**Command:**

public interface ICommand

{

void Execute();

void Undo();

}

public class LightOnCommand : ICommand

{

private Light \_light;

public LightOnCommand(Light light)

{

\_light = light;

}

public void Execute()

{

\_light.On();

}

public void Undo()

{

\_light.Off();

}

}

public class LightOffCommand : ICommand

{

private Light \_light;

public LightOffCommand(Light light)

{

\_light = light;

}

public void Execute()

{

\_light.Off();

}

public void Undo()

{

\_light.On();

}

}

public class Light

{

public void On()

{

Console.WriteLine("Свет включен.");

}

public void Off()

{

Console.WriteLine("Свет выключен.");

}

}

public class Television

{

public void On()

{

Console.WriteLine("Телевизор включен.");

}

public void Off()

{

Console.WriteLine("Телевизор выключен.");

}

}

public class TelevisionOnCommand : ICommand

{

private Television \_tv;

public TelevisionOnCommand(Television tv)

{

\_tv = tv;

}

public void Execute()

{

\_tv.On();

}

public void Undo()

{

\_tv.Off();

}

}

public class TelevisionOffCommand : ICommand

{

private Television \_tv;

public TelevisionOffCommand(Television tv)

{

\_tv = tv;

}

public void Execute()

{

\_tv.Off();

}

public void Undo()

{

\_tv.On();

}

}

public class RemoteControl

{

private ICommand \_onCommand;

private ICommand \_offCommand;

public void SetCommands(ICommand onCommand, ICommand offCommand)

{

\_onCommand = onCommand;

\_offCommand = offCommand;

}

public void PressOnButton()

{

\_onCommand.Execute();

}

public void PressOffButton()

{

\_offCommand.Execute();

}

public void PressUndoButton()

{

\_onCommand.Undo();

}

}

TemplateMethod:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab7.Task2

{

public abstract class Beverage

{

public void PrepareRecipe()

{

BoilWater();

Brew();

PourInCup();

AddCondiments();

}

private void BoilWater()

{

Console.WriteLine("Кипячение воды...");

}

private void PourInCup()

{

Console.WriteLine("Наливание в чашку...");

}

protected abstract void Brew();

protected abstract void AddCondiments();

}

public class Tea : Beverage

{

protected override void Brew()

{

Console.WriteLine("Заваривание чая...");

}

protected override void AddCondiments()

{

Console.WriteLine("Добавление лимона...");

}

}

public class Coffee : Beverage

{

protected override void Brew()

{

Console.WriteLine("Заваривание кофе...");

}

protected override void AddCondiments()

{

Console.WriteLine("Добавление сахара и молока...");

}

}

}

**Mediator:**

public interface IMediator

{

void SendMessage(string message, Colleague colleague);

}

public abstract class Colleague

{

protected IMediator \_mediator;

public Colleague(IMediator mediator)

{

\_mediator = mediator;

}

public abstract void ReceiveMessage(string message);

}

public class ChatMediator : IMediator

{

private List<Colleague> \_colleagues;

public ChatMediator()

{

\_colleagues = new List<Colleague>();

}

public void RegisterColleague(Colleague colleague)

{

\_colleagues.Add(colleague);

}

public void SendMessage(string message, Colleague sender)

{

foreach (var colleague in \_colleagues)

{

if (colleague != sender)

{

colleague.ReceiveMessage(message);

}

}

}

}

public class User : Colleague

{

private string \_name;

public User(IMediator mediator, string name) : base(mediator)

{

\_name = name;

}

public void Send(string message)

{

Console.WriteLine($"{\_name} отправляет сообщение: {message}");

\_mediator.SendMessage(message, this);

}

public override void ReceiveMessage(string message)

{

Console.WriteLine($"{\_name} получил сообщение: {message}");

}

}

**Main:**

static void Main(string[] args)

{

Light livingRoomLight = new Light();

Television tv = new Television();

ICommand lightOn = new LightOnCommand(livingRoomLight);

ICommand lightOff = new LightOffCommand(livingRoomLight);

ICommand tvOn = new TelevisionOnCommand(tv);

ICommand tvOff = new TelevisionOffCommand(tv);

RemoteControl remote = new RemoteControl();

remote.SetCommands(lightOn, lightOff);

Console.WriteLine("Управление светом:");

remote.PressOnButton();

remote.PressOffButton();

remote.PressUndoButton();

remote.SetCommands(tvOn, tvOff);

Console.WriteLine("\nУправление телевизором:");

remote.PressOnButton();

remote.PressOffButton();

//Task2

Beverage tea = new Tea();

Console.WriteLine("Приготовление чая:");

tea.PrepareRecipe();

Console.WriteLine();

Beverage coffee = new Coffee();

Console.WriteLine("Приготовление кофе:");

coffee.PrepareRecipe();

//Task3

Console.WriteLine("");

ChatMediator chatMediator = new ChatMediator();

User user1 = new User(chatMediator, "Алиса");

User user2 = new User(chatMediator, "Боб");

User user3 = new User(chatMediator, "Чарли");

chatMediator.RegisterColleague(user1);

chatMediator.RegisterColleague(user2);

chatMediator.RegisterColleague(user3);

user1.Send("Привет всем!");

user2.Send("Привет ребята!");

user3.Send("Всем привет!");

}