

Лабораторна робота №6

Тема: Поведінкові шаблони

Мета роботи: навчитися реалізовувати структурні шаблони проєктування

Ланцюжок відповідальностей, Посередник, Спостерігач, Стратегія

Завдання на лабораторну роботу

Завдання 1: Ланцюжок відповідальностей.

1. Створіть систему підтримки користувачів.
2. Вона має функціонувати так, як покрокове меню (наприклад, коли телефонуєте до оператора мобільного зв'язку). Але замість голосових повідомлень Ви маєте виводити відповідні повідомлення в консоль і чекати на вибір користувача.
3. Система повинна мати мінімум 4 рівні. Всі питання мають на меті обрати правильний рівень підтримки (тобто Handler) для користувача. Тобто вже на першому питанні може бути підібрано правильний рівень підтримки, тоді меню закінчується. А може бути так, що на жодному питанні не буде знайдено правильний рівень (Handler), тоді меню має повторитися.
4. Покажіть правильність роботи свого коду запустивши його в головному методі програми.

Лістинг програми:

```
abstract class SupportHandler
{
    protected SupportHandler NextHandler;

    public void SetNextHandler(SupportHandler nextHandler)
    {
        this.NextHandler = nextHandler;
    }

    public virtual void HandleRequest(string request)
    {
        if (this.NextHandler != null)
        {
            this.NextHandler.HandleRequest(request);
        }
    }
}

class TechnicalSupportHandler : SupportHandler
{
}
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.6						
Змн.	Арк.	№ докум.	Підпис	Дата							
Розроб.		Ігліньський І.Ю.			Звіт з лабораторної роботи №6			Лім.		Арк.	Аркуші
Перевір.		Левківський В.Л.								1	13
Реценз.								ФІКТ, гр. ІПЗ-21-4			
Н. Контр.											
Зав.каф.											

```

public override void HandleRequest(string request)
{
    if (request == "technical")
    {
        Console.WriteLine("Technical support team will help you.");
    }
    else
    {
        base.HandleRequest(request);
    }
}
}

class BillingSupportHandler : SupportHandler
{
    public override void HandleRequest(string request)
    {
        if (request == "billing")
        {
            Console.WriteLine("Billing support team will help you.");
        }
        else
        {
            base.HandleRequest(request);
        }
    }
}

class GeneralSupportHandler : SupportHandler
{
    public override void HandleRequest(string request)
    {
        if (request == "general")
        {
            Console.WriteLine("General support team will help you.");
        }
        else
        {
            base.HandleRequest(request);
        }
    }
}

class NoSupportHandler : SupportHandler
{
    public override void HandleRequest(string request)
    {
        Console.WriteLine("Sorry, we cannot help you with that.");
    }
}

class Program
{
    static void Main(string[] args)
    {
        var technicalHandler = new TechnicalSupportHandler();
        var billingHandler = new BillingSupportHandler();
        var generalHandler = new GeneralSupportHandler();
        var noSupportHandler = new NoSupportHandler();
        technicalHandler.SetNextHandler(billingHandler);
        billingHandler.SetNextHandler(generalHandler);
    }
}

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						2
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        generalHandler.SetNextHandler(noSupportHandler);

        while (true)
        {
            Console.WriteLine("Which support do you need?  
(technical/billing/general)");
            var request = Console.ReadLine().ToLower();

            technicalHandler.HandleRequest(request);
            Console.WriteLine();
        }
    }
}

```

```

Which support do you need? (technical/billing/general)
technical
Technical support team will help you.

Which support do you need? (technical/billing/general)
billing
Billing support team will help you.

Which support do you need? (technical/billing/general)
general
General support team will help you.

Which support do you need? (technical/billing/general)
exit
Sorry, we cannot help you with that.

