# Практическое занятие № 19. Реализация в классах свойств и индексаторов

**1 Цель занятия**

Получить практические навыки реализации свойств и индексаторов в приложении Windows Forms

**2 Перечень оборудования и программного обеспечения**

Персональный компьютер

Microsoft Office (Word)

Microsoft Visual Studio

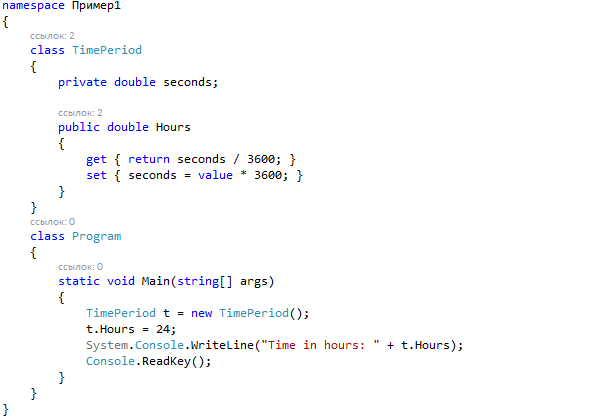
**3 Краткие теоретические сведения**

**3.1 Свойства**

***Свойство*** — это член класса, предоставляющий гибкий механизм для чтения, записи или вычисления значения частного (private) поля. Свойства можно использовать, как если бы они являлись открытыми членами данных, хотя в действительности они являются специальными методами, называемыми методами доступа. Это обеспечивает простой доступ к данным и позволяет повысить уровень безопасности и гибкости методов.

В примере 1 класс TimePeriod хранит сведения о периоде времени. Внутри класса время хранится в секундах, но свойство с именем Hours позволяет клиенту задать время в часах. Методы доступа для свойства Hours выполняют преобразование между часами и секундами.

**Пример 1:**



**3.2 Методы доступа свойств**

Свойства позволяют классу предоставлять общий способ получения и задания значений, скрывая при этом код реализации или проверки.

Метод доступа свойства get используется для *возврата значения свойства* и должен содержать ключевое слово return, а метод доступа set используется для *назначения нового значения*. Эти методы доступа могут иметь различные уровни доступа.

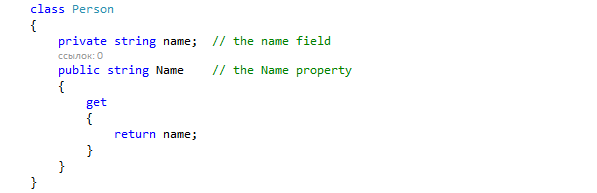
Ключевое слово value используется для определения значения, присваиваемого методом доступа set.

Свойства, которые не реализуют метод доступа set, доступны только для чтения.

**3.2.1 Метод доступа get**

Основная часть метода доступа get похожа на основную часть любого метода. Он должен возвращать значение, имеющее тип заданного свойства. Для этого метод доступа get должен заканчиваться оператором return.

**Пример 2:**



Вызывается метод доступа get для чтения значения свойства при создании на него ссылки:

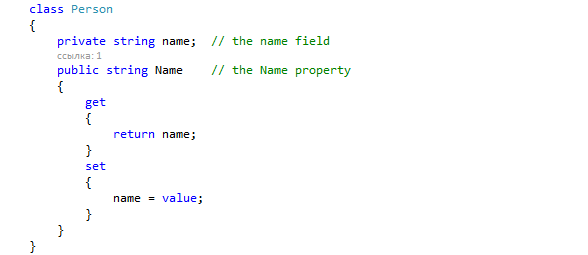


Если в предыдущем фрагменте кода свойству Name не назначается какое-либо значение, это свойство возвращает значение null.

**3.2.2 Метод доступа set**

Метод доступа set похож на метод, имеющий тип возвращаемого значения void. В нем используется неявный параметр value, тип которого соответствует типу свойства. В следующем примере метод доступа set добавляется в свойство Name.

**Пример 3:**



Когда свойству присваивается значение, выполняется вызов метода доступа set с помощью аргумента, предоставляющего новое значение:



Использование имени неявного параметра value для объявления локальной переменной в методе доступа set является ошибкой.

**3.3 Индексаторы**

Индексаторы в С# обеспечивают естественный синтаксис для доступа к элементам в классах или структурах, которые инкапсулируют список или словарь значений. Индексаторы являются подобными свойствам, но доступ к ним происходит с помощью аргумента индекса, а не имени свойства.

Класс string имеет индексатор, который дает вам возможность получить доступ к каждому из символов, который содержится в строке с помощью индекса типа int.

