Как устроен Dictionary<TKey, TValue> в C#

последнее обновление: 22 июня 2023

Связь **ключей** и **значений** это **hash table** или **hash map** (общее название для любого языка программирования).

В C# **Dictionary<TKey, TValue>** это реализация **hash table** или **hash map**

Пример

C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication1
{
    class MyKey
    {
        public string Name { get; set; }
        public int Age { get; set; }

        public override bool Equals(Object obj)
        {
            MyKey key2 = obj as MyKey;
            return (Name.CompareTo(key2.Name) == 0 && Age == key2.Age);
        }

        public override int GetHashCode()
        {
```

```
        return Name.Length * 10 + 3;
    }
}

class Program
{
    static void Main(string[] args)
    {
        Dictionary<MyKey, float> myDict = new Dictionary<MyKey, float>();

        // add item
        MyKey myKey1 = new MyKey() { Name = "Вася", Age = 45 };
        myDict.Add(myKey1, 60.3f);

        // add item
        MyKey myKey2 = new MyKey() { Name = "Евгений", Age = 36 };
        myDict.Add(myKey2, 64.2f);

        // add item
        MyKey myKey3 = new MyKey() { Name = "Петя", Age = 45 };
        myDict.Add(myKey3, 71.6f);

        float weight;
        MyKey myFind;

        // find 1
        myFind = new MyKey() { Name = "Вася", Age = 45 };
        if (myDict.TryGetValue(myFind, out weight))
        {
            Console.WriteLine("{0} {1} {2}", myFind.Name, myFind.Age, weight);
            // на экране увидим
            // Вася 45 60,3
        }

        // find 2
        myFind = new MyKey() { Name = "Петя", Age = 45 };
        if (myDict.TryGetValue(myFind, out weight))
        {
            Console.WriteLine("{0} {1} {2}", myFind.Name, myFind.Age, weight);
            // на экране увидим
            // Петя 45 71,6
        }
    }
```

Отладка кода

- Debug.Assert(false) Отладка кода в C#
- Для отладки, опция "Common Language Runtime Exceptions" увидеть исключения, когда выполняется программа C#
- Атрибут [Obsolete("Мой метод устарел. Не используйте", false)] Предупреждение при компиляции кода в C#

Типы данных C#

- C# типы данных: число (bool, char, byte, int, long, float, double, decimal), текст (string), перечисление (enum), класс (class), структура (struct)
- Структура Boolean В C# это флаг со значениями true или false (bool) и методы для конвертации bool
- Структура Int32 В C# это целое число со знаком (int) и методы для конвертации int
- Структура Single В C# это число с плавающей запятой (float) и методы для конвертации float
- var ... Переменная любого типа В C#. Пример: var str = "Hello!";
- Тип dynamic В C#
- Значения по умолчанию В C#

Хранение объектов в памяти. Удаление объектов из памяти