Which support do you need? (technical/billing/general)

```

Результат запуску

Завдання 2: Посередник.

1. Відрефакторте [код](#) продемонстрований на лекції за допомогою використання шаблону Посередник.
2. В результаті рефакторингу Aircraft не повинен “знати” про Runway і навпаки. Обидві сутності повинні “знати” лише про CommandCentre.
3. Покажіть правильність роботи свого коду запустивши його в головному методі програми.

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						3
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг Aircraft:

```
using DesignPatterns.Mediator;

namespace Poserednik_Task2
{
    class Aircraft
    {
        public int AircraftNumber { get; private set; }
        public bool IsLanded { get; set; }
        public bool IsLandingAllowed { get; set; }

        public Aircraft(int number)
        {
            AircraftNumber = number;
            IsLanded = false;
            IsLandingAllowed = false;
        }

        public void Land(Runway runway, CommandCentre commandCentre)
        {
            commandCentre.RequestLanding(this);
            if (IsLanded)
            {
                Console.WriteLine("Aircraft #" + AircraftNumber + " is landing on runway #" + runway.RunwayNumber);
            }
        }

        public void LeaveRunway(Runway runway, CommandCentre commandCentre)
        {
            commandCentre.FreeRunway(runway, this);
        }
    }
}
```

Лістинг ComandCenter:

```
using Poserednik_Task2;

namespace DesignPatterns.Mediator
{
    class CommandCentre
    {
        public void RequestLanding(Aircraft aircraft)
        {
            if (aircraft.IsLandingAllowed)
            {
                Console.WriteLine("Aircraft #" + aircraft.AircraftNumber + " has been landed successfully!");
                aircraft.IsLanded = true;
                aircraft.IsLandingAllowed = false;
            }
            else
            {
                Console.WriteLine("Aircraft #" + aircraft.AircraftNumber + " cannot land at this moment!");
            }
        }

        public void AllowLanding(Runway runway, Aircraft aircraft)
        {
            if (runway.IsAvailable)
            {
            }
        }
    }
}
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						4
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        {
            Console.WriteLine("Runway #" + runway.RunwayNumber + " is now
available for landing of Aircraft #" + aircraft.AircraftNumber);
            runway.IsAvailable = false;
            aircraft.IsLandingAllowed = true;
        }
        else
        {
            Console.WriteLine("Runway #" + runway.RunwayNumber + " is busy at
the moment! Aircraft #" + aircraft.AircraftNumber + " cannot land now!");
        }
    }

    public void FreeRunway(Runway runway, Aircraft aircraft)
    {
        Console.WriteLine("Aircraft #" + aircraft.AircraftNumber + " has left
the runway #" + runway.RunwayNumber);
        runway.IsAvailable = true;
        aircraft.IsLanded = false;
    }
}
}

```

Лістинг Program.cs:

```

using DesignPatterns.Mediator;
using Poserednik_Task2;

class Program
{
    static void Main(string[] args)
    {
        CommandCentre commandCentre = new CommandCentre();
        Runway runway1 = new Runway(1);
        Runway runway2 = new Runway(2);
        Aircraft aircraft1 = new Aircraft(1);
        Aircraft aircraft2 = new Aircraft(2);

        aircraft1.Land(runway1, commandCentre);
        aircraft2.Land(runway2, commandCentre);
        aircraft1.LeaveRunway(runway1, commandCentre);
        aircraft2.Land(runway1, commandCentre);
        aircraft1.Land(runway2, commandCentre);
        aircraft2.LeaveRunway(runway1, commandCentre);
        aircraft1.LeaveRunway(runway2, commandCentre);
    }
}

```

```

Aircraft #1 cannot land at this moment!
Aircraft #2 cannot land at this moment!
Aircraft #1 has left the runway #1
Aircraft #2 cannot land at this moment!
Aircraft #1 cannot land at this moment!
Aircraft #2 has left the runway #1
Aircraft #1 has left the runway #2

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						5
Змн.	Арк.	№ докум.	Підпис	Дата		

Завдання 3: Спостерігач.

1. Додайте до гри [Blackjack](#) можливість слідкувати за певними подіями в грі.
2. Створіть сервіс, який буде моніторити випадки перебору очків (більше 21) і писати відповідні випадки в окремий файл.
3. Створіть сервіс, який буде збирати аналітику про середньо набрану кількість очок і писати в окремий файл.
4. Для виконання цього завдання створіть PR (він має бути без конфліктів) і відправте посилання на нього.