**Пример 4:**



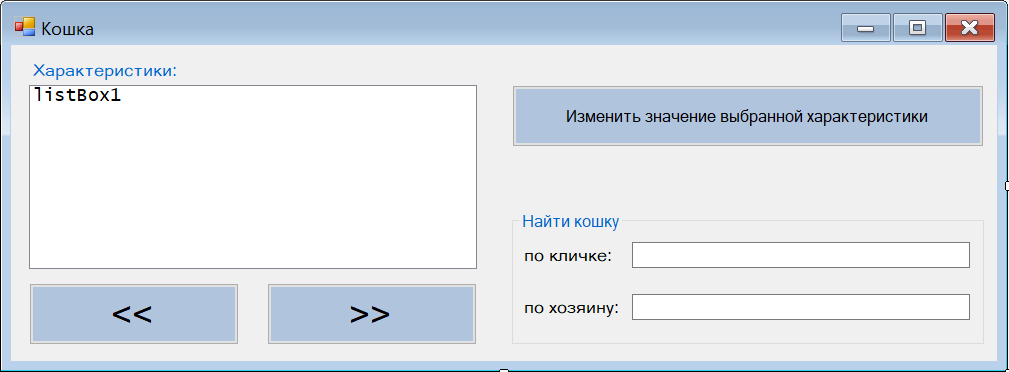
Синтаксис для использования индексатора такой же, как для работы с массивом.

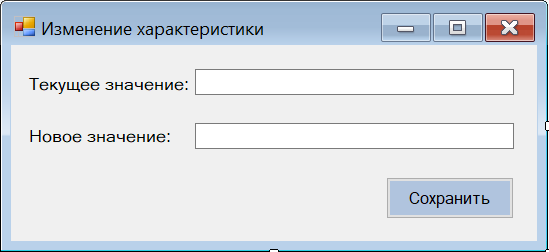
Индексаторы имеют те же модификаторы, которые доступны и для свойств.

**Задание.** Реализовать класс Cats с массивом элементов типа Cat для хранения информации о данных (параметрах) кошек, принимающих участие в выставке. В классе Cat описать массив характеристик кошек. На основе реализованных классов создать объекты. Разработать программу, осуществляющую доступ к элементам объектов с помощью числовых и строковых индексаторов.

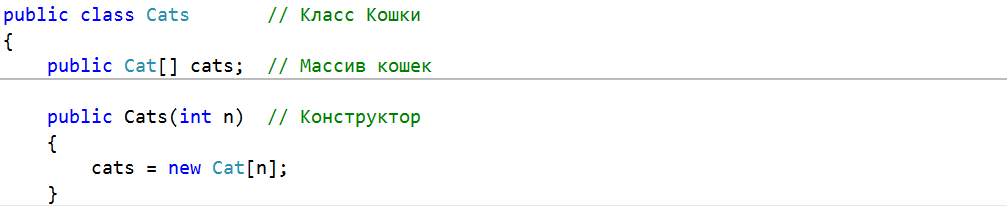
**Решение.**

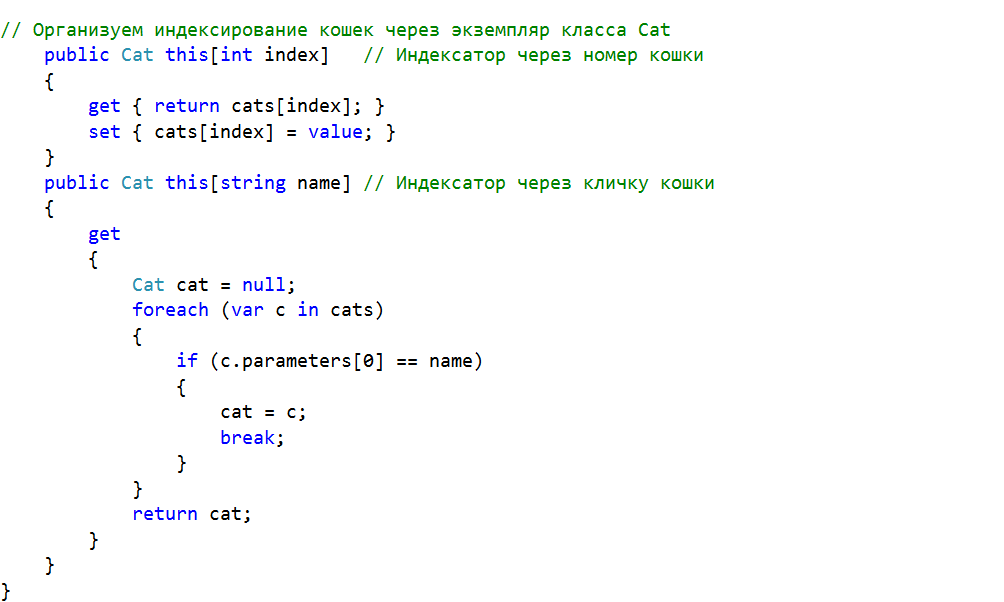
Формы Windows-приложения для работы с массивами в заданных классах:

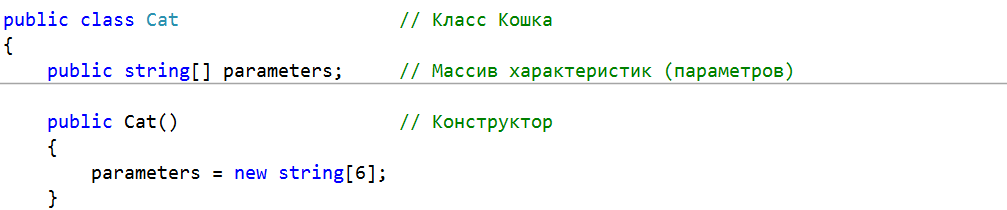


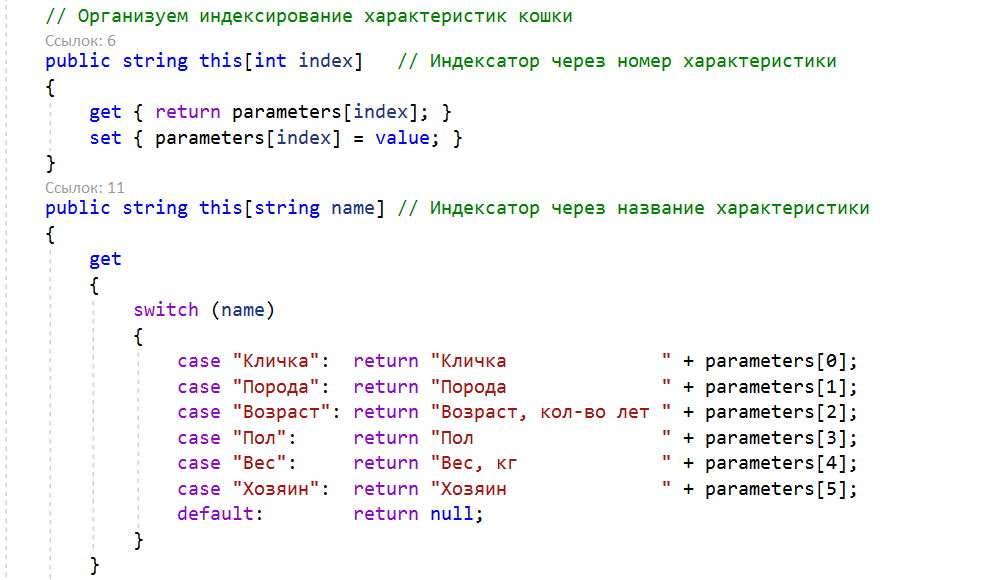


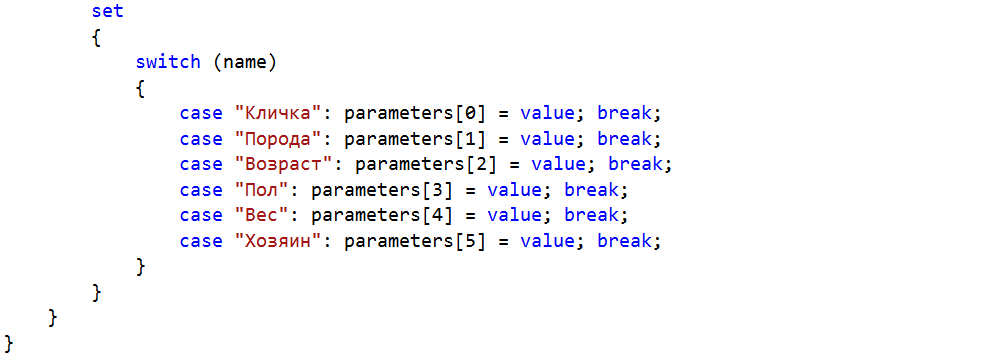
Код класса Cats с массивом элементов типа Cat и код класса Cat с массивом элементов типа string для хранения информации о характеристиках кошек:

