**Лабораторная работа № 7.1**

**Классы и структуры**

**Цель работы:**

Приобретение навыков практического применения, закрепление знаний при создании простейших программ с использованием классов и структур.

**Задачи:**

1. Реализовать решение задания своего варианта с использованием структуры и класса;
2. Реализованный класс должен содержать несколько конструкторов;
3. При хранении списка объектов созданной структуры использовать список (List);
4. Выполнить требования к лабораторным работам (см. файл «Требования к лабораторным»);
5. Ознакомиться с литературой по теме лабораторной работы;
6. Ответить на контрольные вопросы.

**Контрольные вопросы:**

1. Объясните отличия значимых и ссылочных типов;
2. Объясните отличия структур от классов, когда лучше применять структуры и почему?

**Пояснения и примеры:**

Переменные, основанные на *типах значений*, содержат непосредственно значения. При присвоении переменной одного типа значений другому создается копия присваиваемого значения. В этом заключается отличие от переменных *ссылочного типа*, при присвоении которых копируются ссылки на объекты, но не сами объекты.

Все типы значений являются неявными производными от [System.ValueType](http://msdn.microsoft.com/ru-ru/library/system.valuetype(v=vs.90).aspx).

Типы значений состоят из двух основных категорий:

* [структуры](http://msdn.microsoft.com/ru-ru/library/ah19swz4(v=vs.90).aspx)
* [перечисления](http://msdn.microsoft.com/ru-ru/library/sbbt4032(v=vs.90).aspx)

*Структуры* делятся на следующие категории:

* числовые типы
* [целочисленные типы](http://msdn.microsoft.com/ru-ru/library/exx3b86w(v=vs.90).aspx)
* [типы с плавающей запятой](http://msdn.microsoft.com/ru-ru/library/9ahet949(v=vs.90).aspx)
* [decimal](http://msdn.microsoft.com/ru-ru/library/364x0z75(v=vs.90).aspx)
* [bool](http://msdn.microsoft.com/ru-ru/library/c8f5xwh7(v=vs.90).aspx)
* пользовательские структуры.

Переменные *ссылочных типов*, называемые объектами, сохраняют ссылки на фактические данные. В данном разделе описаны следующие ключевые слова, используемые для объявления ссылочных типов:

* [class](http://msdn.microsoft.com/ru-ru/library/0b0thckt(v=vs.90).aspx)
* [interface](http://msdn.microsoft.com/ru-ru/library/87d83y5b(v=vs.90).aspx)
* [delegate](http://msdn.microsoft.com/ru-ru/library/900fyy8e(v=vs.90).aspx)

В данном разделе также описаны следующие встроенные ссылочные типы:

* [object](http://msdn.microsoft.com/ru-ru/library/9kkx3h3c(v=vs.90).aspx)
* [string](http://msdn.microsoft.com/ru-ru/library/362314fe(v=vs.90).aspx)

**Структуры** определяются с помощью ключевого слова [*struct*](http://msdn.microsoft.com/ru-ru/library/ah19swz4(v=vs.90).aspx). Тип struct — это тип значения, который обычно используется для инкапсуляции небольших групп связанных переменных, например координат прямоугольника или характеристик складской номенклатуры. В следующем примере показано простое объявление типа struct:

public struct Book

{

public decimal price;

public string title;

public string author;

}

* Структуры могут также содержать [конструкторы](http://msdn.microsoft.com/ru-ru/library/ace5hbzh(v=vs.90).aspx), [константы](http://msdn.microsoft.com/ru-ru/library/ms173119(v=vs.90).aspx), [поля](http://msdn.microsoft.com/ru-ru/library/ms173118(v=vs.90).aspx), [методы](http://msdn.microsoft.com/ru-ru/library/ms173114(v=vs.90).aspx), [свойства](http://msdn.microsoft.com/ru-ru/library/x9fsa0sw(v=vs.90).aspx), [индексаторы](http://msdn.microsoft.com/ru-ru/library/6x16t2tx(v=vs.90).aspx), [операторы](http://msdn.microsoft.com/ru-ru/library/ms173145(v=vs.90).aspx), [события](http://msdn.microsoft.com/ru-ru/library/awbftdfh(v=vs.90).aspx) и [вложенный типы](http://msdn.microsoft.com/ru-ru/library/ms173120(v=vs.90).aspx), однако, если требуются несколько таких членов, рекомендуется использовать тип class.
* Структуры могут реализовывать интерфейс, но они не могут наследовать от другой структуры. По этой причине члены структуры нельзя объявить как protected.
* Структуры используют большую часть того же синтаксиса, что и классы, однако они более ограничены по сравнению с ними.
* В объявлении структуры поля не могут быть инициализированы до тех пор, пока они будут объявлены как постоянные или статические.
* Структура может не объявлять используемый по умолчанию конструктор (конструктор без параметров) или деструктор.
* Структуры не могут наследовать от классов или других структур.
* Структуры копируются при присваивании. При присваивании структуры к новой переменной выполняется копирование всех данных, а любое изменение новой копии не влияет на данные в исходной копии.
* Структуры являются типами значений, а классы — ссылочными типами.
* В отличие то классов, структуры могут быть созданы без использования нового оператора.
* Структуры могут объявлять конструкторы, имеющие параметры.
* Структура не может быть унаследованной от другой структуры или класса и не может быть основой для других классов. Все структуры наследуют непосредственно отSystem.ValueType, который наследует от System.Object.
* Структуры могут реализовывать интерфейсы.
* Структура может использоваться как тип, допускающий значение NULL, и ей можно назначить значение NULL.