```
    }
}
```

На заметку!
Когда мы в **dictionary** используем свой класс для **key**, то мы должны:
1) в нашем классе написать метод **GetHashCode** чтобы было как можно меньше коллизий
2) Написать метод **IsEqual** чтобы при коллизиях найти значение
то есть
в примере мы вызываем
myDict.**TryGetValue**(myFind, out weight)
если возникла **коллизия** (ключ преобразуется в hashCode, а по hashCode много значений)
то вызывается **IsEqual**

В Dictionary<TKey, TValue> чтобы по ключу хранить значение используется 2 коллекции

List<int> **buckets**	**buckets** содержит индексы к элементам **entries** **buckets** в переводе на русский это **ведро или корзина**
List<Entry> **entries**	**entries** содержит элементы

C#
```
private struct Entry
{
    public int hashCode;
    public TKey key;
    public TValue value;
    public int next;
}
```


• Ссылочные типы и типы значений В C#

• Стэк (stack) - память для параметров метода и локальных переменных В C#

• Heap - динамическая память доступная во время выполнения программы В C#

• Интерфейс IDisposable. Пишем код для правильного освобождения неуправляемых ресурсов в деструкторе и в интерфейсе IDisposable В C#

• Память. Сборщик мусора (garbage collector). Автоматическое освобождение памяти В C#

C# конвертация типов

• C# конвертация строки в число (string → short, int, long, ushort, uint, ulong, float, double, decimal) | используем Culture (настройки системы)

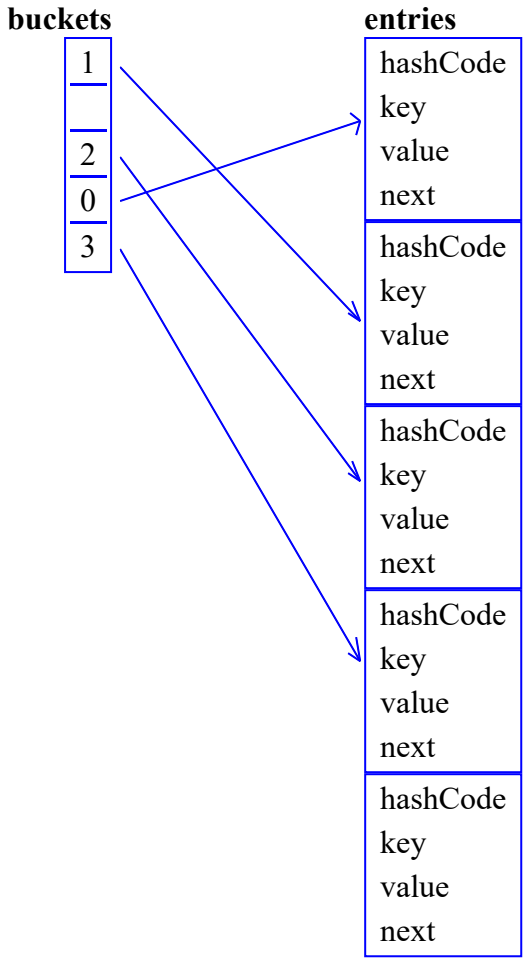
• C# конвертация числа в строку (int, double, short, ... → string) с требуемой точностью

Текст в C# (тип string и класс String)

• Алгоритм пересечения прямоугольников

• Что такое текст В C# ? Тип string и класс String. Методы для работы с текстом.

• Length (длина строки в C#). Пример: string str1 = "Hello"; int



На заметку! Всегда
buckets.Length = entries.Length

При добавлении элемента
Коротко
При добавлении **key** и **value**
1) **получаем hashCode по ключу** то есть так: key.GetHashCode();
2) **считаем индекс по hashCode** вот так: остаток от деления на capacity внутренней коллекции dictionary
3) **записываем key и value по индексу** во внутреннюю коллекцию dictionary

Подробнее

Общая теория	**Как работает на примере**

