

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 “Програмування 1. Об'єктно-орієнтоване програмування та шаблони проєктування”

тема “ C# .Net. Структурні шаблони у мові C#”

|  |  |  |
| --- | --- | --- |
| Виконала  студентка IІ курсу  групи КП-02  Гудзіцька Дарина Сергіївна  Варіант №3 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Заболотня Тетяна Миколаївна  (*прізвище, ім’я, по батькові*) |

Київ 2021

**Мета роботи**

Ознайомитися з такими можливостями мови програмування C# як структурні шаблони проєктування, набути розуміння як і коли їх використовувати.

**Постановка 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».
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.

**Обгрунтування вибору шаблону**

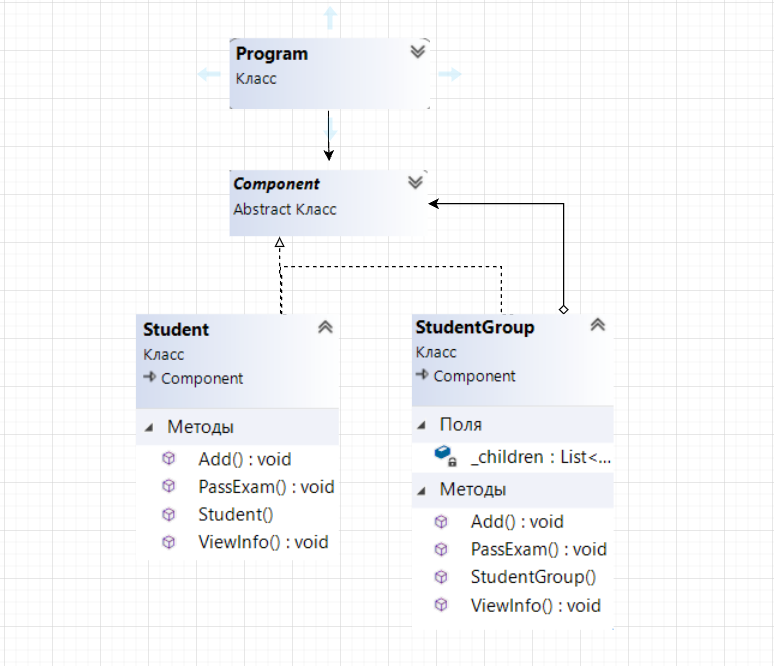
**Завдання 1. Composite**

Дана структура класів може бути представлена у вигляді дерева. Тобто, «Група студентів» може містити як і просто «Студента», так і іншу «Групу студентів» зі своєю ієрархією.

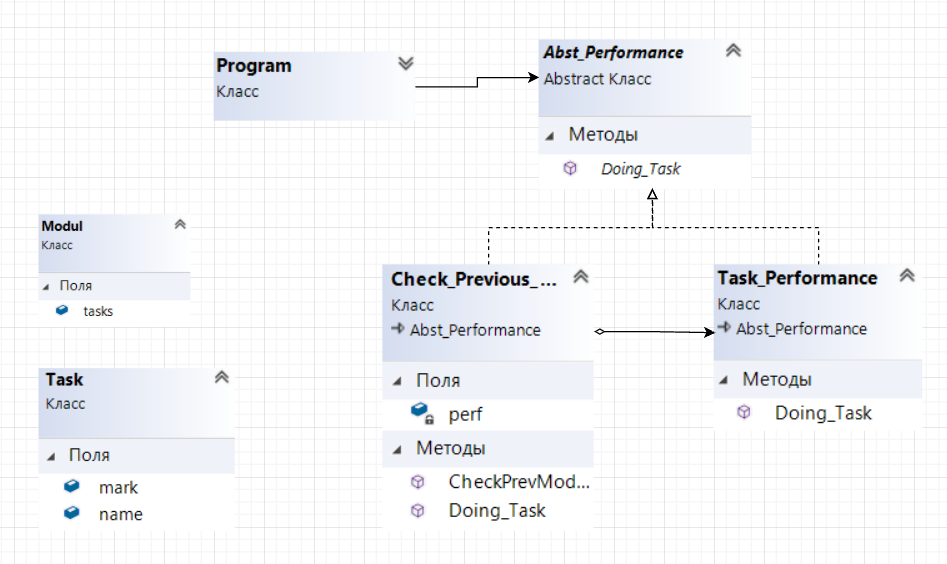
**Завдання 2. Proxy**

Не має сенсу створювати метод «Робити завдання», коли ми не впевнені чи зробимо ми це завдання, адже потрібно переконатися, що завдання з попередніх модулів виконані з задовільною оцінкою. Тому найкращим рішенням буде створити заступника і перевірити його, а коли всі умови виконані – створити реальний об’єкт.

