МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Кафедра «Вычислительная техника»

Дисциплина «Технология программирования»

**Лабораторная работа №3.**

**Паттерн Стратегия (Strategy)**

Выполнил:

студент группы ИВТАПбд-21

Кондратьев П. С.

Проверил:

Фролов В. А.

Ульяновск, 2018

1. Паттерн Стратегия определяет семейство алгоритмов, инкапсулирует каждый из них и обеспечивает взаимозаменяемость
2. Принцип проектирования: отделить постоянные части программы от изменяемых
3. Принцип проектирования: отдавать предпочтение композиции нежели наследованию
4. Принцип программирования: программируйте на уровне интерфейса, а не реализации

**Техническое задание**

* Реализовать Паттерн Стратегия и показать его наглядное применение

Исходный код

class Program

{

static void Main(string[] args)

{

List<DuckBase> ducks = new List<DuckBase>();

ducks.Add(new ExoticDuck());

ducks.Add(new SimpleDuck());

ducks.Add(new WoodenDuck());

ducks.Add(new RubberDuck());

foreach (var duck in ducks)

{

duck.Display();

duck.Swim();

duck.Quack();

duck.Fly();

Console.WriteLine();

}

DuckBase upgradableDuck = new UpgradableDuck();

upgradableDuck.Display();

upgradableDuck.Swim();

upgradableDuck.Quack();

upgradableDuck.Fly();

upgradableDuck.SetFlyBehaviour(new FlyWithWings());

upgradableDuck.SetQuackBehaviour(new ExoticQuack());

upgradableDuck.Quack();

upgradableDuck.Fly();

Console.ReadKey();

}

} public abstract class DuckBase

{

protected IFlyable flyBehaviour;

protected IQuackable quackBehaviour;

public DuckBase()

{

flyBehaviour = new FlyWithWings();

quackBehaviour = new SimpleQuack();

}

public void SetQuackBehaviour(IQuackable newQuackBehaviour)

{

quackBehaviour = newQuackBehaviour;

}

public void SetFlyBehaviour(IFlyable newFlyBehaviour)

{

flyBehaviour = newFlyBehaviour;

}

public void Swim()

{

Console.WriteLine("I'm swimming");

}

public void Quack()

{

quackBehaviour.Quack();

}

public void Fly()

{

flyBehaviour.Fly();

}

public abstract void Display();

} public class UpgradableDuck : DuckBase

{

public UpgradableDuck()

{

flyBehaviour = new NoFly();

quackBehaviour = new NoQuack();

}

public override void Display()

{

Console.WriteLine("I'm an upgradable duck!");

}

} public class WoodenDuck : DuckBase

{

public WoodenDuck()

{

flyBehaviour = new NoFly();

quackBehaviour = new NoQuack();

}

public override void Display()

{

Console.WriteLine("Hi! I'm a wooden duck!");

}

}