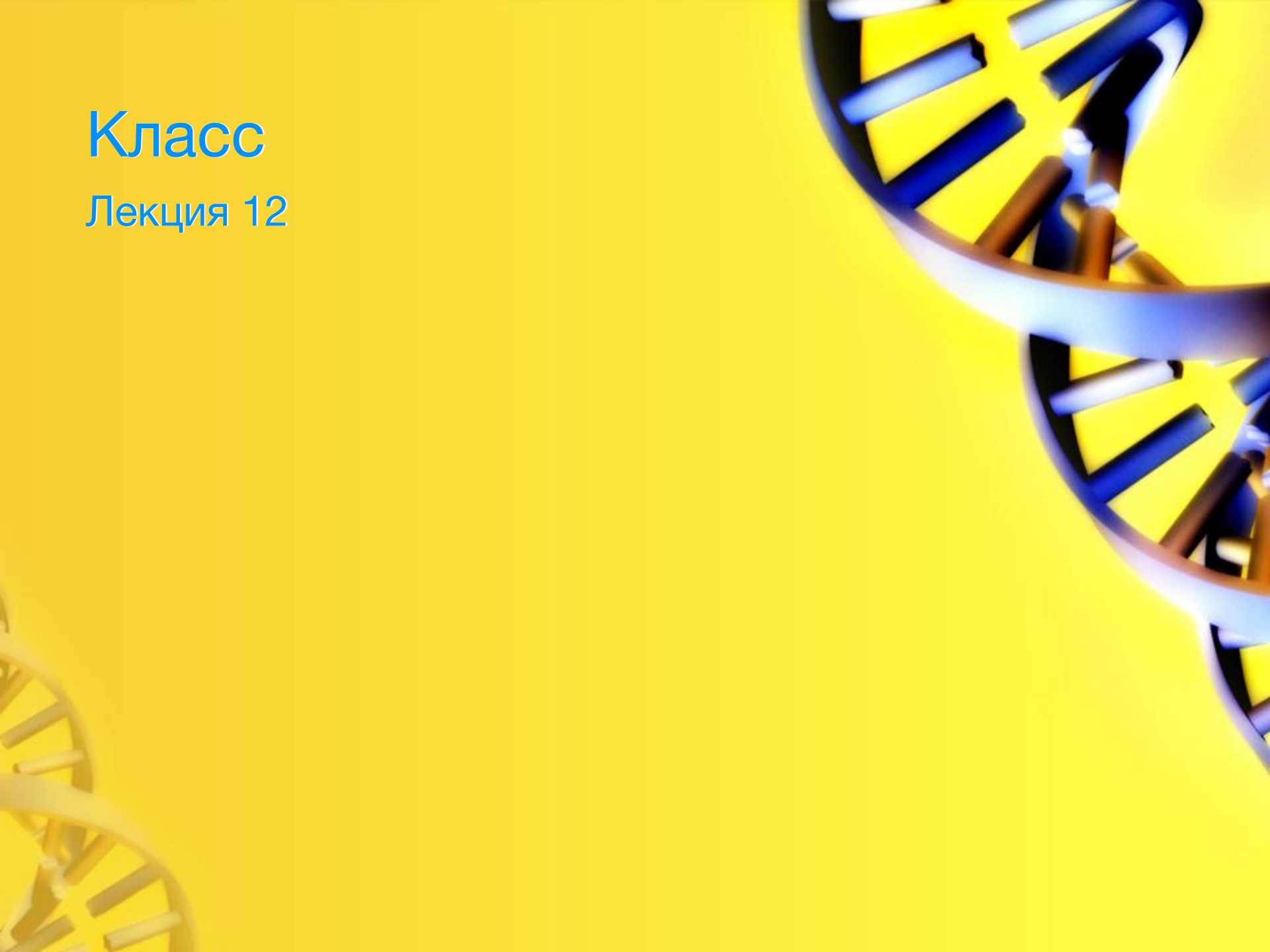


# Класс

## Лекция 12



# Класс определение

- **Класс** представляет собой шаблон, по которому определяется форма объекта.

