
        /// </summary>
        Scorpio = 7,
        /// <summary>
        /// Стрелец
        /// </summary>
        Sagittarius = 8,
        /// <summary>
        /// Козерог
        /// </summary>
        Capricorn = 9,
        /// <summary>
        /// Водолей
        /// </summary>
        Aquarius = 10,
        /// <summary>
        /// Рыбы
        /// </summary>
        Pisces = 11,
    }
}
```

					МИ ВлГУ 09.03.04	Лист
						5
Изм.	Лист	№ докум.	Подпись	Дата		

Статичный класс, в котором содержатся все знаки зодиака и методы работы с ними:

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

namespace lab2
{
    /// <summary>
    /// Статический класс хранящий в себе коллекцию знаков зодиака и поиска их
    /// </summary>
    static class ZodiacList
    {
        #region Приватные переменные
        /// <summary>
        /// Статическая коллекция всех знаков зодиака
        /// </summary>
        private static List<Zodiac> list = new List<Zodiac>()
        {
            new Zodiac("овен", 3, 21, 4, 20),
            new Zodiac("телец", 4, 21, 5, 21),
            new Zodiac("близнецы", 5, 22, 6, 21),
            new Zodiac("рак", 6, 22, 7, 22),
            new Zodiac("лев", 7, 23, 8, 21),
            new Zodiac("дева", 8, 22, 9, 23),
            new Zodiac("весы", 9, 24, 10, 23),
            new Zodiac("скоприон", 10, 24, 11, 22),
            new Zodiac("стрелец", 11, 23, 12, 22),
            new Zodiac("козерог", 12, 23, 1, 20),
            new Zodiac("водолей", 1, 21, 2, 19),
            new Zodiac("рыбы", 2, 20, 3, 20)
        };
        #endregion
        #region Публичные свойства
        /// <summary>
        /// Кол-во знаков зодиака
        /// </summary>
        public static int Length { get => list.Count; }
        /// <summary>
        /// Кол-во знаков зодиака
        /// </summary>
        public static int Count { get => Length; }
        #endregion
        #region Функция получения знака зодиака по его номеру (и его перегрузки)
        /// <summary>
        /// Знак зодиака
        /// </summary>
        /// <param name="i">Номер знака зодиака от 0 до Length (11)</param>
        /// <returns>Информация о знаке зодиака</returns>
        public static Zodiac Get(int i)
        {
            if (i < 0) i = 0;
            if (i > Length) i = Length;
            return list[i];
        }
        /// <summary>
        /// Знак зодиака
        /// </summary>
        /// <param name="i">Знак зодиака</param>
        /// <returns>Информация о знаке зодиака</returns>
        public static Zodiac Get(ZodiacEnum i) { return Get((int)i); }
        #endregion
    }
}
```

```

#region Функция поиска знака зодиака (и его перегрузки)
/// <summary>
/// Функция поиска знака зодиака
/// </summary>
/// <param name="Month">Месяц</param>
/// <param name="Day">День</param>
/// <returns>Знак зодиака</returns>
public static Zodiac Search(int Month, int Day)
{
    foreach(Zodiac zodiac in list)
        if(zodiac.Check(Month, Day))
            return zodiac;
    return null;
}
/// <summary>
/// Функция поиска знака зодиака
/// </summary>
/// <param name="Date">Дата</param>
/// <returns></returns>
public static Zodiac Search(DateTime Date) => Search(Date.Month, Date.Day);
/// <summary>
/// Функция поиска знака зодиака
/// </summary>
/// <param name="Ticks">Тики</param>
/// <returns>Знак зодиака</returns>
public static Zodiac Search(long Ticks) => Search(new DateTime(Ticks));
#endregion
}
}

```

Класс содержащий в себе информацию о персоне:

```

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

namespace lab2
{
    class Person
    {
        #region Приватные переменные
        /// <summary>
        /// Статичная переменная необходимая для работы с текстом
        /// </summary>
        private static CultureInfo cultureInfo = new CultureInfo("ru-RU");
        /// <summary>
        /// Переменная хранящая в себе фамилию
        /// </summary>
        private string lastName;
        /// <summary>
        /// Переменная хранящая в себе дату рождения
        /// </summary>
        private DateTime dateBirth;
        #endregion
        #region Публичные свойства
        /// <summary>
        /// Фамилия
        /// </summary>
        public string LastName { get => lastName; }
        /// <summary>
        /// Дата рождения
        /// </summary>
        public DateTime DateBirth { get => dateBirth; }
        /// <summary>

