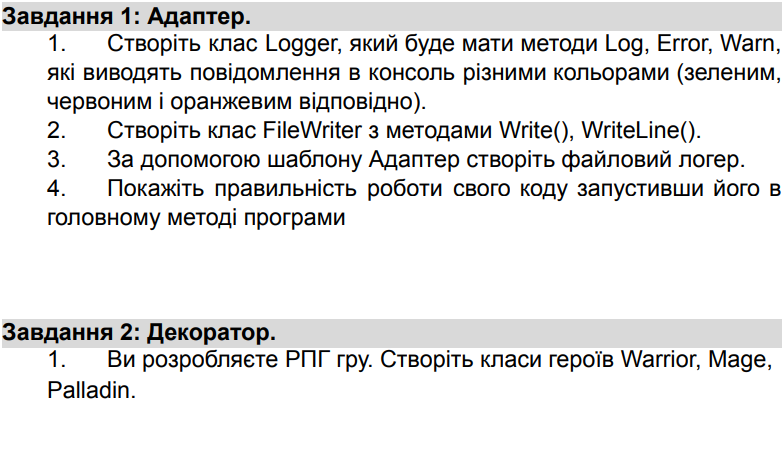
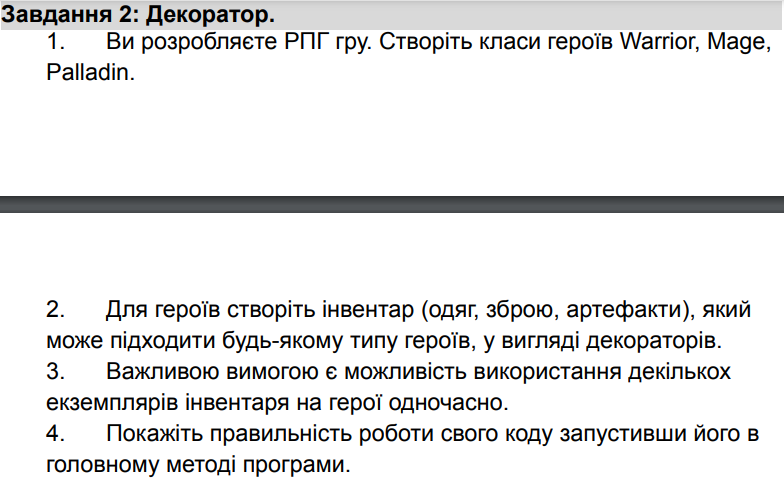
**Лабораторна робота № 4**

**СТРУКТУРНІ ШАБЛОНИ**

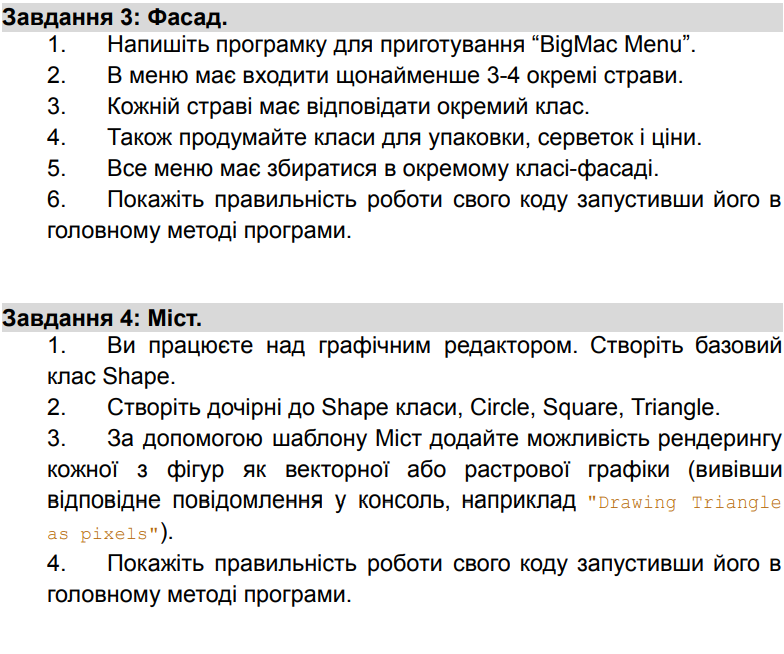
**Мета:** навчитися реалізовувати структурні шоблони проєктування Адаптер, Декоратор, Фасад, Міст.

