МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 3

з дисципліни “Основи програмування”
тема “Структурні шаблони проектування”

Виконав	Перевірив
студент ІІ курсу	“____” “_____” 20___ р.
групи КП-01	викладач
Беліцький Олександр Сергійович (*прізвище, ім’я, по батькові*)	Заболотня Тетяна Миколаївна (*прізвище, ім’я, по батькові*)

варіант № 3

Київ 2021

Постановка завдання

Завдання 1.

За допомогою відповідного шаблона проєктування створити структуру класів, до якої будуть входити класи, які реалізуються у об’єкти «Студент» та «Група студентів». Обидва об’єкти повинні мати імена та метод «Показати інформацію». Крім того, *з*абезпечити підтримку об’єктами метода «Скласти іспит».

Завдання 2.

В системі дистанційної освіти реалізувати механізм, який буде дозволяти користувачу он-лайн курсу виконувати завдання з поточного модуля (генерувати список завдань з наявного загального масиву завдань) тільки за умови наявності виконаних завдань (позитивних оцінок) з минулих модулів.

Аналіз вимог і проектування

Завдання 1.

У даній задачі доцільно використати шаблон Компонувальник.

Вибір впав саме на цей шаблон, адже він забезпечує роботу об'єкта і групу йому подібних. Доцільно, коли користувачі однаково повинні керувати як цілими об'єктами,так і їх складовими частинами. Тобто ціле та його частини повинні реалізувати один і той самий інтерфейс.

А "Студент" і "Група студентів" - саме той варіант.

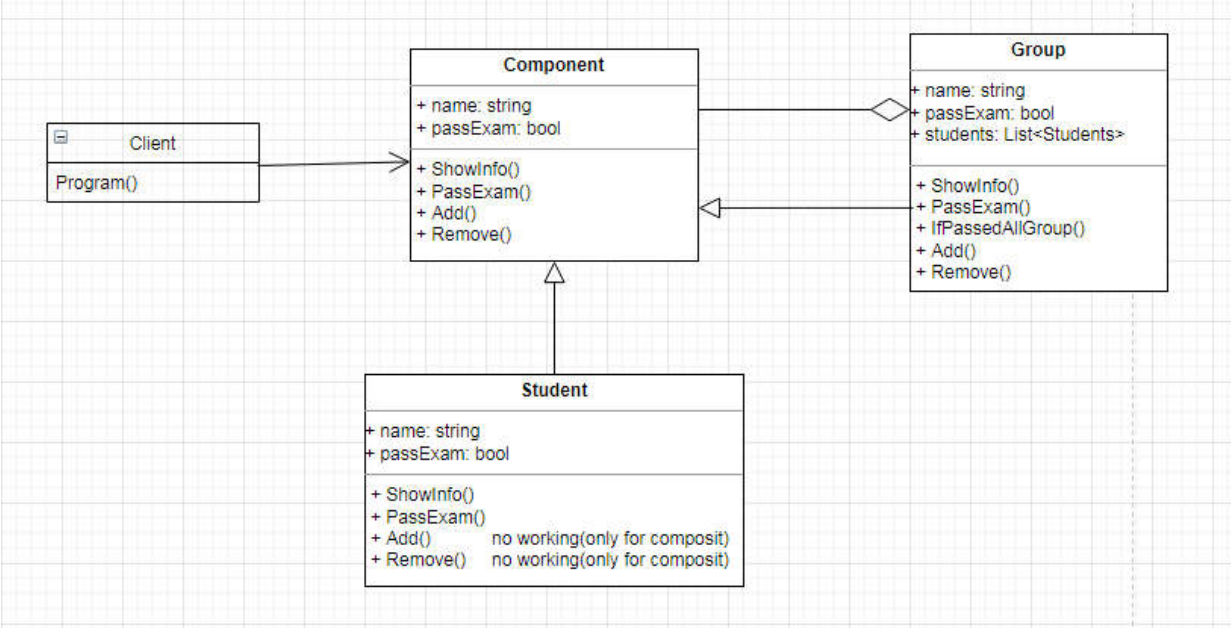

рис.1 UML-діаграма класів

Завдання 2.