```

					МИ ВлГУ 09.03.04	Лист
Изм.	Лист	№ докум.	Подпись	Дата		7

```

    /// Сегодня день рождения?
    /// </summary>
    public bool IsBirth { get => IsBirthDate(DateTime.Now); }
    /// <summary>
    /// Возраст
    /// </summary>
    public int Age { get => AgeDate(DateTime.Now); }
#endregion
#region Приватные функции
    /// <summary>
    /// Функция преобразования фамилии
    /// </summary>
    /// <remarks>
    /// Убирает лишнии пробелы и записывает текст с заглавной буквы
    /// </remarks>
    /// <param name="lastName">Фамилия</param>
    /// <returns>преобразованная фамилия</returns>
    private string _lastName(string lastName) =>
cultureInfo.TextInfo.ToTitleCase(lastName.Trim());
#endregion
    /// <summary>
    /// Функция сравнения фамилий
    /// </summary>
    /// <param name="LastName">Фамилия</param>
    /// <returns>true если фамилии одинаковые</returns>
    public bool CheckLastName(string LastName) =>
this.LastName.Equals(_lastName(LastName));
#region Функция определения является ли дата – днём рождения (и его перегрузки)
    /// <summary>
    /// Функция определения является ли дата – днём рождения
    /// </summary>
    /// <param name="Ticks">Проверяемая дата в тиках</param>
    /// <returns>true если дата является днем рождения</returns>
    public bool IsBirthDate(long Ticks) => IsBirthDate(new DateTime(Ticks));
    /// <summary>
    /// Функция определения является ли дата – днём рождения
    /// </summary>
    /// <param name="Date">Проверяемая дата</param>
    /// <returns>true если дата является днем рождения</returns>
    public bool IsBirthDate(DateTime Date) => IsBirthDate(Date.Month, Date.Day);
    /// <summary>
    /// Функция определения является ли дата – днём рождения
    /// </summary>
    /// <param name="Month">Месяц проверяемой даты</param>
    /// <param name="Day">День проверяемой даты</param>
    /// <returns>true если дата является днем рождения</returns>
    public bool IsBirthDate(int Month, int Day) => Day == DateBirth.Day && Month ==
DateBirth.Month;
#endregion
#region Функция определения возраста по дате (и его перегрузки)
    /// <summary>
    /// Функция определения возраста по дате
    /// </summary>
    /// <param name="Date">Дата по которой смотрим возраст</param>
    /// <returns>Возраст</returns>
    public int AgeDate(DateTime Date)
    {
        if (Date.Ticks > DateBirth.Ticks)
            return new DateTime(Date.Ticks - DateBirth.Ticks).Year;
        return -1;
    }
    /// <summary>
    /// Функция определения возраста по дате
    /// </summary>
    /// <param name="Year">Год даты по которой смотрим возраст</param>
    /// <param name="Month">Месяц даты по которой смотрим возраст</param>

```

					МИ ВлГУ 09.03.04	Лист
						8
Изм.	Лист	№ докум.	Подпись	Дата		


```

    /// <param name="Day">День даты по которой смотрим возраст</param>
    /// <returns>Возраст</returns>
    public int AgeDate(int Year, int Month, int Day) => AgeDate(new DateTime(Year,
Month, Day));
    /// <summary>
    /// Функция определения возраста по дате
    /// </summary>
    /// <param name="Ticks">Дата по которой смотрим возраст в тиках</param>
    /// <returns>Возраст</returns>
    public int AgeDate(long Ticks) => AgeDate(new DateTime(Ticks));
#endregion
#region Конструктор (и его перегрузки)
    /// <summary>
    /// Конструктор
    /// </summary>
    /// <param name="Lastname">Фамилия</param>
    /// <param name="DateBirth">Дата рождения</param>
    public Person(string Lastname, DateTime DateBirth)
    {
        lastName = _lastName(Lastname);
        dateBirth = DateBirth;
    }
    /// <summary>
    /// Конструктор
    /// </summary>
    /// <param name="Lastname">Фамилия</param>
    /// <param name="YearBirth">Год рождения</param>
    /// <param name="MonthBirth">Месяц рождения</param>
    /// <param name="DayBirth">День рождения</param>
    public Person(string Lastname, int YearBirth, int MonthBirth, int DayBirth)
        : this(Lastname, new DateTime(YearBirth, MonthBirth, DayBirth)) { }
    /// <summary>
    /// Конструктор
    /// </summary>
    /// <param name="Lastname">Фамилия</param>
    /// <param name="Tick">Дата рождения в тиках</param>
    public Person(string Lastname, long Tick)
        : this(Lastname, new DateTime(Tick)) { }
#endregion

#region Переопределенные методы и функции
    public override string ToString()
    {
        return $"{LastName} {dateBirth.Day} {dateBirth.ToString("MMMM",
cultureInfo)} {dateBirth.Year}";
    }
#endregion
}
}