v1 = str1.Length;

• CompareTo (сравнивает текст с учетом регистра в C#). Пример: bool bIsSame = str1.CompareTo(str2)==0;

• ToLower (конвертирует текст в нижний регистр в C#). Пример: string str1 = "HELLO World!"; string str2 = str1.ToLower();

• ToUpper (конвертирует текст в верхний регистр в C#). Пример: string str1 = "Hello World!"; string str2 = str1.ToUpper();

• Split (разбить строку на слова в C#). Пример: string[] arrWords = strText.Split(' ');

• StartsWith (проверяет начало текста с указанным текстом с учетом регистра в C#). Пример: bool bStart = str1.StartsWith(str2);

• Contains (проверяет содержит текст указанный текст или нет с учетом регистра в C#). Пример: bool bFound = str1.Contains(str2);

• IndexOf (ищет строку с учетом регистра и возвращает позицию в C#). Пример: int pos = str1.IndexOf(str2);

• Substring (возвращает часть текста с указанной позиции и длиной В C#). Пример: string str1 = "Hello World!"; string str2 = str1.Substring(2, 5);

• IsNullOrEmpty (проверяет текст на пустой или на null В C#). Пример: string name = "Hello World!"; bool bFlag = String.IsNullOrEmpty(name);

Добавляем **key** и **value** в dictionary	dictionary.Add(key, value);	C# Код из примера MyKey myKey1 = new MyKey() { Name = "Вася", Age = 45 }; myDict.Add(myKey1, 60.3f);

класс Dictionary автоматически делает следующее:

key преобразуется в **hashCode**	int **hashCode** = key.**GetHashCode**();
	
	когда для различных **ключей** получается одно и то же **hashCode** это называется коллизией
		int **hashCode** = myKey1.**GetHashCode**();
		// hashCode = длина строки("Вася") * 10 + 3
		// hashCode = 43
		
		так как
		C#
		class **MyKey**
		{
		...
		public override int **GetHashCode**()
		{
		return Name.Length * 10 + 3;
		}
		}

hashCode преобразуется в **bucketNum**	int **bucketNum** = (hashcode & 0x7fffffff) % **capacity**;
	
	На заметку!
	capacity это количество элементов выделенных в памяти для коллекции **buckets**
	capacity = **buckets**.Capacity;
		hashCode = 43
		
		Обычно для созданного списка с одним элементом **capacity** = 4
		
		int **bucketNum** = 43 % 4;
		int **bucketNum** = 3;

