Министерство образования Республики Беларусь

Учреждение образования

Брестский государственный технический университет

Кафедра ИИТ

Лабораторная работа №4

За 5 семестр

По дисциплине «Современные платформы программирования»

Выполнил: студент 3 курса

Группы ПО-4(2)

Коташевич С.Н.

Проверил: Монтик Н. С.

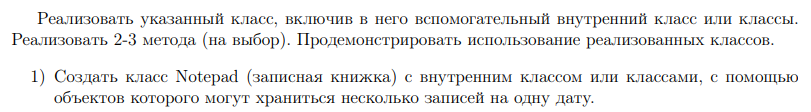
Брест 2021

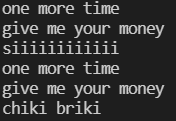
Лабораторная работа №4

Вариант 1

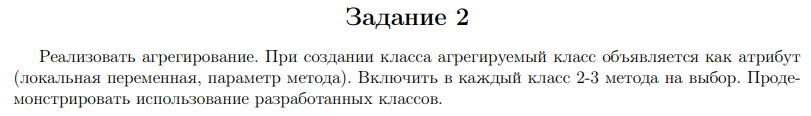
Цель: приобрести практические навыки в области объектно-ориентированного проектирования

Задание 1





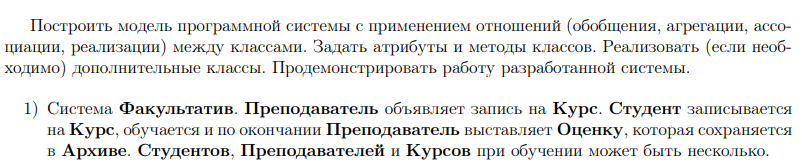
Задание 2







Задание 3





Текст программы:

Program.cs

using System;

namespace lab04

{

    class Program

    {

        static void Task1() {

            Notepad notepad = new Notepad();

            notepad.AddRecord(new DateTime(1999, 11, 12), "one more time");

            notepad.AddRecord(new DateTime(1999, 11, 12), "give me your money");

            notepad.ShowRecords(new DateTime(1999, 11, 12));

            notepad.AddRecord(new DateTime(2000, 11, 12), "siiiiiiiiiiii");

            notepad.ShowRecords(new DateTime(2000, 11, 12));

            notepad.AddRecord(new DateTime(1998, 11, 12), "kill me please");

            notepad.AddRecord(new DateTime(1999, 11, 12), "chiki briki");

            notepad.ShowRecords(new DateTime(1999, 11, 12));

        }

        static void Task2() {

            Strings strings = new Strings();

            Word word = new Word("ajsdbhasbdsabsaduabs");

            Word word1 = new Word("jdngkdfmgkmdgkmdfg");

            Character character = new Character(' ');

            strings.AddWord(word);

            strings.Show();

            strings.AddCharacter(character);

            strings.AddWord(word1);

            strings.Show();

        }

        static void Task3() {

            Teacher teacher1 = new Teacher("Name1");

            Teacher teacher2 = new Teacher("Name2");

            teacher1.AddCourse("PHP");

            teacher1.AddCourse("Scala");

            teacher2.AddCourse("Scala");

            Student student1 = new Student("Danya");

            Student student2 = new Student("Sanya");

            student1.EnterToCourse(teacher1, "PHP");

            student2.EnterToCourse(teacher1, "PHP");

            student2.EnterToCourse(teacher2, "Scala");

            teacher1.SetMark(9, student1, "PHP");

            teacher1.SetMark(7, student2, "PHP");

            teacher2.SetMark(10, student2, "Scala");

            Archive.ShowMarks();

        }

        static void Main(string[] args)

        {

            //Task1();

            //Task2();

            Task3();

        }

    }

}

Classes.cs

using System;

namespace lab04 {

    class Strings {

        private string str;

        public void AddWord(Word word) {

            str += word.ToString();