Тип struct подходит для создания несложных объектов, таких как Point, Rectangle и Color. Хотя точку удобно представить в виде [класса](http://msdn.microsoft.com/ru-ru/library/0b0thckt(v=vs.90).aspx) с [автоматически реализуемыми свойствами](http://msdn.microsoft.com/ru-ru/library/bb384054(v=vs.90).aspx), в некоторых сценариях [структура](http://msdn.microsoft.com/ru-ru/library/ah19swz4(v=vs.90).aspx) может оказаться более эффективной. Например, при объявлении массива из 1000 объектов Point потребуется выделить дополнительную память для хранения ссылок на все эти объекты, и структура в таком случае будет более экономичным решением. Поскольку .NET Framework уже содержит объект с именем [Point](http://msdn.microsoft.com/ru-ru/library/system.drawing.point(v=vs.90).aspx), структура в данном примере называется вместо этого "CoOrds".

public struct CoOrds

{

public int x, y;

public CoOrds(int p1, int p2)

{

x = p1;

y = p2;

}

}

Определение конструктора по умолчанию (без параметров) для структуры является ошибкой. Ошибкой также является инициализация поля экземпляра в основной части структуры. Можно инициализировать члены структуры с помощью параметризованного конструктора или путем доступа к членам по отдельности после объявления структуры. Любые закрытые или в ином случае недоступные члены можно инициализировать только в конструкторе.

При создании объекта структуры с помощью оператора [new](http://msdn.microsoft.com/ru-ru/library/51y09td4(v=vs.90).aspx) объект создается и вызывается соответствующий конструктор. В отличие то классов структуры можно создавать без использования оператора new. В таком случае вызов конструктора отсутствует, что делает выделение более эффективным. Однако поля остаются без значений и объект нельзя использовать до инициализации всех полей.

Класс является ссылочным типом. При создании объекта класса переменная, к которой назначается объект, сохраняет только ссылку на память. При назначении ссылки на объект к новой переменной новая переменная ссылается на исходный объект. Изменения, внесенные через одну переменную, отображаются в другой переменной, поскольку обе они ссылаются на одни данные.

Структура является типом значения. При создании структуры переменная, к которой она назначается, сохраняет фактические данные структуры. При назначении структуры новой переменной выполняется ее копирование. Поэтому новая переменная и исходная переменная содержат две отдельных копии одних данных. Изменения, внесенные в одну копию, не влияют на другую копию.

В целом, классы используются для моделирования более сложного поведения, или же создаются данные, предназначенные для изменения после объекта класса. Структуры лучше всего подходят для небольших структур данных, которые содержат преимущественно те данные, которые не предназначены для изменения после создания структуры.

**Литература:**

1. Павловская Т.А. С#. Программирование на языке высокого уровня. Глава 5. Классы: основные понятия 100 – 120 стр., Структуры 212 - 215 стр.;
2. Троелсен Э. - C# и платформа .NET. Структурные и ссылочные типы 85 – 89 стр., Определение структур в С# 131 – 133 стр.;
3. Рихтер C# via .Net. Ссылочные и значимые типы 117 – 123 стр.;
4. Статья «Правда о значимых типах»;
5. Простое объяснение класса: c# для школьников 16 – 45 стр.;
6. C# - лекции IntUit (Биллиг В.А.). Лекция 16: Классы, Лекция 17: Структуры и перечисления;
7. статья «Класс List в C# и его методы».