**UML діаграма класів**

**Завдання 1.**

**Завдання 2.**



**Фрагменти коду**

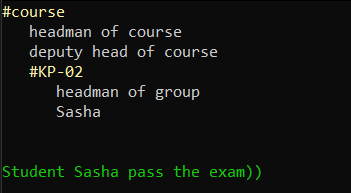
**Завдання 1.**

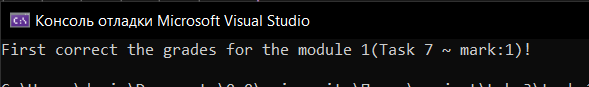
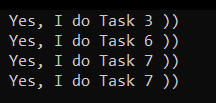
|  |
| --- |
| using System;  using System.Collections.Generic;  namespace task\_1  {  abstract class Component  {  protected string name;  public Component(string name)  {  this.name = name;  }  public abstract void Add(Component c);  public abstract void ViewInfo(int step);  public abstract void PassExam();  }  class StudentGroup : Component  {  private List<Component> \_children = new List<Component>();  // Constructor  public StudentGroup(string name)  : base(name)  {  }  public override void Add(Component component)  {  \_children.Add(component);  }  public override void ViewInfo(int step)  {  Console.ForegroundColor = ConsoleColor.Yellow;  Console.WriteLine(new String(' ', step) + $"#{name}");  foreach (Component component in \_children)  {  component.ViewInfo(step + 3);  }  }  public override void PassExam()  {  Console.WriteLine("For the group to pass the exam, 70% of the students need to pass.");  }  }  class Student : Component  {  // Constructor  public Student(string name)  : base(name)  {  }  public override void Add(Component c)  {  Console.WriteLine("Nothing can be added to the student");  }  public override void ViewInfo(int step)  {  Console.ForegroundColor = ConsoleColor.Gray;  Console.WriteLine(new String(' ', step) + name);  }  public override void PassExam()  {  Console.ForegroundColor = ConsoleColor.Green;  Random rand = new Random();  if (rand.Next() % 2 == 0)  Console.WriteLine($"Student {this.name} pass the exam))");  else  Console.WriteLine($"Sorry, {this.name}, you must retake((");  }  }  } |

**Завдання 2.**

|  |
| --- |
| using System;  using System.Collections.Generic;  namespace task\_2  {  public class Task  {  public string name;  public int mark;  }  public class Modul  {  public List<Task> tasks;  }  public abstract class Abst\_Performance  {  public abstract void Doing\_Task(Modul m);  }  public class Task\_Performance : Abst\_Performance  {  public override void Doing\_Task(Modul m)  {  foreach(Task t in m.tasks)  Console.WriteLine($" Yes, I do {t.name} ))");  }  }  public class Check\_Previous\_Task : Abst\_Performance  {  Task\_Performance perf = new Task\_Performance();  public override void Doing\_Task(Modul current)  {  perf.Doing\_Task(current);  }  public void CheckPrevModule(List<Modul> course)  {  for (int i = 0; i < course.Count - 1; i++)  {  if (course[i].tasks == null)  {  Console.WriteLine("First do the tasks from the previous module!");  return;  }  foreach (Task t in course[i].tasks)  {  if (t.mark == 0 || t.mark <= 3)  {  Console.WriteLine($"First correct the grades for the module {i+1}({t.name} ~ mark:{t.mark})!");  return;  }  }  }    Doing\_Task(course[course.Count - 1]);  }  } } |

**Приклади результатів**

**Завдання 1.**

**Завдання 2.**

**Висновки**

Виконавши дану лабораторну роботу відбулося ознайомлення з такими можливостями мови програмування C# як структурні шаблони проєктування, було набуте розуміння як і коли їх використовувати.