**Лабораторная работа №7**

**Саутов Давид Сергеевич**

**А-12-18**

**Задание:**

Построить иерархию классов в соответствии с вариантом задания. (Иерархию классов построить с использованием виртуального класса в качестве основы иерархии)

- Разработать методы и свойства для каждого из определяемых классов.

- Показать на примере одного из методов, присутствующих в каждом классе, свойство полиморфизма.

- Реализовать для иерархии механизм интерфейсов, при этом один из классов должен реализовывать как минимум два интерфейса.

- Вывести результаты.

- Реализовать программу на C#.

1. Студент, преподаватель, персона, заведующий кафедрой.

**1. Предметная область**

Необходимо разработать приложение в **Windows Forms** на языке **С#,** содержащей следующий классы и взаимодействия между ними:Студент, преподаватель, персона, заведующий кафедрой. Класс персона будет абстрактным и от него будут наследоваться все остальные классы. Будет два интерфейса: обучение и выплаты.

**Ограничения:** Количество сданных предметов студента не может превышать 7

**Вид приложения:** Оконное приложение. Управление в меню, осуществляется с помощью кнопок.

**Среда разработки:** Visual Studio Community 2019 С# .NET FRAMEWORK

**2. Разработка программы**

**2.1 Разработка структуры программы**

В соответствии с заданием разработаем программу, я думаю разбить программу на несколько частей: файлы классов (4 файла), 2 интерфейса, вызываемая форма и файл программы.

Все поля, методы класса указаны в таблицах ниже (Табл.1 – Табл. 6)

Выходные данные будут показываться при помощи MessageBox.

Табл. 1

*Интерфейс Pay*

|  |  |
| --- | --- |
| Методы | Смысл методов |
| pay | Возвращает кол-во ожидаемых выплат |

Табл. 2

*Интерфейс Study*

|  |  |
| --- | --- |
| Методы | Смысл методов |
| study | Обучает человека |

Табл. 3

*Абстрактный класс Person*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | Тип данных | Смысл поля | Методы | Смысл методов |
| fio (protected) | string | Ф.И.О. человека | greeatings | Возвращает информацию о человеке |
| oklad (protected) | int | Оклад человека |  |  |
| studyCount (protected) | int | Количество сданных предметов/повышенний квалификиции |  |  |

Табл. 4

*Класс Student*

|  |  |
| --- | --- |
| Методы | Смысл методов |
| greeatings | Возвращает информацию о студенте |
| pay | Возвращает значение ожидаемой стипендии в зависимости от кол-ва закрытых предметов |
| study | Закрывает 1 предмет студента при вызове |

Табл. 5

*Класс Teacher*

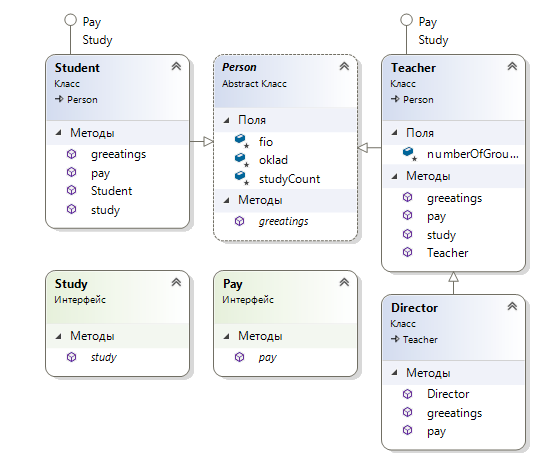
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Поля | Тип данных | Смысл поля | Методы | Смысл методов |
| numberOfGroupps | int | Кол-во групп, которые ведет преподаватель | greeatings | Возвращает информацию о преподавателе |
|  |  |  | pay | Возвращает значение зарплаты преподавателя |
|  |  |  | study | Позволяет преподавателю повысить квалификацию |

Табл. 6

*Класс Director*

|  |  |
| --- | --- |
| Методы | Смысл методов |
| greeatings | Возвращает значение следующего элемента стека |
| pay | Возвращает значение зарплаты зав. кафедры |

Ниже представлена диаграмма классов (Рис.1) для проекта lab7, разработанного в соответствии с заданием.