У даній задачі доцільно використати шаблон Заступник.

Паттерн Заступник (Proxy) надає об'єкт-заступник, який керує доступом до іншого об'єкта. Тобто створюється об'єкт-сурогат, який може виступати в ролі іншого об'єкта та замінювати його.

В даній задачі нам потрібно контролювати доступ студента до можливості виконання завдань з наступного модуля при наявності позитивних оцінок з попереднього.

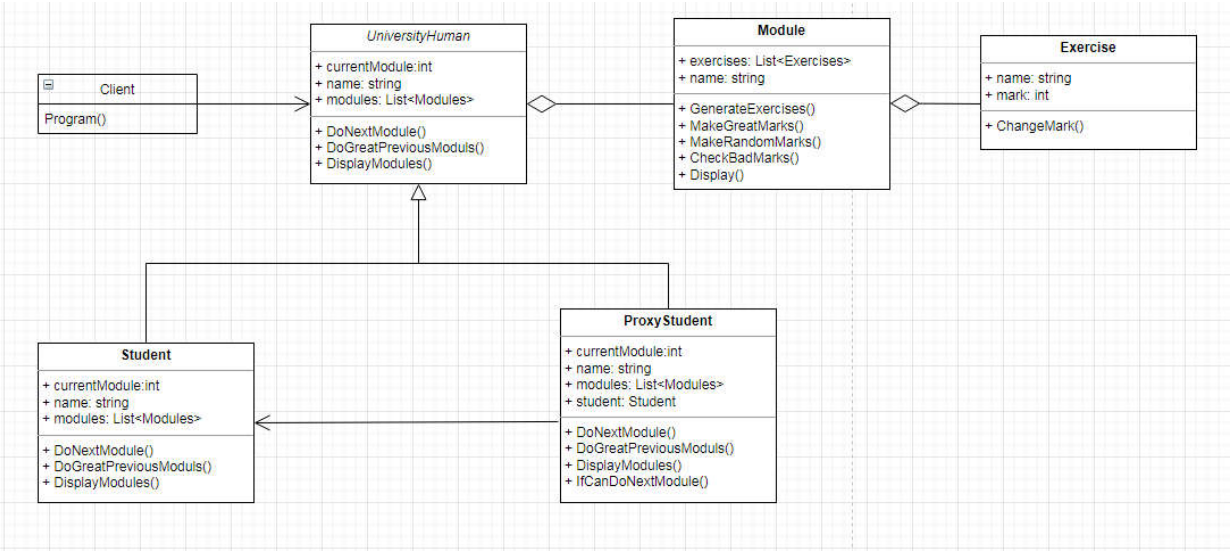

рис.2 UML-діаграма класів

Тексти коду програм

Завдання 1.

Program.cs

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

class Program
{
    static void Main(string[] args)
    {
        Group group1 = new Group("First");
        group1.Add(new Student("Igor"));
        group1.Add(new Student("Nata"));
        group1.Add(new Student("Georg"));

        group1.ShowInfo(1);
        group1.PassExam();
        Console.WriteLine();

        group1.ShowInfo(1);

        Console.WriteLine("=====");

        Group group2 = new Group("Second");
        Student gerd = new Student("Gerd");

        group2.Add(gerd);
        group2.Add(new Student("Bob"));
        group2.Add(new Student("Pidge"));

        group2.ShowInfo(1);
        gerd.PassExam();
        Console.WriteLine();

        group2.ShowInfo(1);
    }
}

abstract class Component //основа
{
    protected string _name;
    protected bool _passExam;

    public Component(string name)
    {
        this._name = name;
        this._passExam = false;
    }

    public abstract void ShowInfo(int indent);
    public abstract bool PassExam();
    public abstract void Add(Component a);
    public abstract void Remove(Component a);
}

class Student : Component
{
    public Student(string name) : base(name) { }

    public override void ShowInfo(int indent)
    {
        Console.WriteLine(new String('-', indent) + "+ " + _name + "\t" + _passExam);
    }
    public override bool PassExam()
    {
        return this._passExam = true;
    }
```

