Зразок оформлення титульного аркуша лабораторної роботи

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

„КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**ЛАБОРАТОРНА РОБОТА No1**

з дисципліни «Об’єктно-орієнтоване програмування»

на тему

Реалізація структурних шаблонів проектування

Варіант No16

Виконав студент

ІІ курсу групи КП-ХХ

Палій Дмитро Володимирович

залікова книжка КП-8321

Перевірив

доцент, к.т.н. Заболотня Т.М.

Оцінка

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата, підпис)

КИЇВ 2020

**Мета роботи**

Ознаойомлення з основними характеристиками шаблонів «Декоратор», «Компонувальник» та «Міст», запам’ятовування поширених ситуацій, коли використання цих шаблонів є доцільним, набуття вмінь та навичок реалізації шаблонів під час створення програмного коду.

**Постановка завдання**

Варіант 16

1. Реалізувати декілька різноманітних способів збегірання текста до файлу. Перший спосіб – звичайний: просто зберігаємо текст без жодних змін. Другий спосіб – видаляємо всі зайві пробіли з тексту перед збереженням. Третій спосіб – застосовуємо кодування текста (або архівацію). В залежності від необхідності забезпечити збереження тексту до файлу доцільним способом.

2. За допомогою шаблона проектування реалізувати модель бібліотеки. Бібліотека містить відділи, відділи – стелажі з різної тематики (наприклад, література 19 ст., детективи тощо), стелажі складаються з полиць, які в свою чергу містять книжки. Кожна книга має свій унікальний код та шифр. Книг з однаковим кодом не може бути. А екземпляри однієї книжки можуть мати однаковий шифр. Забезпечити виведення на екран інформації про об’єкти будь-якого типу (книга, полиця тощо). Крім того, реалізувати виведення кількості найменувань (не екземплярів!) книг на

полиці, стелажі, відділі, а також загальну кількість книг на полиці, стелажі, у відділі.

**Обгрунтування вибору шаблонів**

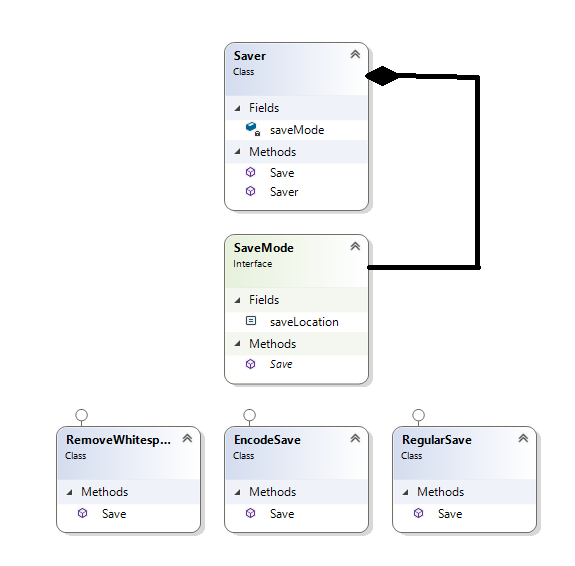
**Завдання 1:** обрано шаблон «Міст», оскільки конкретну реалізацію зберігання необхідно обрати під час виконання програми.

**Завдання 2:** обрано шаблон «Компонувальник», оскільки у наявності є оригінальна структура, що складається з об’єктів та композицій об’єктів.

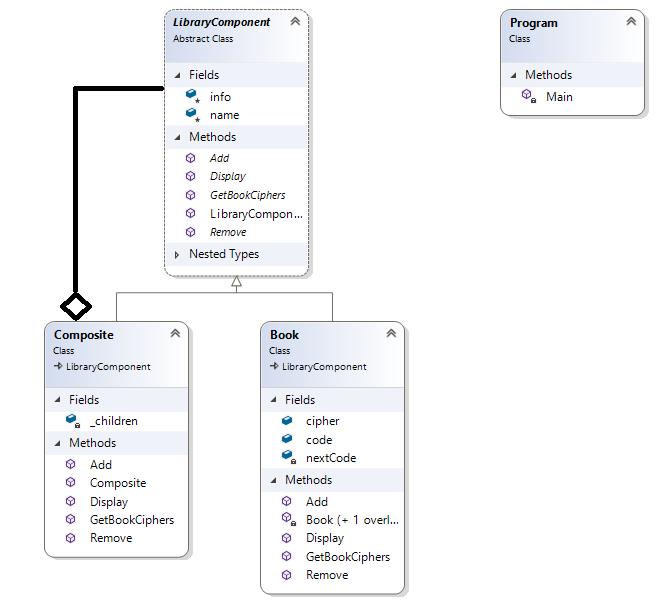
**Тексти коду програм**

Діаграма класів:

**Завдання 1:**

****

**Завдання 2:**

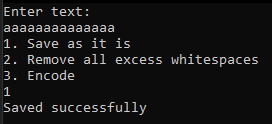
****

|  |
| --- |
| **Завдання 1** |
| interface SaveMode  {  bool Save(string text);  }  class Saver  {  SaveMode saveMode;  public Saver(SaveMode mode)  {  this.saveMode = mode;  }  public bool Save(string text)  {  return this.saveMode.Save(text);  }  }  class RegularSave : SaveMode  {  public bool Save(string text)  {  File.WriteAllText(saveLocation, text);  return true;  }  }  class RemoveWhitespacesSave : SaveMode  {  public bool Save(string text)  {  StringBuilder sb = new StringBuilder();  bool removeWhitespace = true;  for (int i = 0; i < text.Length; i++)  {  if (!removeWhitespace && Char.IsWhiteSpace(text[i]))  {  removeWhitespace = true;  sb.Append(text[i]);  }  else if (!Char.IsWhiteSpace(text[i]))  {  removeWhitespace = false;  sb.Append(text[i]);  }  }  File.WriteAllText(saveLocation, sb.ToString());  return true;  }  }  class EncodeSave : SaveMode  {  public bool Save(string text)  {  var plainTextBytes = System.Text.Encoding.UTF8.GetBytes(text);  File.WriteAllText(saveLocation, Convert.ToBase64String(plainTextBytes));  return true;  }  }  } |

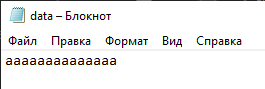
|  |
| --- |
| **Завдання 2** |
| abstract class LibraryComponent  {  public class BookInfo  {  public int titles;  public int total;  public BookInfo(int total, int titles)  {  this.total = total;  this.titles = titles;  }  public override string ToString()  {  return "total: " + this.total.ToString() + " titles: " + this.titles.ToString();  }  }  protected string name;  protected BookInfo info;  public LibraryComponent(string name)  {  this.name = name;  }  public abstract void Add(LibraryComponent c);  public abstract void Remove(LibraryComponent c);  public abstract List<int> GetBookCiphers();  public abstract void Display(int depth);  }  class Composite : LibraryComponent  {  private List<LibraryComponent> \_children = new List<LibraryComponent>();  public Composite(string name) : base(name)  {  this.info = new BookInfo(0, 0);  }  public override void Add(LibraryComponent component)  {  \_children.Add(component);  }  public override void Remove(LibraryComponent component)  {  \_children.Remove(component);  }  public override void Display(int depth)  {  Console.WriteLine(new String('-', depth) + name + " " + this.info.ToString());  foreach (LibraryComponent component in \_children)  component.Display(depth + 2);  }  public override List<int> GetBookCiphers()  {  var list = new List<int>();  var uniques = new List<int>();  foreach (LibraryComponent component in \_children)  {  var compList = component.GetBookCiphers();  list.AddRange(compList);  }  this.info.total = list.Count;  foreach (int code in list)  {  if (!uniques.Contains(code)) uniques.Add(code);  }  this.info.titles = uniques.Count;  return list;  }  }  class Book : LibraryComponent  {  static private int nextCode;  public readonly int code;  public readonly int cipher;  static Book()  {  nextCode = 0;  }  public Book(string name) : base(name)  {  this.cipher = name.GetHashCode();  this.code = nextCode;  nextCode += 1;  this.info = new BookInfo(0, 0);  }  public override void Add(LibraryComponent c)  {  Console.WriteLine("Impossible operation");  }  public override void Remove(LibraryComponent c)  {  Console.WriteLine("Impossible operation");  }  public override void Display(int depth)  {  Console.WriteLine(new String('-', depth) + name + " code:" + this.code.ToString() + " cipher: " + this.cipher.ToString());  }  public override List<int> GetBookCiphers()  {  var list = new List<int>();  list.Add(this.cipher);  return list;  }  } |

**Результати виконання**

**Завдання 1**

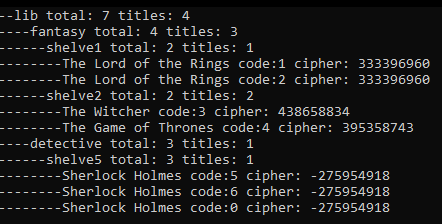
****

*Консольний інтерфейс*

**

*Збережений текст*

**Завдання 2**

****

**Висновки**

У цій лабораторній роботі я ознайомився з основними характеристиками шаблонів «Декоратор», «Компонувальник» та «Міст», запам’ятав поширені ситуації, коли використання цих шаблонів є доцільним, набув вмінь та навичок реалізації шаблонів під час створення програмного коду.