- IsNullOrWhiteSpace (проверяет текст на null или на текст с пробелами В C#). Пример: string name = " "; bool bFlag = String.IsNullOrWhiteSpace(name);
- [] (возвращает символ с указанной позиции В C#). Пример: char symbol = str[1];
- Format (форматирование текста, строки В C#). Пример: string strNew = String.Format("Hello {0}, {1}", name, year);
- + (добавление строк и текста В C#). Пример: string str = str1 + str2 + " people!";
- $ (интерполяция строк В C#). Пример: string result = $"Hello {a} + {b} = {a + b}";
- Символ @ перед началом строки В C#. Пример: string str1 = @"aaa";
- Используем вместе @ и $ (интерполяцию строк в C#)

DateTime (дата и время) в C#

- Что такое DateTime в C# ? Конвертация в строку с форматом

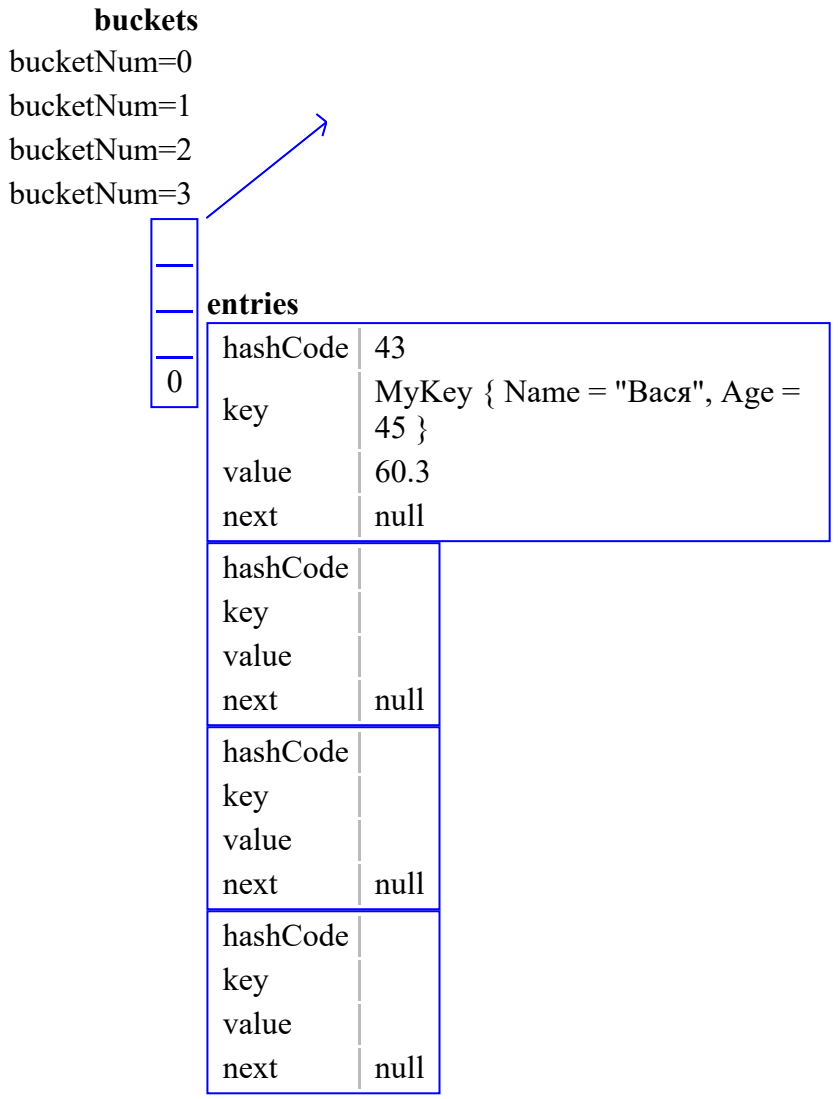
Перечисления в C# (enum)

- Что такое перечисление (enum) В C# ?
- Как преобразовать текст в enum в C#
- Как перечислить все элементы в enum в C#

заполняем **buckets** и **entries**

находим indexEntries это свободный **entries**

записываем **buckets**[bucketNum] = indexEntries;
entries[indexEntries] = {hashCode, next=null, key, value};



добавим новый **key value** в **dictionary**

C# Код из примера
```
MyKey myKey2 = new MyKey(){ Name="Евгений", Age=36};
myDict.Add(myKey2, 64.2f);
```

hashCode = длина строки("Евгений") * 10 + 3
hashCode = 7 * 10 + 3
hashCode = 73

null

- null значение для простых типов. Используем ? или Nullable В C#
- Оператор ?? (null-объединение) В C#

try-catch

- Обработка исключений в C#. Оператор try catch finally

Классы в C# (class)

- Что такое класс В C#?
- Модификаторы доступа класса В C#. Модификаторы доступа для методов, свойств, полей В C#
- 'partial class' В C#. Описание класса в разных файлах

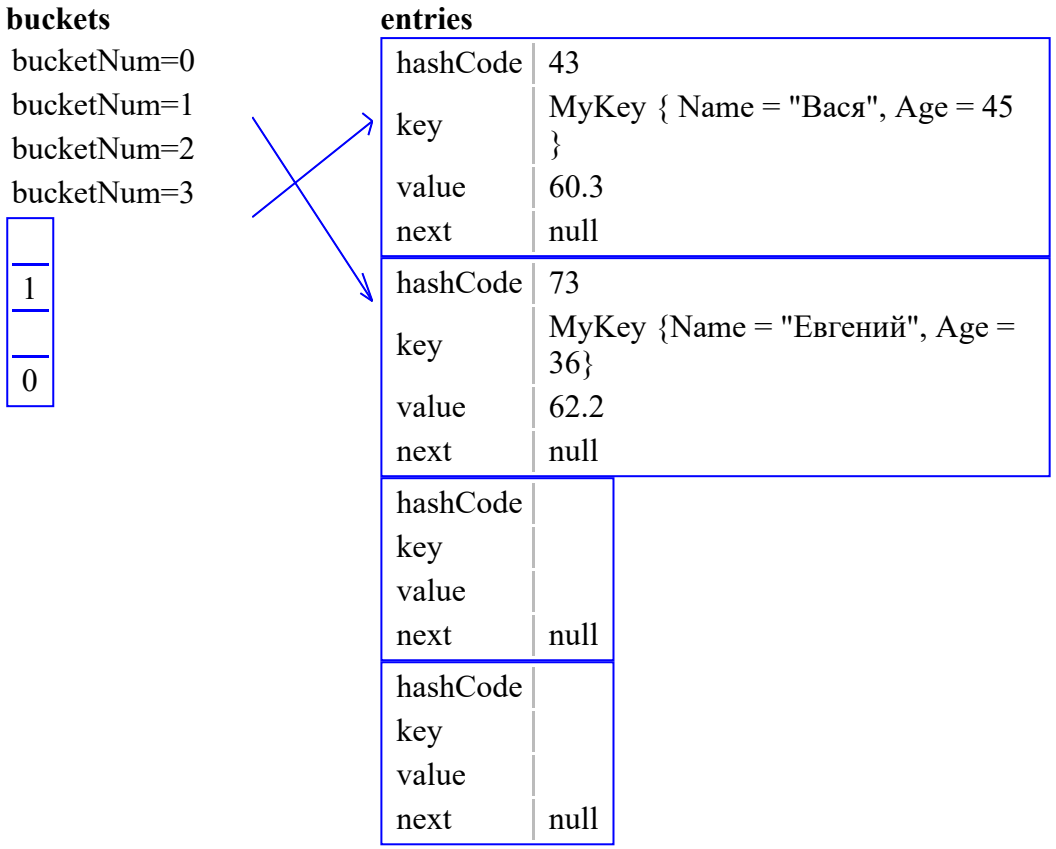
Конструкторы для класса

- Конструктор класса В C#
- Инициализация объекта класса (установка значений для полей) В C#
- Вызов конструктора у базового класса В C#
- Статический конструктор в классе C#
- 'base' Для вызова метода из базового класса. Для вызова переменной из базового класса. Для вызова конструктора из базового класса. C#

hashCode = 73

по прежнему **capacity** = 4

int **bucketNum** = 73 % 4;
int **bucketNum** = 1;



добавим новый **key value** в **dictionary**

C# Код из примера