**Хід роботи:**

****

****

# 



**Завдання 1:** Адаптер.

Створення класу **Logger**

Лістинг коду класу **Logger:**

using PatternsPartTwo.Adapter.Interfaces;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Adapter

{

internal class Logger :ILogger

{

public string Message { get; set; } = "Simple sentence";

public virtual void Log()

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine($"Prompt: {Message}");

Reset();

}

private void Reset()

{

Console.ForegroundColor = ConsoleColor.White;

}

public virtual void Error()

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine($"Erorr: {Message}");

Reset();

}

public virtual void Warn()

{

Console.ForegroundColor = ConsoleColor.DarkYellow;

Console.WriteLine($"Warning: {Message}");

Reset();

}

}

}

Лістинг коду:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Adapter

{

internal class FileWriter

{

private static string \_path = $"{Directory.GetCurrentDirectory()}/Logs.txt";

public static void Write(string message)

{

File.AppendAllText(\_path, message);

}

public static void WriteLine(string message)

{

File.AppendAllText(\_path,$"{message}\n");

}

}

}

namespace PatternsPartTwo.Adapter

{

internal class FileLogger : ILogger

{

public void Error()

{

ExplainLine("XD");

FileWriter.WriteLine("An erorr occurred ");

ExplainLine("XD");

}

private void ExplainLine(string line)

{

FileWriter.WriteLine($"\n-----------------------{line}-----------------------\n");

}

public void Log()

{

ExplainLine("XD");

FileWriter.WriteLine("The best way to do this it`s....");

ExplainLine("XD");

}

public void Warn()

{

ExplainLine("XD");

FileWriter.WriteLine("An warning occurred");

ExplainLine("XD");

}

}

}

namespace PatternsPartTwo.Adapter

{

internal class FileLoggerAdapter : Logger

{

private FileLogger \_logger;

public FileLoggerAdapter(FileLogger logger)

{

\_logger = logger;

}

public override void Log()

{

\_logger.Log();

}

public override void Error()

{

\_logger.Error();

}

public override void Warn()

{

\_logger.Warn();

}

}

}

# Результат виконання програми:

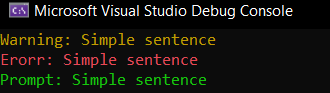


Рис.4.1. Реалізація завдання.

**Завдання 2:** Декоратор.

Класи героїв

Лістинг коду**:**

using PatternsPartTwo.Decorator.Interface;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Decorator.Heroes

{

internal class Mage : Hero

{

public override int GetArmor()

{

return Armor += 2;

}

public override int GetDamage()

{

return Damage += 8;

}

public override int GetHitPoints()

{

return HitPoints += 3;

}

}

}

namespace PatternsPartTwo.Decorator.Heroes

{

internal class Palladin : Hero

{

public override int GetArmor()

{

return Armor += 6;

}

public override int GetDamage()

{

return Damage += 2;

}

public override int GetHitPoints()

{

return HitPoints += 7;

}

}

}

namespace PatternsPartTwo.Decorator.Heroes

{

internal class Warrior : Hero

{

public override int GetArmor()

{

return Damage += 3;

}

public override int GetHitPoints()

{

return HitPoints += 5;

}

public override int GetDamage()

{

return Damage += 3;

}

}

}

Реалізація інвентару

Лістинг коду:

using PatternsPartTwo.Decorator.Interface;

using PatternsPartTwo.Decorator.Items;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Decorator

{

internal abstract class BaseHeroSet : Hero

{

protected Hero \_hero;

public BaseHeroSet(Hero hero) : base()

{

\_hero = hero;

Inventory = new List<string>();

}

}

}

Результат виконання програми:

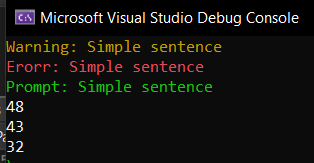


Рис.4.2. Реалізація завдання.

**Завдання 3:** Фасад.

Створення класу для BigMacMenu

Лістинг коду:

using PatternsPartTwo.Facade.Dishes;

using PatternsPartTwo.Facade.ServItems;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Facade

{

internal class BigMacFacade

{

private CheeseBurger \_cheeseBurger;

private ChikenBurger \_chikenBureger;

private DoubleBurger \_doubleBurger;

private Potatoes \_potatoes;

private Packing \_packingService;

private Napkings \_naps;

private Price \_price;

public BigMacFacade()

{

\_cheeseBurger = new CheeseBurger();

\_chikenBureger = new ChikenBurger();

\_doubleBurger = new DoubleBurger();

\_potatoes = new Potatoes();

\_packingService = new Packing();

\_naps = new Napkings();

\_price = new Price();

}

public void AllInclusive()

{

\_packingService.StartPacking("green");

Console.WriteLine($"{\_cheeseBurger.Name}\n{\_chikenBureger.Name}\n{\_doubleBurger.Name}\n{\_potatoes.Name}\n" +

$"price:{\_price.ComputeDiscount(\_price.ComputePrice(4), \_price.GetDiscount(4))}$, napks count:{\_naps.GetForBigPortion()}");

\_packingService.Message = "(: See you later :)";

\_packingService.EndPacking("blue");

}

public void PotatoCheese()

{

\_packingService.StartPacking("blue");

Console.WriteLine($"{\_potatoes.Name}\n{\_cheeseBurger.Name},naps count:{\_naps.ComputeForPeople(2)},price {\_price.ComputePrice(2)}");

\_packingService.Message = "Have a nice day!";

\_packingService.EndPacking("green");

}

public void DoubleCheese()

{

\_packingService.StartPacking("yellow");

Console.WriteLine($"{\_cheeseBurger.Name}\n{\_doubleBurger.Name},naps count:{\_naps.GetDefaultCout()},price {\_price.ComputePrice(2)}");

\_packingService.Message = "Сome back for more!";