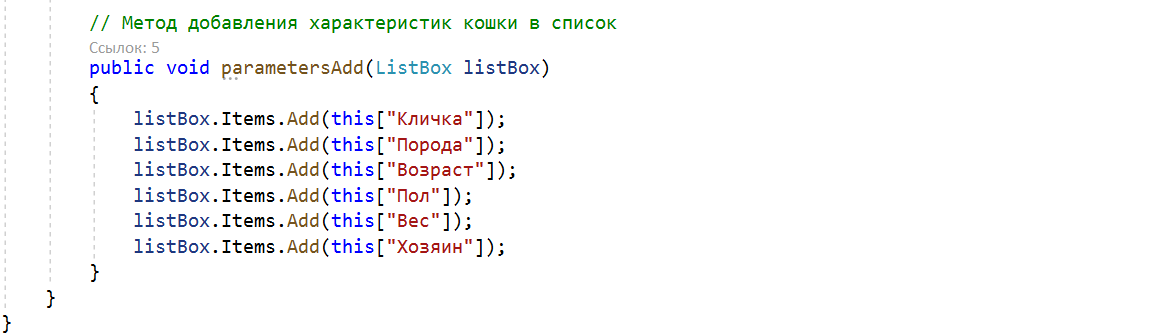




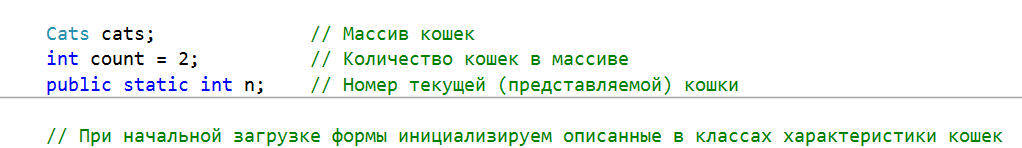


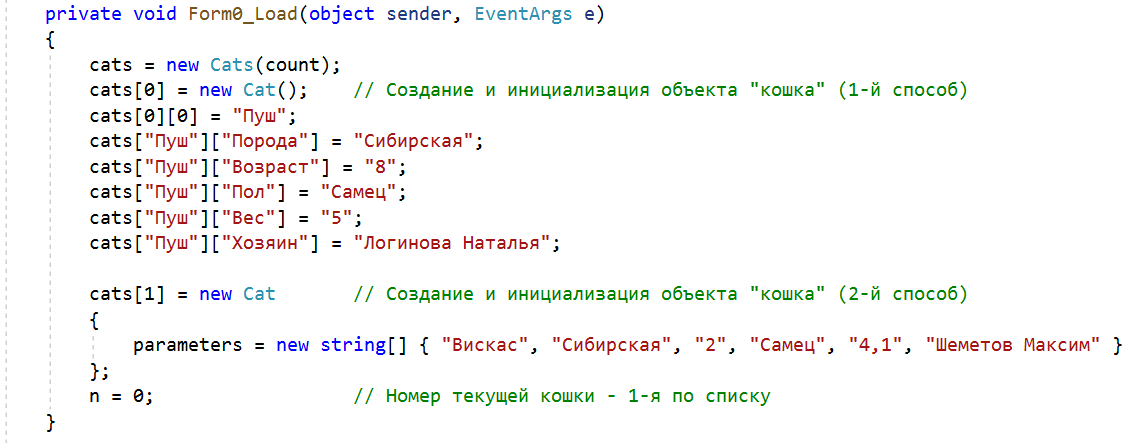


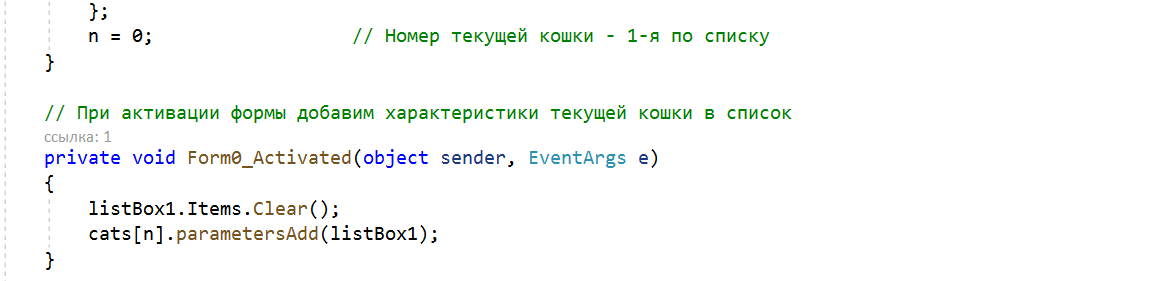


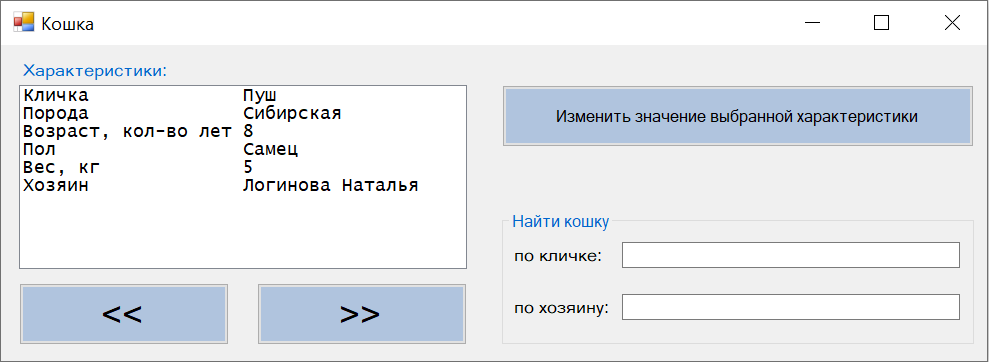


Коды методов класса Form1, инициализирующих данные (характеристики) созданных классов и отображающих их в списке:

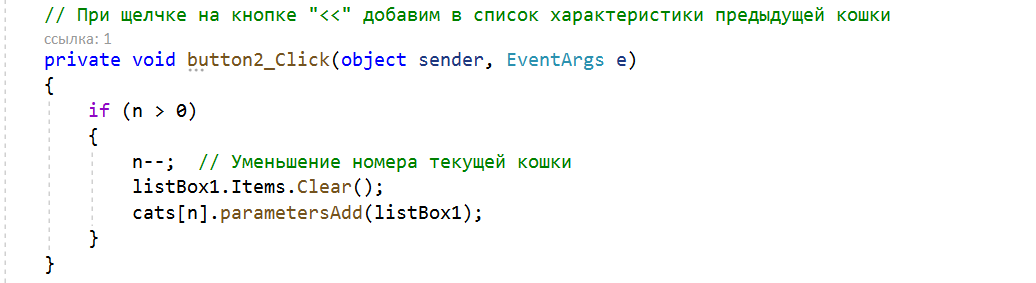


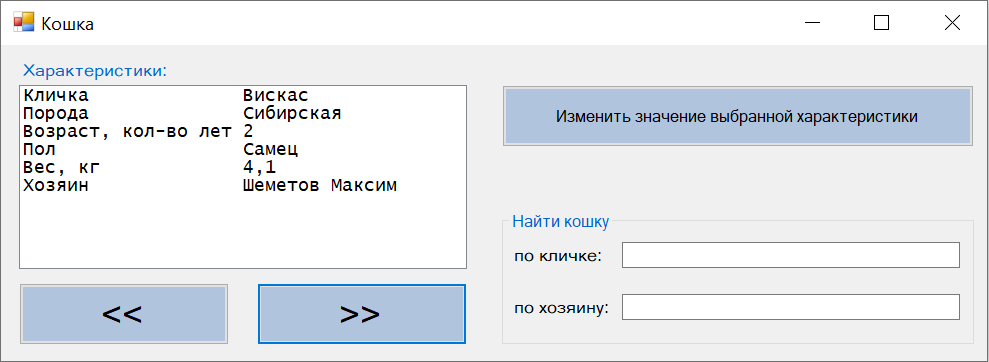


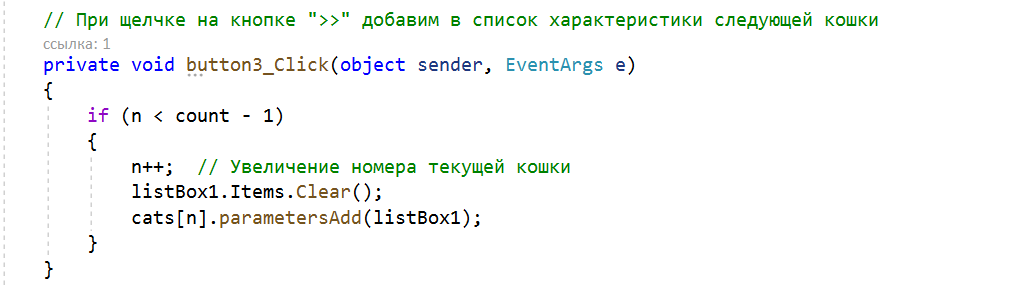


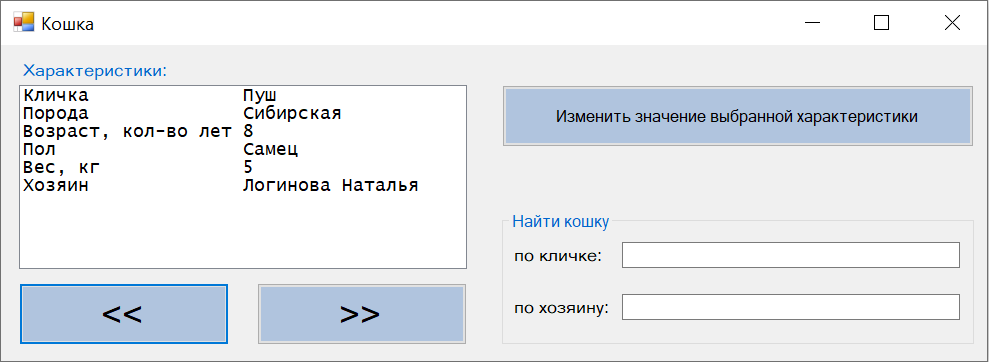


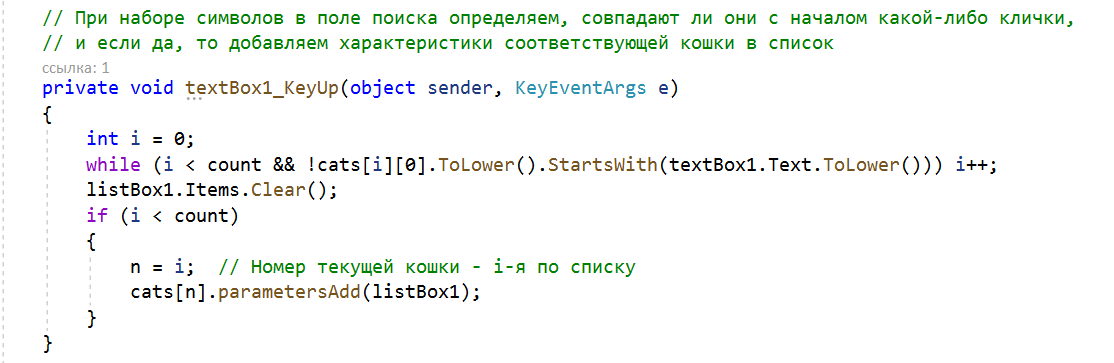
Методы класса Form1 для работы с массивами кошек и их характеристик:

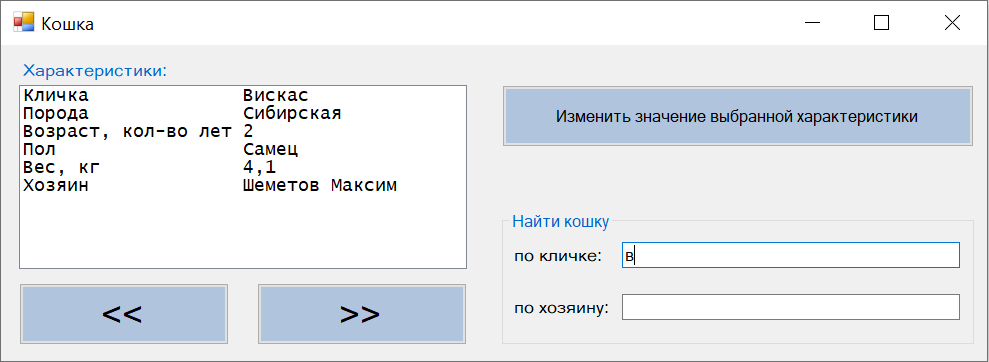


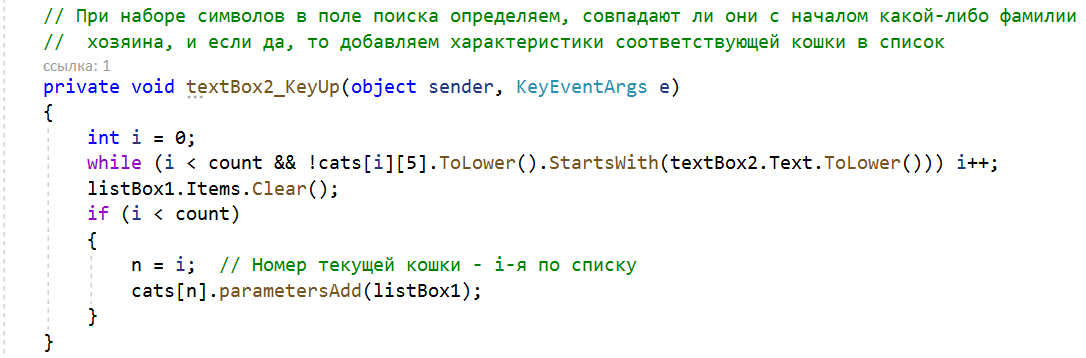


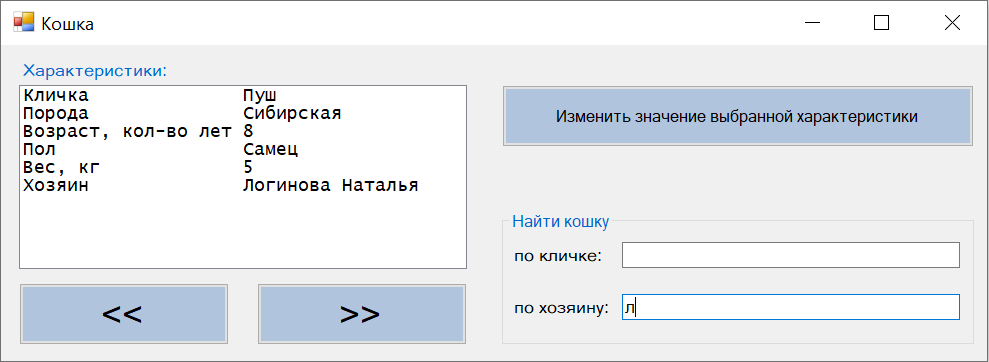


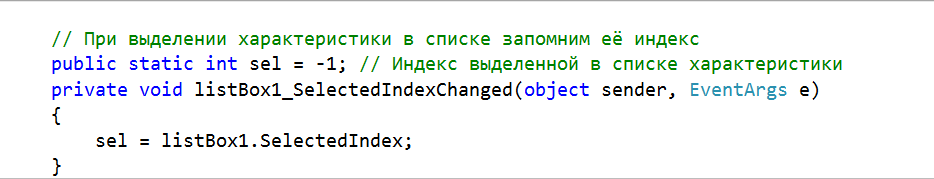


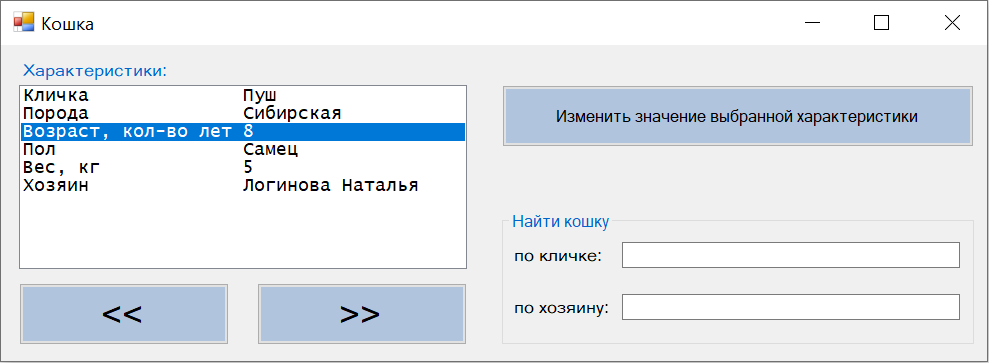


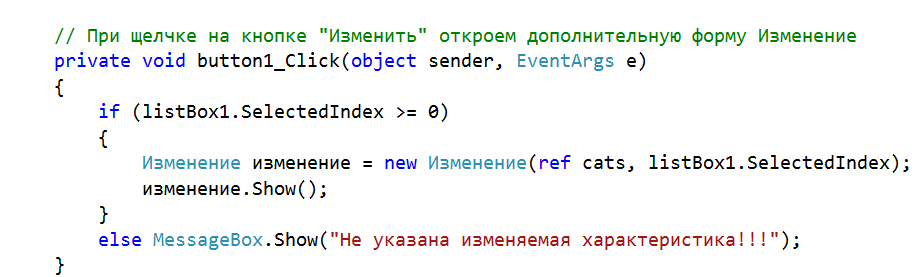




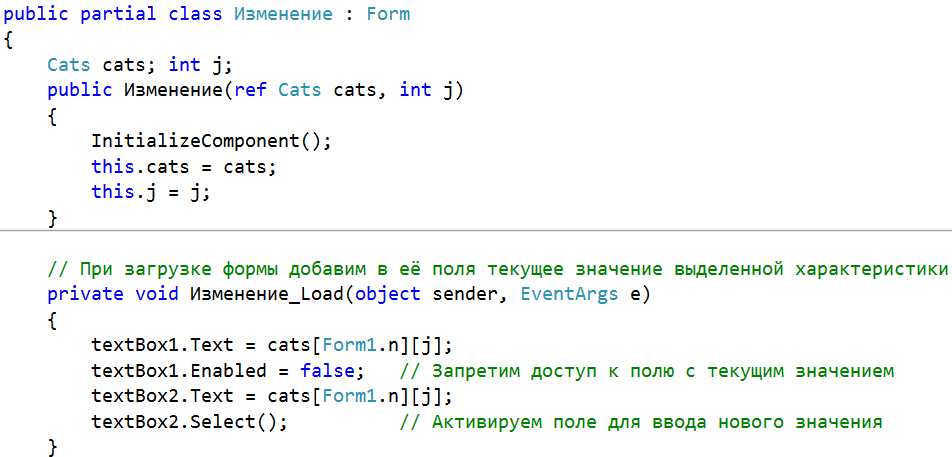


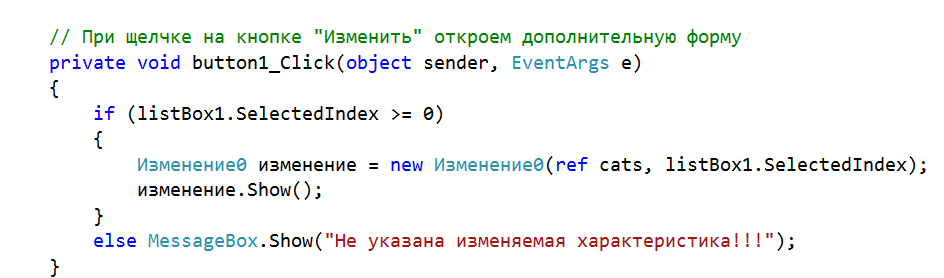


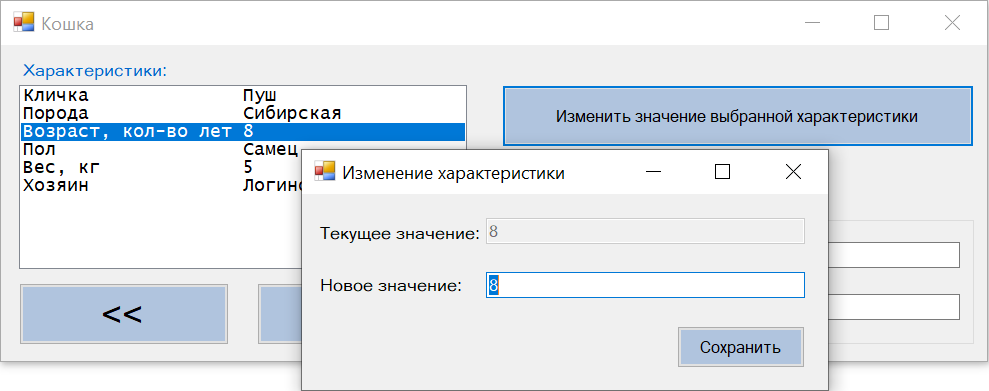


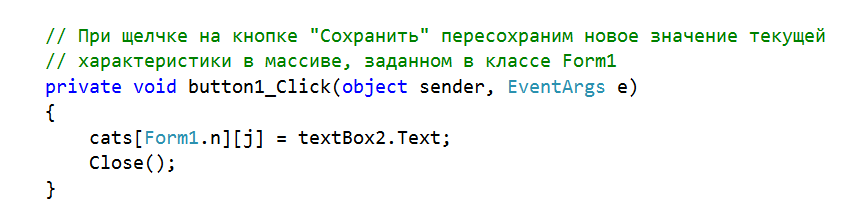


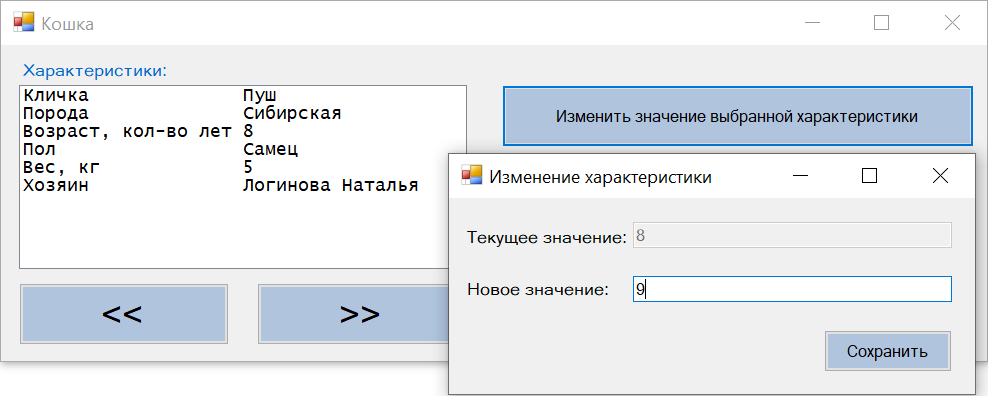
Коды методов класса Изменение (Form2), поддерживающие изменение значений в массиве характеристик:

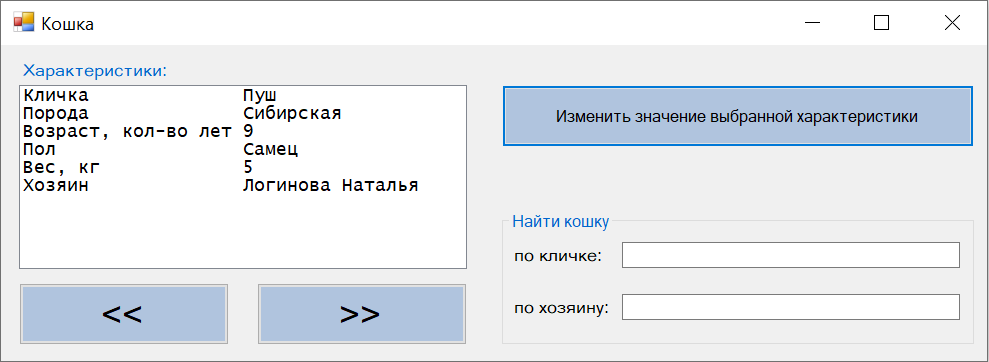












**4 Порядок выполнения работы**

4.1 Изучить теоретические сведения и задание к работе.

4.2. В соответствии с вариантом задания реализовать класс с массивом объектов, где каждый объект описывается как класс с массивом текстовых характеристик объекта. Определить в классах конструкторы, а также индексаторы числового и строкового типа.

4.3. Разработать и отладить методы: Form\_Load() – для инициализации массивов данных (характеристик) созданных классов; Form\_Activated() – для отображения их в списке.

4.4. Разработать методы для выборки и поиска данных в массиве созданных объектов.