```
MyKey myKey3 = new MyKey() { Name = "Петя", Age = 45 };
myDict.Add(myKey3, 71.6f);
```

hashCode = длина строки("Петя") * 10 + 3
hashCode = 4 * 10 + 3
hashCode = 43

- 'this' Для установки или получения значения у поля класса. Для вызова конструктора из класса. C#

Деструкторы для класса

- Деструктор класса В C#
- Деструкторы в классах (как вызываются базовые деструкторы) C#

Наследование

- Что такое наследование класса в C# ?

Наследование с использованием new

- Используем new для метода интерфейса. Наследование интерфейса от интерфейса с одинаковым методом
- Используем new для метода класса. Наследование класса от класса в C#.

Наследование с использованием sealed

- sealed class. Запрет наследоваться В C#
- Наследование класса от класса в C#. Используем слова virtual, override, sealed для методов класса

Абстрактный класс

- Что такое абстрактный класс В C# ? Абстрактные методы,

hashCode = 43

по прежнему **capacity** = 4

int **bucketNum** = 43 % 4;
int **bucketNum** = 3;

bucketNum = 3 bucket[bucketNum] уже занята и это коллизия
мы не можем записать

hashCode	43
key	MyKey { Name = "Петя", Age = 45 }
value	71.6
next	null

на

hashCode	43
key	MyKey { Name = "Вася", Age = 45 }
value	60.3
next	null

поэтому для решения колизии мы меняем только **entries**
1) ищем свободный **entries** и туда записываем старый **Entry**

hashCode	43
key	MyKey { Name = "Вася", Age = 45 }
value	60.3
next	null

2) по высчитанному индексу записываем новый **key value** и меняем **next**

hashCode	43
key	MyKey { Name = "Петя", Age = 45 }
value	71.6
next	**индекс в таблице Entry где находится старый Entry**

свойства, индексы.

• Наследование от класса abstract В C#. Используем abstract и override для методов класса

Константы и readonly поля в классе

• Константы в классе C#

• readonly . Для поля класса. Это поле только для чтения в C#

Свойства get и set в классе C# (аксессоры)

• get set Свойства в классе C#

• Наследование (virtual, override) для аксессоров get и set в C#

Операторы, индексаторы в C#

• Операторы в классе C#. Перегрузка операторов: > < ++ + true false

• Индексаторы в классе C#

Вложенные типы в C#

• Вложенный класс, структура в C#

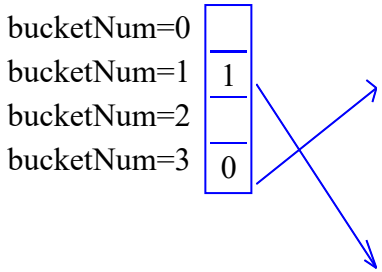
Параметры в методе класса C#

• ref и out (возврат параметров по ссылке в методе) C#. Пример: public void AddValue(ref int value)

• Параметры по умолчанию (необязательные параметры) в методе C#. Пример: public int CalculateSum(int a, int b, int c=7)

вот что получилось:

buckets



entries

hashCode	43
key	MyKey { Name = "Петя", Age = 45 }
value	71.6
next	2

hashCode	73
key	MyKey {Name = "Евгений", Age = 36}
value	62.2
next	null

hashCode	43
key	MyKey { Name = "Вася", Age = 45 }
value	60.3
next	null

hashCode	
key	
value	
next	null

Этот метод для решения коллизий называется метод цепочек

Когда коллекция **entries** полностью заполнена (нет пустых), тогда коллекции **entries** и **bucket** пересоздаются на новый размер и перехешурется **entries**.

На заметку!
Сложность добавления элемента O(1) или O(n) в случае коллизии.

Удаление элемента

• Именованные параметры C#. Пример: public void CalculateSum(a:7, b:3);

Универсальные методы, универсальные классы в C# (шаблоны)

• Метод с универсальными параметрами в C# (шаблоны). Пример: public double Sum<T1, T2>(T1 value1, T2 value2) { ... }

• Обобщенный (типизированный) класс в C# (шаблоны). Пример class Book<T> { ... }

• where Ограничение типа в обобщенном (типизированном) классе в C# (шаблоны). Пример class Dog<T> where T : Cat

Преобразование объекта класса из одного типа в другой

• explicit это явный оператор преобразования в классе C#

• implicit это неявный оператор преобразования в классе C#