```
    public override void Add(Component student)
    {
        Console.WriteLine("Cannot add to student");
    }


    public override void Remove(Component student)
    {
        Console.WriteLine("Cannot remove from student");
    }
}

class Group : Component  //компонувальник
{
    private List<Component> students = new List<Component>();

    public Group(string name) : base(name) { }

    public override void Add(Component a)
    {
        students.Add(a);
    }

    public override void Remove(Component a)
    {
        students.Remove(a);
    }

    public override void ShowInfo(int indent)
    {
        Console.WriteLine(new String('-', indent) + "+ " + _name + "\t" + _passExam);

        foreach (var student in students)
        {
            student.ShowInfo(indent + 2);
        }
    }

    private void IfPassedAllGroup()
    {
        bool ifPassedAllGroup = true;
        foreach (var student in students)
        {
            if (!student.PassExam())
            {
                ifPassedAllGroup = false;
            }
        }
        _passExam = ifPassedAllGroup;
    }

    public override bool PassExam()
    {
        IfPassedAllGroup();
        foreach (var student in students)
        {
            student.PassExam();
        }
        return true;
    }
}
```

Завдання 2.

Program.cs

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

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

        ProxyStudent student = new ProxyStudent("Gerd");
        student.DisplayModules();
        Console.WriteLine();

        student.DoNextModule();
        Console.WriteLine();

        student.DoGreatPrevousModuls();
        student.DisplayModules();

        student.DoNextModule();
        Console.WriteLine();
        student.DisplayModules();
    }
}

abstract class UnivetsityHuman
{
    protected string _name;
    protected int currentmodule;
    protected List<Module> modules;

    public UnivetsityHuman(string name)
    {
        this._name = name;
        this.currentmodule = 1;
        modules = new List<Module>();
        for (int i = 0; i < 2; i++)
        {
            modules.Add(new Module($"Module {i + 1}"));
        }
    }
    public abstract void DoNextModule();
    public abstract void DoGreatPrevousModuls();
    public abstract void DisplayModules();
}

class ProxyStudent : UnivetsityHuman
{
    Student student;
    public ProxyStudent(string name) : base(name)
    {
        student = new Student(name);
        this.modules = student.Modules;
    }

    public override void DisplayModules()
    {
        student.DisplayModules();
    }

    public override void DoGreatPrevousModuls()
    {
        student.DoGreatPrevousModuls();
    }

    private bool IfCanDoNextModule()
    {
        if (modules[currentmodule - 1].CheckBadMark())
            return false;
        return true;
    }
    public override void DoNextModule()
    {
        if (IfCanDoNextModule())
        {
            student.DoNextModule();
```

```
            Console.WriteLine("Next module done");
        }
        else
            Console.WriteLine("You should do previous modules great (>= 10) ");
    }
}
class Student : UnivetsityHuman
{
    public Student(string name) : base(name) { }

    public List<Module> Modules
    {
        get { return this.modules; }
    }
    public override void DisplayModules()
    {
        foreach (var item in modules)
        {
            Console.WriteLine($"{item.Name}");
            item.Display();
            Console.WriteLine();
        }
    }

    public override void DoGreatPrevousModuls()
    {
        for (int i = 0; i < currentmodule; i++)
        {
            modules[i].MakeGreatMarks();
        }
    }

    public override void DoNextModule()
    {
        currentmodule++;
        modules[currentmodule - 1].MakeRandomMarks();
    }
}

class Module
{
    private List<Exercise> _exercises;
    private string _name;

    public string Name
    {
        get { return _name; }
    }
    public Module(string name)
    {
        this._name = name;
        this._exercises = GenerateExercises();
    }

    private List<Exercise> GenerateExercises()
    {
        Random rnd = new Random();
        List<Exercise> exercises = new List<Exercise>();
        string[] randomExercises = { "Math", "PE", "IT", "Biology", "History", "English",
"Literature" };

        string[] randomFirstFour = randomExercises.OrderBy(x => rnd.Next()).ToArray();

        for (int i = 0; i < 4; i++)
        {
            Exercise ex = new Exercise(randomFirstFour[i]);
            exercises.Add(ex);
        }
        return exercises;
    }

    public void Display()
    {
```

```
        foreach (var item in _exercises)
        {
            Console.WriteLine($"{item.Name,-15}\t{item.Mark,4}");
        }
    }

    public void MakeGreatMarks()
    {
        Random rnd = new Random();
        foreach (var item in _exercises)
        {
            item.ChangeMark(rnd.Next(10, 13));
        }
    }

    public void MakeRandomMarks()
    {
        Random rnd = new Random();
        foreach (var item in _exercises)
        {
            item.ChangeMark(rnd.Next(1, 13));
        }
    }

    public bool CheckBadMark()
    {
        foreach (var item in _exercises)
        {
            if (item.Mark < 10)
                return true;
        }
        return false;
    }

}

class Exercise
{
    private string _name;
    private int _mark;

    public Exercise(string name)
    {
        this._name = name;
        //Random rnd = new Random();
        //this._mark = rnd.Next(1,10);
        this._mark = 0;
    }

    public void ChangeMark(int mark)
    {
        if (mark > 12 || mark < 0)
            throw new ArgumentException("Error: mark must be from 0 to 12");
        this._mark = mark;
    }

    public string Name
    {
        get { return _name; }
    }

    public int Mark
    {
        get { return _mark; }
    }
}
```