4.5. Разработать методы, поддерживающие изменение значений в массиве характеристик объектов.

**5 Содержание отчета**

5.1 Название работы

5.2 Цель работы

5.3 Программные коды и скрины работы разработанного приложения Windows Forms в соответствии с вариантом задания

**6 Варианты заданий**

6.1 Реализовать классы STUDENTS и STUDENT для хранения информации об успеваемости студентов по разным дисциплинам за семестр.

6.2 Реализовать классы EMPLOYEES и EMPLOYEE для хранения информации о заработной плате сотрудников за год.

6.3 Реализовать классы ATHLETES и ATHLETE для хранения информации о физических данных спортсменов.

6.4 Реализовать классы CARS и CAR для хранения информации о параметрах автомобилей.

6.5 Реализовать классы BORROWERS и BORROWER для хранения информации по предоставляемым заемщикам кредитам.

6.6 Реализовать классы PATIENTS и PATIENT для хранения информации о данных пациентов.

6.7 Реализовать классы RATES и RATE для хранения информации о курсах разных валют за неделю.

6.8 Реализовать классы PRODUCTS и PRODUCT для хранения информации о товарах в магазине.

6.9 Реализовать классы BUYERS и BUYER для хранения информации о расходах покупателей за неделю.

6.10 Реализовать классы COLLEGES и COLLEGE для хранения информации о звонках на перемену в колледжах.

6.11 Реализовать классы BUILDINGS и BUILDING для хранения информации о параметрах продаваемых домов.

6.12 Реализовать классы SESSIONS и SESSION для хранения информации о начале сеансов фильмов в кинотеатре.

6.13 Реализовать классы EXAMS и EXAM для хранения информации о количестве оценок разного типа на экзаменах.

6.14 Реализовать классы TEACHERS и TEACHER для хранения информации о данных преподавателей.

6.15 Реализовать классы LIBRARIES и LIBRARY для хранения информации о количестве разных книг в библиотеках.

6.16 Реализовать классы LOCALITIES и LOCALITY для хранения информации о погодных условиях в населенных пунктах.

6.17 Реализовать классы INSTITUTIONS и INSTITUTION для хранения информации о телефонах образовательных учреждений.

**7 Используемая литература**

7.1 Г. Гагарина, Е. В. Кокорева, Б. Д. Виснадул Технология разработки программного обеспечения. Форум, Инфра-М, 2009

7.2 Эндрю Троелсен Язык программирования С# 2010 и платформа .NET 4.0, М., ВИЛЬЯМС, 2011

7.3 А. Марченко C#. Введение в программирование, М, Вильямс, 2009

7.4 http://msdn.microsoft.com/ru-ru/library/67ef8sbd.aspx.