• Преобразование объекта класса из одного типа в другой в C#. Используем try ( ) is as

• Преобразование объекта класса из одного типа в другой в C#. Используем pattern matching is switch

Объект класса в C#

• ? оператор условного null в C#

При удалении элемента мы затираем его содержимое значениями по умолчанию
меняем указатели next других элементов при неоходимости
Сложность O(1) или O(n) в случае коллизии.

При очистке всего **dictionary**, его внутренний размер не изменяется.

Взять значение по ключу
Сложность O(1) или O(n) в случае коллизии.

Литература для изучения
1) Под капотом у Dictionary и ConcurrentDictionary: https://habr.com/ru/post/198104/
2) Хеш-таблица: https://ru.wikipedia.org/wiki/Хеш-таблица

← Предыдущая тема
Инициализация элементов в конструкторе Dictionary<TKey, TValue> в C#

Следующая тема →
Как в C# сконвертировать IEnumerable в Dictionary<TKey, TValue> . Используем метод ToDictionary

Ваши Отзывы ... 4 комментариев

гость
17 января 2022
17:05

Спасибо, самая адекватная статья. ответить

Andrey
17 июня 2024 0:27

Читал до этого похожие статьи, но здесь самые понятные примеры, спасибо! ответить

Admin
17 июня 2024 17:55

Спасибо за хорошие отзывы.
Когда я с опытом работы проходил собеседования по **C#** меня часто спрашивали про **Dictionary**
Чтобы поделиться и не забыть со временем подробности о **Dictionary** я добавил на сайт.
C# язык программирования мне очень нравится. Но пришлось немного :) писать и на других языках.
C# рекомендую изучать. ответить

• Объект класса содержит ссылку в C#

• Как чтобы при копировании объектов в C# копировались данные класса, а не ссылка?

Статический конструктор и статические свойства и методы

• Статический конструктор в классе C#

• Статические методы, свойства, члены в классе C#

Дополнительные возможности класса в C#

• Метод расширения в C# (this в первом параметре метода). Пример: static public void AddValues(this List<int> myList, int value1, int value2)

Правила именования классов в C#

• Какими буквами строчными или заглавными называть классы, методы, свойства ... в C#

• Правильно ли для каждого класса в C# создавать свой .cs файл? Или писать классы C# в одном .cs файле?

Статический класс

• Статический класс в C#

Анонимный класс

гость
30 октября 2024
21:54

В статье написано, мол при добавлении нового значения и возникновении коллизии, старый Entry как-будто перемещается на свободное место, а на его место встает новый с новым значением. Немного просмотрел исходных код и кажется, что новое значение встает в свободное Entry, его поле next начинает указывать на старый Entry, а индекс в Bucket меняется на индекс нового созданного Entry. Могу ошибаться, но в других статьях тоже объясняется примерно так. ответить

гость (18 ноября 2024 10:11) Согласен с вами ответить

Ваше имя гость

Ваш комментарий (www ссылки может добавлять только залогиненный пользователь)

+ картинку

• Объект с анонимным (отсутствующим) типом в C#. Пример: var book = new { BookName = "Властелин Колец", Price = 100 };

Интерфейсы

• Что такое interface в C# ?

• Наследование interface от interface в C#

• Наследование класса от класса от interface в C#. Используем override и virtual для методов класса

• Обобщенный (типизированный) интерфейс в C# (шаблоны). Пример interface IUser<T> { ... }

Структура struct

• Что такое структура в C#?

• Модификаторы доступа структуры в C#. Модификаторы доступа для методов, свойств, полей структуры в C#

• Инициализация объекта структуры (установка значений для полей) в C#

• Как поменять значение в массиве структур или в коллекции структур (List) в C#

• Вложенная структура в C#

Преобразование объекта структуры из одного типа в другой