Приклади результатів

Завдання 1.

Створимо дві групи студентів "First" та "Second".

Студенти першої групи до сесії у полях зданої сесії мають значення False.

```
-+ First        False
---+ Igor       False
---+ Nata       False
---+ Georg      False
```

Вчителі вирішили поставити всій групі "автомат", тому вся група здала сесію(в полях здачі сесії значення True). Значення напроти назви групи означає чи вся група здала сесію.

```
-+ First        True
---+ Igor       True
---+ Nata       True
---+ Georg      True
```

Студенти другої групи до сесії у полях зданої сесії мають значення False.

```
-+ Second       False
---+ Gerd       False
---+ Bob        False
---+ Pidge      False
```

Але у другій групі вчителі вирішили провести екзамени і здав лише студент "Gerd". Тому лише у нього зміниться значення в полі здачі сесії, а всі інші з групи підуть на перездачу.

```
-+ Second       False
---+ Gerd       True
---+ Bob        False
---+ Pidge      False
```

Завдання 2.

Маємо два модулі, які має виконати студент "Module 1" та "Module 2". Спочатку студент знаходиться на першому модулі і, пропустивши його виконання, намагається виконати завдання із другого. Йому виводиться помилка, що він має виконати попередній модуль на відмінно.

```
Module 1
Literature        0
Biology           0
IT                0
English           0

Module 2
IT                0
History           0
Biology           0
Literature        0


You should do previous modules great (>= 10)
```

Після проходження першого модуля студент має відмінні оцінки.

```
Module 1
Literature       10
Biology          10
IT               12
English          12

Module 2
IT                0
History           0
Biology           0
Literature        0
```

Студент намагається виконати наступний модуль і програма йому це дозволяє, після чого його оцінки з двох модулів становлять:

```
Next module done

Module 1
Literature       10
Biology          10
IT               12
English          12

Module 2
IT                3
History           4
Biology          12
Literature       12
```

Висновки

Виконавши дану лабораторну роботу було проведено вивчення структурних шаблонів, їх особливостей, основних задач, принципів роботи та варіанти для вдалого використання.

Я визначив для себе, що структурні шаблони в основному пов'язані з композицією об'єктів, іншими словами, про те, як сутності можуть використовувати один одного.

Окремо детально було розглянуто патерни Заступник та Компонувальник. Визначив, що їх можна охарактеризувати як:

- Проксі замінює інший об'єкт для контролю доступу до нього.
- образно реалізацію Компонувальника можна уявити як меню, що має різні пункти. Ці пункти можуть містити підменю, в яких також є пункти.

Компіляція всього коду відбувалася за допомогою утиліти dotnet.