        }

        public void AddCharacter(Character character) {

            str += character.ToString();

        }

        public void Show() {

            Console.WriteLine(str);

        }

    }

    class Word {

        public string word { get; set; }

        public Word(string wrd) {

            word = wrd;

        }

        public override string ToString() {

            return word.ToString();

        }

    }

    class Character {

        public char character { get; set; }

        public Character(char ch) {

            character = ch;

        }

        public override string ToString()

        {

            return character.ToString();

        }

    }

}

Course.cs

using System;

using System.Collections.Generic;

namespace lab04 {

    abstract class Person {

        protected string \_name;

        public virtual void ShowName() {

            Console.WriteLine($"Name: {\_name}");

        }

        public override string ToString()

        {

            return \_name;

        }

    }

    class Teacher : Person {

        public Dictionary<Course, List<Student>> \_lst = new Dictionary<Course, List<Student>>();

        public Teacher(string name) {

            \_name = name;

        }

        public void AddCourse(string nameofcourse) {

            Course course = new Course(nameofcourse);

            \_lst.Add(course, new List<Student>());

        }

        public override string ToString()

        {

            return \_name.ToString();

        }

        public Course GetCourse(string name) {

            foreach (var sd in \_lst.Keys) {

                if (sd.ToString() == name){

                    return sd;

                }

            }

            return null;

        }

        public Mark SetMark(int mark, Student student, string course) {

            var n = GetCourse(course);

            var v = \_lst[n];

            v.Remove(student);

            \_lst[n] = v;

            Mark mar = new Mark(mark, this, student, n);

            Archive.AddMark(mar);

            return mar;

        }

    }

    class Course {

        private string \_nameofcourse;

        public Course (string coursename) {

            \_nameofcourse = coursename;

        }

        public override string ToString()

        {

            return \_nameofcourse;

        }

    }

    class Student : Person {

        private List<Course> \_courses = new List<Course>();

        public Student(string name) {

            \_name = name;

        }

        public void EnterToCourse(Teacher teacher, string coursename) {

            var crs = teacher.GetCourse(coursename);

            var vrs = teacher.\_lst[crs];

            vrs.Add(this);

            teacher.\_lst[crs] = vrs;

        }

    }

    static class Archive {

        private static List<Mark> lst = new List<Mark>();

        public static void AddMark(Mark mark) {

            lst.Add(mark);

        }

        public static void ShowMarks() {

            foreach (Mark mark in lst) {

                Console.WriteLine(mark.ToString());

            }

        }

    }

    class Mark {

        public int mark;

        public Teacher teacher;

        public Student student;

        public Course course;

        public Mark(int mark1, Teacher teacher1, Student student1, Course course1) {

            mark = mark1;

            teacher = teacher1;

            student = student1;

            course = course1;

        }

        public override string ToString()

        {

            return $"Student name: {student.ToString()}, Teacher name: {teacher.ToString()}, Course name: {course.ToString()}, Mark: {mark}";

        }

    }

}

Notepad.cs

using System;

using System.Collections.Generic;

namespace lab04 {

    class Notepad {

        private Dictionary<DateTime, Record> \_dates = new Dictionary<DateTime, Record>();

        public void AddRecord(DateTime dateTime, string rec) {

            if (!\_dates.ContainsKey(dateTime)) {

                \_dates.Add(dateTime, new Record());

            }

            var res = \_dates[dateTime];

            res.AddRec(rec);

            \_dates[dateTime] = res;

        }

        public void ShowRecords(DateTime dateTime) {

            if (\_dates.TryGetValue(dateTime, out Record record)) {

                foreach (string rec in record.rec) {

                    Console.WriteLine (rec);

                }

            }

        }

        public class Record {

            public List<string> rec = new List<string>();

            public void AddRec(string r) {

                rec.Add(r);

            }

        }

    }

}

Вывод: приобрёл практические навыки в области объектно-ориентированного проектирования.