\_packingService.EndPacking("blue");

}

}

}

Клас із 4-ма видами страв

Лістинг коду:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Facade.Dishes

{

internal class CheeseBurger : IDish

{

public string Name => "Burger with cheese";

}

}

namespace PatternsPartTwo.Facade.Dishes

{

internal class ChikenBurger : IDish

{

public string Name => "Burger with chicken";

}

}

namespace PatternsPartTwo.Facade.Dishes

{

internal class DoubleBurger : IDish

{

public string Name => "Double burger";

}

}

namespace PatternsPartTwo.Facade.Dishes

{

internal class Potatoes : IDish

{

public string Name => "Potatoes x20";

}

}

Створення класу для пакування

Лістинг коду:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Facade.ServItems

{

internal class Packing

{

public string Message { get; set; } = "Thank you for you order :)";

private void PrintPacking()

{

Console.WriteLine("\\------------------------------------------------/\n" +

$" {Message}\n"+

"\\------------------------------------------------/");

}

private void ResetMessage()

{

Message = "Thank you for you order :)";

}

private void GetPackingColor(string name)

{

switch (name)

{

case "green": Console.ForegroundColor = ConsoleColor.Green;

break;

case "yellow": Console.ForegroundColor = ConsoleColor.Yellow;

break;

case "blue":

Console.ForegroundColor = ConsoleColor.Blue;

break;

default: Console.WriteLine("Packing not exist");

break;

}

}

private void Reset()

{

Console.ForegroundColor = ConsoleColor.White;

}

public void StartPacking(string color)

{

GetPackingColor(color);

PrintPacking();

Reset();

ResetMessage();

}

public void EndPacking(string color)

{

GetPackingColor(color);

PrintPacking();

Reset();

ResetMessage();

Console.WriteLine("\n");

}

}

}

namespace PatternsPartTwo.Facade.ServItems

{

internal class Price

{

public double ComputePrice(int dishesCount)

{

return 50 \* dishesCount;

}

public double ComputeDiscount(double price,int discount)

{

return price-=price \* discount/100;

}

public int GetDiscount(int dishesCount)

{

if (dishesCount > 3)

{

return 20;

}

if (dishesCount > 2)

{

return 5;

}

return 1;

}

}

}

namespace PatternsPartTwo.Facade.ServItems

{

internal class Napkings

{

public int Count { get; set; } = 4;

public int GetDefaultCout()

{

return Count;

}

public int GetDoubleCout()

{

return Count\*2;

}

public int ComputeForPeople(int people)

{

return 2\*people;

}

public int GetForBigPortion()

{

return 20;

}

}

}

Результат виконання програми:

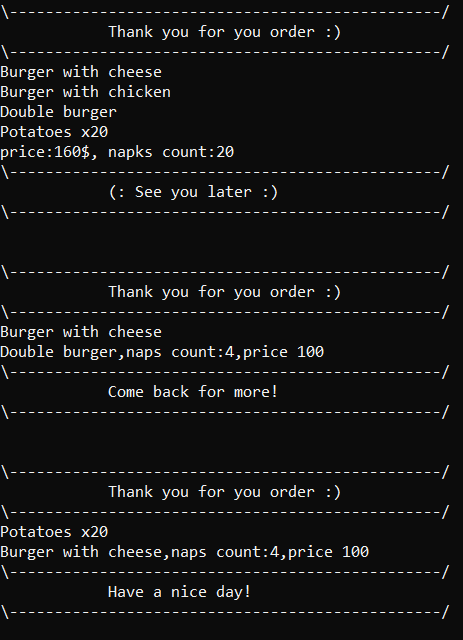


Рис.4.3. Реалізація завдання.

**Завдання 4:** Міст.

Створення класу Shape

Лістинг коду:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PatternsPartTwo.Bridge

{

internal abstract class Shape

{

public abstract string Name { get; }

public abstract void Draw();

}

}

Створення дочірніх класів

namespace PatternsPartTwo.Bridge.Figures

{

internal class Circle:Shape

{

public override string Name => "Circle";

public override void Draw()

{

Console.Write($"Drawing {Name}");

}

}

}

namespace PatternsPartTwo.Bridge.Figures

{

internal class Square:Shape

{

public override string Name => "Square";

public override void Draw()

{

Console.Write($"Drawing {Name}");

}

}

}

namespace PatternsPartTwo.Bridge.Figures

{

internal class Triangle : Shape

{

public override string Name => "Triangle";

public override void Draw()

{

Console.Write($"Drawing {Name}");

}

}

}

Результат виконання програми:

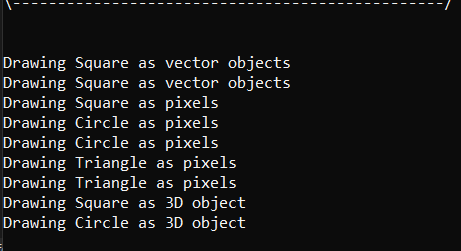


Рис.4.4. Реалізація завдання.

**Висновок:** на лабораторній роботі було отримано навички з реалізовування структурних шаблонів проєктування Адаптер, Декоратор, Фасад, Міст.