*Рис.1 Диаграмма классов*

Табл. 4

**2.2 Разработка алгоритма**

Стипендия студента:

закрыто 7 предметов – стипендия = 3\*оклад

закрыто от 6 до 4 предметов – стипендия = 2\*оклад

закрыто менне 4 предмотов – стипендия = 0

Зарплата преподавателю:

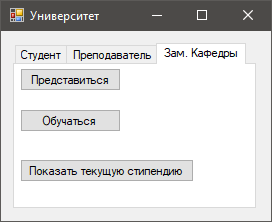
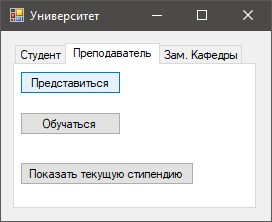
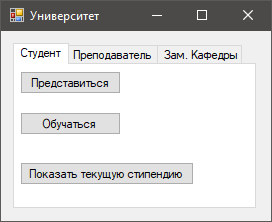
Оклад\*(1+Кол-во повышений квалификаций/10)+1000\*Кол-групп

Зарплата зав. кафедры:

Оклад\*(1+Кол-во повышений квалификаций/10)+1000\*Кол-групп + 5000

**2.3 Разработка пользовательского интерфейса**

В данной программе будет разработана форма на подобии этой (Рис. 2)



*а) б) в)*

*Рис.2 Форма где а) Взаимодействие с классом студента б) Взаимодействие с классом преподавателя в) Взаимодействие с классом зав. кафедры*

Перечень используемых элементов управления приведён на Таблице 3.

Табл.3

*Список элементов управления*

|  |  |  |
| --- | --- | --- |
| Элемент управления | Класс | Описание |
| Button1-9 | Button | Командные кнопки |
| tabControl1 | TabControl | Строка для перелистывания областей |

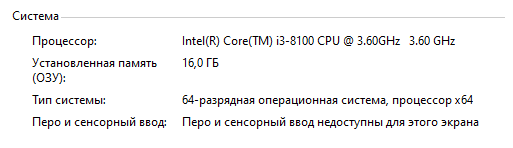
Программа предназначена для всех категорий пользователей.

**3. Реализация и тестирование приложения**

**Объект испытаний:** lab7.exe (Программа разработана на языке C#).

**Цель испытаний:** выявить недостатки программы для дальнейшей модернизации.

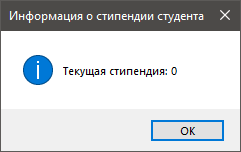
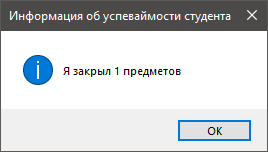
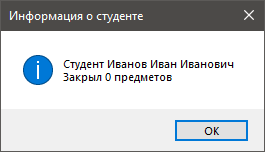
**Описание компьютера:** характеристики компьютера использовавшегося для тестирования программы приведены ниже (рис.3).



*Рис.3 Сведения о системе*

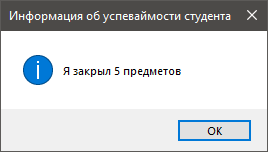
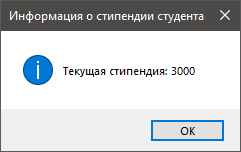
**Метод испытаний:** функциональное тестирование.

**Тестирование (**рис 4-8**)**



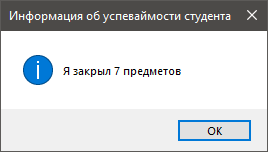
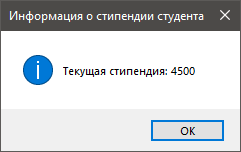
*а) б) в)*

*Рис.4 Тестирование студента при закрытых предметах <4 где а) Информация о студенте б) закрытие студентом +1 предмета в)ожидаемая стипендия при 0+1 закрытом предмете*

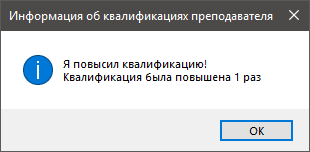
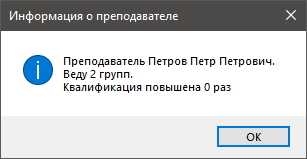
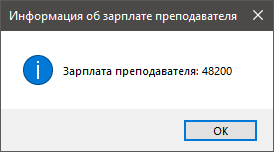
 

*а) б)*

*Рис. 5 Тестирование студента при закрытых предметах 4< и <7 где а) кол-во закрытых предметов б) ожидаемая стипендия при таком кол-ве закрытых предметах*

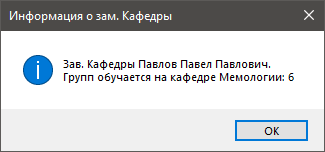
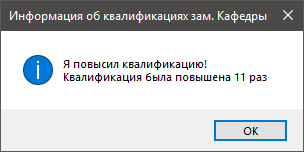
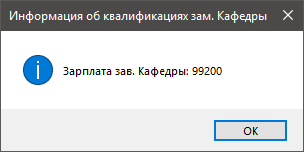
 

*а) б)*

*Рис. 6 Тестирование при закрытых предметах = 7 где а) кол-во закрытых предметов (при дальнейшей попытке обучения кол-во предметов не будет увеличиваться) б) ожидаемая стипендия при 7 закрытых предметах*