```

Класс содержащий в себе коллекции карточечек персон и работ с ними:

```

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

namespace lab2
{
    class PersonList
    {
        #region Приватные переменные
        /// <summary>
        /// Лист хранящий в себе все карточки
        /// </summary>

```

					МИ ВлГУ 09.03.04	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

```

private List<Person> list;
#endregion
#region Приватные функции
/// <summary>
/// Поиск карточки по фамилии
/// </summary>
/// <param name="lastname">Фамилия</param>
/// <returns>Индекс карточки</returns>
private int index(string lastname)
{
    for(int i = 0; i < list.Count; i++)
        if(list[i].CheckLastName(lastname))
            return i;
    return -1;
}
#endregion
#region Функция поиска карточки (и его перегрузки)
/// <summary>
/// Поиск карточки
/// </summary>
/// <param name="Lastname">Фамилия</param>
/// <returns>Карточка</returns>
public Person Search(string Lastname)
{
    int i = index(Lastname);
    if(i >= 0)
        return list[i];
    return null;
}
/// <summary>
/// Поиск карточки
/// </summary>
/// <param name="person">Карточка</param>
/// <returns>Карточка</returns>
public Person Search(Person person) => Search(person.LastName);
#endregion
#region Функция удаления карточки (и его перегрузки)
/// <summary>
/// Удаление карточки
/// </summary>
/// <param name="Lastname">Фамилия</param>
/// <returns>результат операции</returns>
public bool Remove(string Lastname)
{
    int i = index(Lastname);
    if (i >= 0)
    {
        list.RemoveAt(i);
        return true;
    }
    return false;
}
/// <summary>
/// Удаление карточки
/// </summary>
/// <param name="person">Карточка</param>
/// <returns>Результат операции</returns>
public bool Remove(Person person) => Remove(person.LastName);
#endregion
#region Функция добавление карточки (и его перегрузки)
/// <summary>
/// Добавление карточки
/// </summary>
/// <param name="person">Карточка</param>
/// <returns>Результат операции</returns>
public bool Add(Person person)

```

```

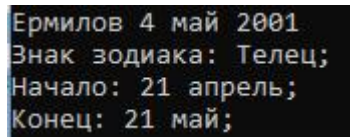
{
    int i = index(person.LastName);
    if (i == -1)
    {
        list.Add(person);
        return true;
    }
    return false;
}
/// <summary>
/// Добавление карточки
/// </summary>
/// <param name="Lastname">Фамилия</param>
/// <param name="DateBirth">Дата рождения</param>
/// <returns>Результат операции</returns>
public bool Add(string Lastname, DateTime DateBirth) => Add(new Person(Lastname,
DateBirth));
/// <summary>
/// Добавление карточки
/// </summary>
/// <param name="Lastname">Фамилия</param>
/// <param name="YearBirth">Год рождения</param>
/// <param name="MonthBirth">Месяц рождения</param>
/// <param name="DayBirth">День рождения</param>
/// <returns>Результат операции</returns>
public bool Add(string Lastname, int YearBirth, int MonthBirth, int DayBirth)
=> Add(new Person(Lastname, YearBirth, MonthBirth, DayBirth));
/// <summary>
/// Добавление карточки
/// </summary>
/// <param name="Lastname">Фамилия</param>
/// <param name="Tick">Дата рождения в тиках</param>
/// <returns>Результат операции</returns>
public bool Add(string Lastname, long Tick) => Add(new Person(Lastname, Tick));
#endregion
#region Конструктор и его перегрузки
/// <summary>
/// Конструктор
/// </summary>
public PersonList() { }
/// <summary>
/// Конструктор
/// </summary>
/// <param name="personList">Другой PersonList</param>
public PersonList(PersonList personList) : this(personList.list) { }
/// <summary>
/// Конструктор
/// </summary>
/// <param name="personList">Коллекция карточек</param>
public PersonList(List<Person> personList) => list = personList;
/// <summary>
/// Конструктор
/// </summary>
/// <param name="person">Массив карточек</param>
public PersonList(Person[] person) : this(person.ToList()) { }
#endregion
}
}

```

Основная программа:

```
using lab2;

class Program
{
    static void Main(string[] arg)
    {
        PersonList p = new PersonList(new List<Person>(){
            new Person("Ермилов", 2001, 5, 4),
            new Person("Пушкин", 1799, 5, 26),
            new Person("Толстой", 1828, 9, 9)
        });
        Person p1 = p.Search(new Person("Ермилов", 2001, 5, 4));
        Console.WriteLine(p1);
        Console.WriteLine(ZodiacList.Search(p1.DateBirth));
    }
}
```



Ермилов 4 май 2001
Знак зодиака: Телец;
Начало: 21 апрель;
Конец: 21 май;

Рис. 1 - Пример работы программы.