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 №5.**

**Заместитель (Прокси)**

Выполнил:

студент группы ИВТАПбд-21

Кондратьев П. С.

Проверил:

Фролов В. А.

Ульяновск, 2018

Паттерн Заместитель (Proxy) предоставляет объект-заместитель, который управляет доступом к другому объекту. То есть создается объект-суррогат, который может выступать в роли другого объекта и замещать его.

**Когда использовать прокси?**

* Когда надо осуществлять взаимодействие по сети, а объект-проси должен имитировать поведения объекта в другом адресном пространстве. Использование прокси позволяет снизить накладные издержки при передачи данных через сеть. Подобная ситуация еще называется удалённый заместитель (remote proxies)
* Когда нужно управлять доступом к ресурсу, создание которого требует больших затрат. Реальный объект создается только тогда, когда он действительно может понадобится, а до этого все запросы к нему обрабатывает прокси-объект. Подобная ситуация еще называется виртуальный заместитель (virtual proxies)
* Когда необходимо разграничить доступ к вызываемому объекту в зависимости от прав вызывающего объекта. Подобная ситуация еще называется защищающий заместитель (protection proxies)
* Когда нужно вести подсчет ссылок на объект или обеспечить потокобезопасную работу с реальным объектом. Подобная ситуация называется "умные ссылки" (smart reference)

**Техническое задание**

Реализовать паттерн и продемонстрировать его работу с различными типами объектов (показать участников паттерна).

Исходный код

/// The 'Subject interface

public interface IClient

{string GetData();}

/// The 'RealSubject' class

public class RealClient : IClient

{

string Data;

public RealClient()

{Console.WriteLine("Real Client: Initialized");Data = "Dot Net Tricks";}

public string GetData()

{return Data; }

}

/// The 'Proxy Object' class

public class ProxyClient : IClient

{

RealClient client = new RealClient();

public ProxyClient()

{ Console.WriteLine("ProxyClient: Initialized");}

public string GetData()

{return client.GetData();}

/// Proxy Pattern Demo

class Program

{

static void Main(string[] args)

{

ProxyClient proxy = new ProxyClient();

Console.WriteLine("Data from Proxy Client = {0}", proxy.GetData());

Console.ReadKey();

}

}