*а) б) в)*

*Рис. 7 Тестирование преподавателя где а) Информация о преподавателе б) Прохождение курсов для повышении квалификации преподавателей в) Зарплата при наличии 2 групп и 1 поыеннии квалификции*

 *а) б) в)*

*Рис. 8 Тестирование Зав. Кафедры где а) Информация о зав. Кафедры б) повышение квалификации зав. Кафедры в) Зарплата зав. Кафедры*

**Код будет представлен ниже в разделе «Листинг кода».**

**«Листинг кода»**

**Класс Person**

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab7

{

public abstract class Person

{

protected string fio;

protected int oklad;

protected int studyCount;

public abstract string greeatings();

}

}

**Интерфейс Study**

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab7

{

interface Study

{

string study();

}

}

**Интерфейс Pay**

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab7

{

interface Pay

{

int pay();

}

}

**Класс Student**

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab7

{

class Student:Person, Pay, Study

{

public Student(string fio, int oklad = 1500, int studyCount = 0) {

this.fio = fio;

this.oklad = oklad;

this.studyCount = studyCount;

}

public int pay() {

if (studyCount >= 7)

return 3 \* oklad;

if (studyCount >= 4)

return 2 \* oklad;

return 0;

}

public string study() {

if (studyCount >= 7)

studyCount = 7;

else

studyCount++;

return "Я закрыл " + studyCount.ToString() + " предметов";

}

public override string greeatings()

{

return "Студент " + fio + "\nЗакрыл " + studyCount + " предметов";

}

}

}

**Класс Teacher**

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab7

{

class Teacher:Person, Pay, Study

{

protected int numberOfGroupps;

public Teacher(string fio, int oklad = 42000, int studyCount = 0, int numberOfGroupps = 0)

{

this.fio = fio;

this.oklad = oklad;

this.studyCount = studyCount;

this.numberOfGroupps = numberOfGroupps;

}

public int pay()

{

return Convert.ToInt32(oklad \* (Convert.ToDouble(1 + studyCount/10.0))+1000\*numberOfGroupps);

}

public string study()

{

studyCount++;

return "Я повысил квалификацию!\nКвалификация была повышена " + studyCount.ToString() + " раз";

}

public override string greeatings()

{

return "Преподаватель " + fio + ".\nВеду " + numberOfGroupps.ToString() + " групп.\nКвалификация повышена "+ studyCount.ToString() +" раз";

}

}

}

**Класс Director**

using System;

using System.Collections.Generic;

using System.Text;

namespace Lab7

{

class Director:Teacher

{

public Director(string fio, int oklad = 42000, int studyCount = 10, int numberOfGroupps = 0) : base(fio, oklad, studyCount, numberOfGroupps) { }

public override string greeatings()

{

return "Зав. Кафедры "+ fio + ".\nГрупп обучается на кафедре Мемологии: " + numberOfGroupps.ToString();

}

public new int pay() {

return base.pay() + 5000;

}

}

}

**Форма**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab7

{

public partial class Form1 : Form

{

Student student = new Student("Иванов Иван Иванович");

Teacher teacher = new Teacher("Петров Петр Петрович", 42000, 0, 2);

Director director = new Director("Павлов Павел Павлович", 42000, 10, 6);

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

MessageBox.Show(

student.greeatings(),

"Информация о студенте",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button2\_Click(object sender, EventArgs e)

{

MessageBox.Show(

student.study(),

"Информация об успеваймости студента",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button3\_Click(object sender, EventArgs e)

{

MessageBox.Show(

"Текущая стипендия: " + student.pay().ToString(),

"Информация о стипендии студента",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button6\_Click(object sender, EventArgs e)

{

MessageBox.Show(

teacher.greeatings(),

"Информация о преподавателе",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button5\_Click(object sender, EventArgs e)

{

MessageBox.Show(

teacher.study(),

"Информация об квалификациях преподавателя",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button4\_Click(object sender, EventArgs e)

{

MessageBox.Show(

"Зарплата преподавателя: "+ teacher.pay().ToString(),

"Информация об зарплате преподавателя",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button9\_Click(object sender, EventArgs e)

{

MessageBox.Show(

director.greeatings(),

"Информация о зам. Кафедры",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button8\_Click(object sender, EventArgs e)

{

MessageBox.Show(

director.study(),

"Информация об квалификациях зам. Кафедры",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

private void button7\_Click(object sender, EventArgs e)

{

MessageBox.Show(

"Зарплата зав. Кафедры: " + director.pay().ToString(),

"Информация об квалификациях зам. Кафедры",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

}

}

**Программа**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab7

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

**Вывод:** Во время выполнения данной работы я познакомился с основами ООП в C#, создал иерархию классов и реализовал представленные мне классы. Программа работает корректно. Программу можно доработать, добавив возможность добавления студентов, найма преподавателей и повышение преподавателей до зав. Кафедры или же наоборот, понижение.