Лістинг класу IObservable.cs:

```
using Blackjack;

namespace blackjack.Game;

public interface IObservable
{
    void AddObserver(IObserver observer);
    void RemoveObserver(IObserver observer);
    void NotifyObservers(string message);
}
```

Лістинг класу IObserver.cs:

```
namespace Blackjack;

public interface IObserver
{
    void Update(string message);
}
```

Лістинг класу EventLogger.cs:

```
using Blackjack;

namespace ClassLibrary.Blackjack.Observer
{
    public class EventLogger : IObserver
    {
        private string filePath;
        public EventLogger(string filePath)
        {
            this.filePath = filePath;
        }
        public void Update(string message)
        {
            using (var writer = new StreamWriter(filePath, true))
            {
                writer.WriteLine(message);
            }
        }
    }
}
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						6
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг класу GameObservable.cs:

```
using BlackJack;
using blackjack.Game;

namespace ClassLibrary.Blackjack.Observer
{
    public class GameObservable : IObservable
    {
        private List<IObserver> observers = new List<IObserver>();
        public void AddObserver(IObserver observer)
        {
            this.observers.Add(observer);
        }
        public void RemoveObserver(IObserver observer)
        {
            this.observers.Remove(observer);
        }
        public void NotifyObservers(string message)
        {
            foreach (var observer in this.observers)
            {
                observer.Update(message);
            }
        }
    }
}
```

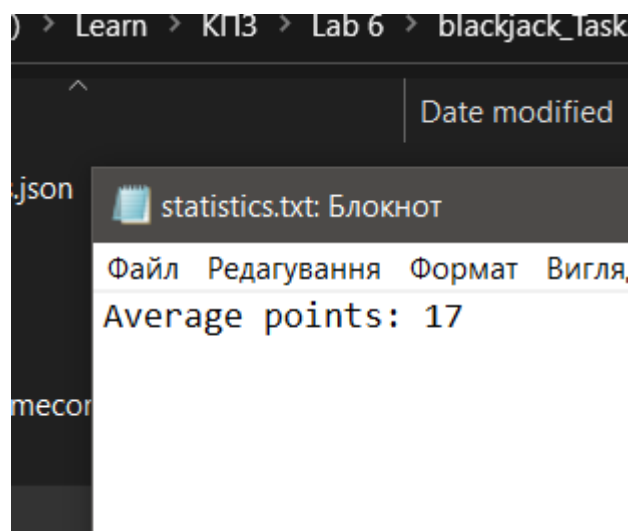
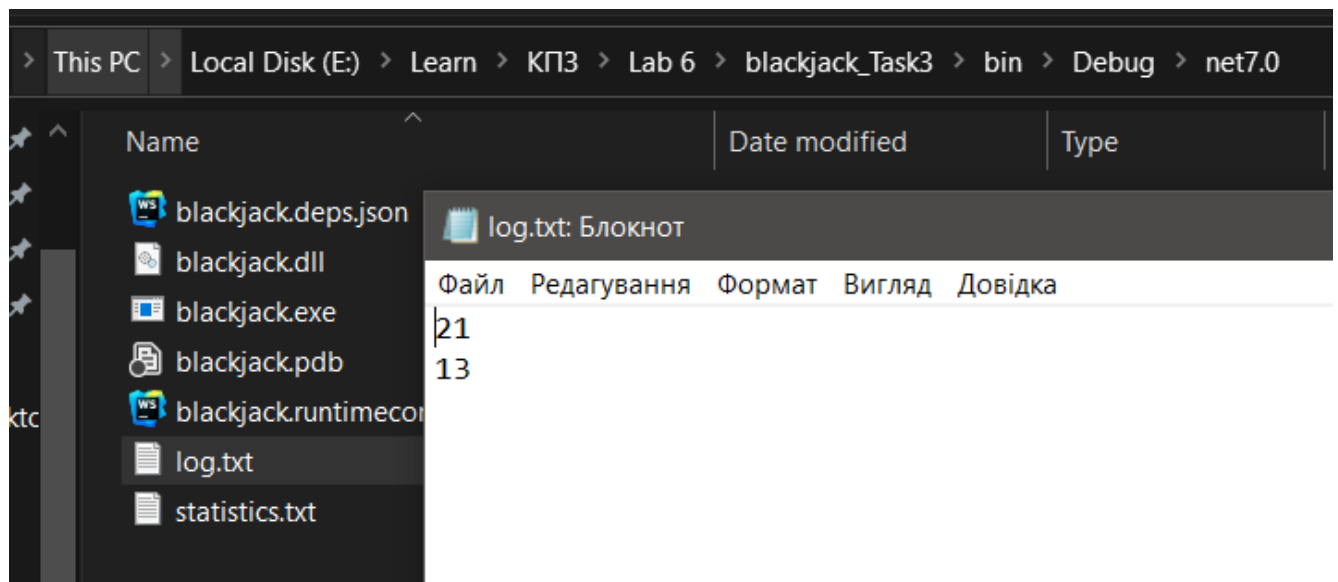
Лістинг класу PointsStatistics.cs:

```
using BlackJack;

namespace ClassLibrary.Blackjack.Observer
{
    public class PointsStatistics : IObserver
    {
        private string filePath;
        private int totalPointsCount;
        private int gamesPlayedCount;
        public PointsStatistics(string filePath)
        {
            this.filePath = filePath;
            totalPointsCount = 0;
            gamesPlayedCount = 0;
        }
        public void Update(string message)
        {
            totalPointsCount += int.Parse(message);
            gamesPlayedCount++;
            using (var writer = new StreamWriter(filePath))
            {
                writer.WriteLine($"Average points: {totalPointsCount /
(double) gamesPlayedCount}");
            }
        }
    }
}
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						7
Змн.	Арк.	№ докум.	Підпис	Дата		

Результат виконання програми:



Завдання 4: Стратегія.

1. Додайте до гри [Blackjack](#) комп'ютерного персонажа.
2. За допомогою патерну Стратегія реалізуйте динамічний вибір опонента.
3. За допомогою патерну Стратегія реалізуйте різні алгоритми гри комп'ютерного персонажу: обережний (буде зупинятися після 13 набраних очок), ризиковий (після 19), рандомний (за Вашим власним алгоритмом).
4. Для виконання цього завдання створіть PR (він має бути без конфліктів) і відправте посилання на нього.

Лістинг класу IStrategy.cs:

```
using BlackJack;
using ClassLibrary.Blackjack;

namespace blackjack.Game;

public interface IStrategy
{
    void Play(Player player, GameState state);
}
```

Лістинг класу CautiousStrategy.cs:

```
using BlackJack;
using ClassLibrary.Blackjack;

namespace blackjack.Game;

public class CautiousStrategy : IStrategy
{
    public void Play(Player player, GameState state)
    {
        int playerPoints = PointsCounter.CountSum(player.DrawnCards);
        while (playerPoints < 13 && player.ConfirmNextDraw())
        {
            player.DrawCard(state.Deck);
            playerPoints = PointsCounter.CountSum(player.DrawnCards);
        }
    }
}
```

Лістинг класу RandomStrategy.cs:

```
using BlackJack;
using blackjack.Game;

namespace ClassLibrary.Blackjack.Strategy
{
    public class RandomStrategy : IStrategy
    {
        public void Play(Player player, GameState state)
        {
            Random random = new Random();
            int playerPoints = PointsCounter.CountSum(player.DrawnCards);
            while (playerPoints < 21 && player.ConfirmNextDraw())
            {
                if (playerPoints > 17 && random.NextDouble() < 0.5)
                    break;
                player.DrawCard(state.Deck);
                playerPoints = PointsCounter.CountSum(player.DrawnCards);
            }
        }
    }
}
```

Лістинг класу RiskyStrategy.cs:

```
using BlackJack;
using blackjack.Game;

namespace ClassLibrary.Blackjack.Strategy
{
    public class RiskyStrategy : IStrategy
    {
        public void Play(Player player, GameState state)
        {
            int playerPoints = PointsCounter.CountSum(player.DrawnCards);
            while (playerPoints < 21 && player.ConfirmNextDraw())
            {
                player.DrawCard(state.Deck);
                playerPoints = PointsCounter.CountSum(player.DrawnCards);
            }
        }
    }
}
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

```
public class RiskyStrategy : IStrategy
{
    public void Play(Player player, GameState state)
    {
        int playerPoints = PointsCounter.CountSum(player.DrawnCards);
        while (playerPoints < 19 && player.ConfirmNextDraw())
        {
            player.DrawCard(state.Deck);
            playerPoints = PointsCounter.CountSum(player.DrawnCards);
        }
    }
}
```

Лістинг класу Player.cs:

```
using BlackJack;
using blackjack.Game;

using ClassLibrary.Blackjack.Strategy;
namespace ClassLibrary.Blackjack
{
    public class Player
    {
        public string Name { get; }
        public List<Card> DrawnCards { get; } = new List<Card>();
        public IStrategy Strategy { get; set; }
        public Player(string name, IStrategy strategy)
        {
            this.Name = name;
            Strategy = strategy;
        }
        public bool ConfirmNextDraw()
        {
            return InputHandler.Confirm("Do you want to draw next card?");
        }
        public Card DrawCard(CardsDeck cardsDeck)
        {
            Card card = cardsDeck.Draw();
            this.DrawnCards.Add(card);
            Logger.ShowDrawnCard(card, PointsCounter.CountSum(this.DrawnCards));
            return card;
        }
        public void Play(GameState state)
        {
            this.Strategy.Play(this, state);
        }
    }
}
```

Лістинг класу Game.cs:

```
using BlackJack;
using blackjack.Game;
using ClassLibrary.Blackjack.Observer;
using ClassLibrary.Blackjack.Strategy;
namespace ClassLibrary.Blackjack
{
    public class Game : GameObservable
    {

```

```

    public static readonly int PLAYER_COUNT = 2;
    public static readonly int CARDS_WITHOUT_CONFIRMATION_COUNT = 2;
    private GameState _state = new GameState();
    public Game()
    {
        AddObserver(new EventLogger("log.txt"));
        AddObserver(new PointsStatistics("statistics.txt"));
    }
    private List<Player> _createPlayers()
    {
        var players = new List<Player>();
        for (int i = 1; i <= PLAYER_COUNT; i++)
        {
            string defaultName = $"Player {i}";
            string name = InputHandler.RequestAnswer($"Write a name for [{defaultName}]");
            ;
            Player player = new Player(name, ChooseStrategy());
            players.Add(player);
        }
        return players;
    }
    private void _greet()
    {
        Logger.Greet();
    }
    private void _initiateState()
    {
        this._state.SetPlayers(this._createPlayers());
    }
    public void Start()
    {
        this._greet();
        this._initiateState();
        do
        {
            {
                if (this._state.ActivePlayer == null)
                {
                    throw new Exception("No active player detected!");
                }
                this.HandlePlayer(this._state.ActivePlayer);
            }
            while (this._state.SwitchPlayer());
            this.End();
        }
        public void End()
        {
            List<Player> winners = this._state.GetWinners();
            Logger.EndGame(winners);
        }
        public void HandlePlayer(Player player)
        {
            Logger.StartPlayersTurn(player.Name);
            for (int i = 0; i < CARDS_WITHOUT_CONFIRMATION_COUNT; i++)
            {
                player.DrawCard(this._state.Deck);
            }
            player.Play(this._state);
            NotifyObservers(PointsCounter.CountSum(player.DrawnCards).ToString());
        }
        public IStrategy ChooseStrategy()
        {
            int strategyChoice = int.Parse(InputHandler.RequestAnswer("Choose a strategy: 1 - Cautious, 2 - Risky, 3 - Random, any - Random"));
            switch (strategyChoice)

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		11

```

{
case 1:
return new CautiousStrategy();
case 2:
return new RiskyStrategy();
case 3:
return new RandomStrategy();
default:
return new RandomStrategy();
}
}
}
}
}

```

```

"E:/Learn/КПЗ/Lab 6/blackjack_Task3/bin/Debug/net7.0/blackjack.exe"
=====
Hello, let's play BlackJack
=====
Write a name for [Player 1]
Player 1
Choose a strategy: 1 - Cautious, 2 - Risky, 3 - Random, any - Random
3
Write a name for [Player 2]

Choose a strategy: 1 - Cautious, 2 - Risky, 3 - Random, any - Random
1
Now it's Player 1's turn!
Get ready







You have drawn Six of Clubs
Your total points count is: 6


You have drawn Seven of Clubs
Your total points count is: 13

Do you want to draw next card?
Please type y/N:
y
y

You have drawn Two of Hearts
Your total points count is: 19

```

 Git
  Run
  Problems
  NuGet
  Terminal
  Build

 Build succeeded at 6:48:47 PM (a minute ago)

Демонстрація виконаного завдання

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						12
Змн.	Арк.	№ докум.	Підпис	Дата		

Висновок: у ході виконання лабораторної роботи ми навчитися реалізовувати структурні шоблони проєктування Ланцюжок відповідальностей, Посередник, Спостерігач, Стратегія.

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.23.121.14.000 – Лр.4	Арк.
						13
Змн.	Арк.	№ докум.	Підпис	Дата		