*Он должен представлять собой одну логическую сущность, например, являться моделью реального объекта или процесса. Элементами класса являются данные и функции, предназначенные для их обработки.*

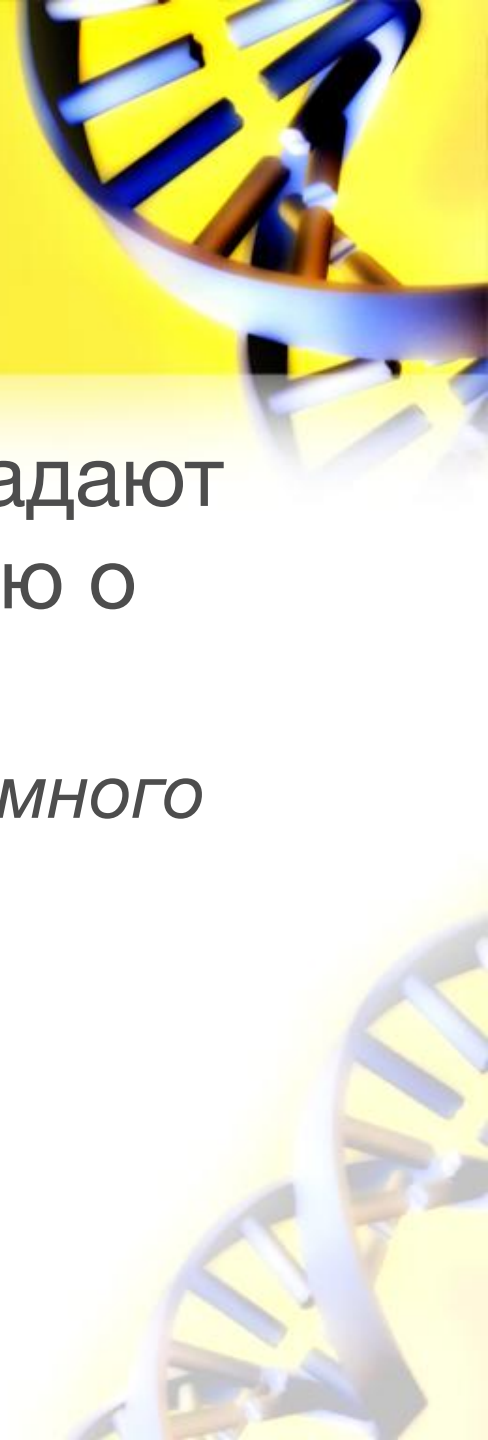
# Описание класса

- **[ атрибуты][модификатор][  
модификатор доступа] class имя  
класса [ : предки ]  
{ тело-класса}**

Как видите, обязательными являются только ключевое слово **class** , а также имя и тело класса.

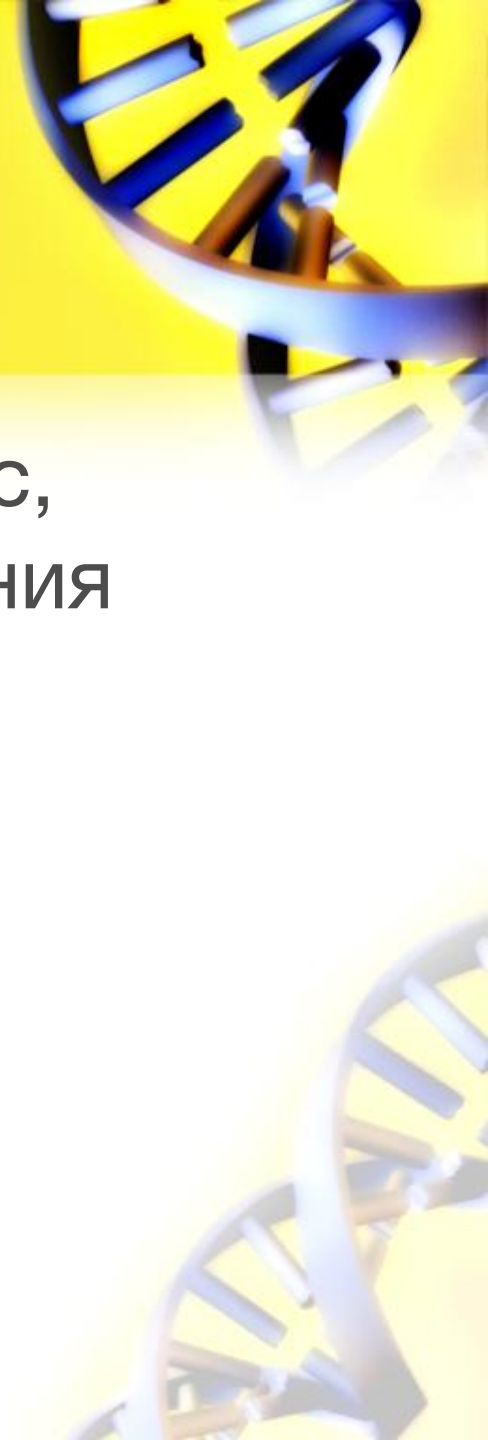
# Атрибуты

- Необязательные атрибуты задают дополнительную информацию о классе.
- *Изучение их будет отдельно и намного позднее*



# Модификаторы

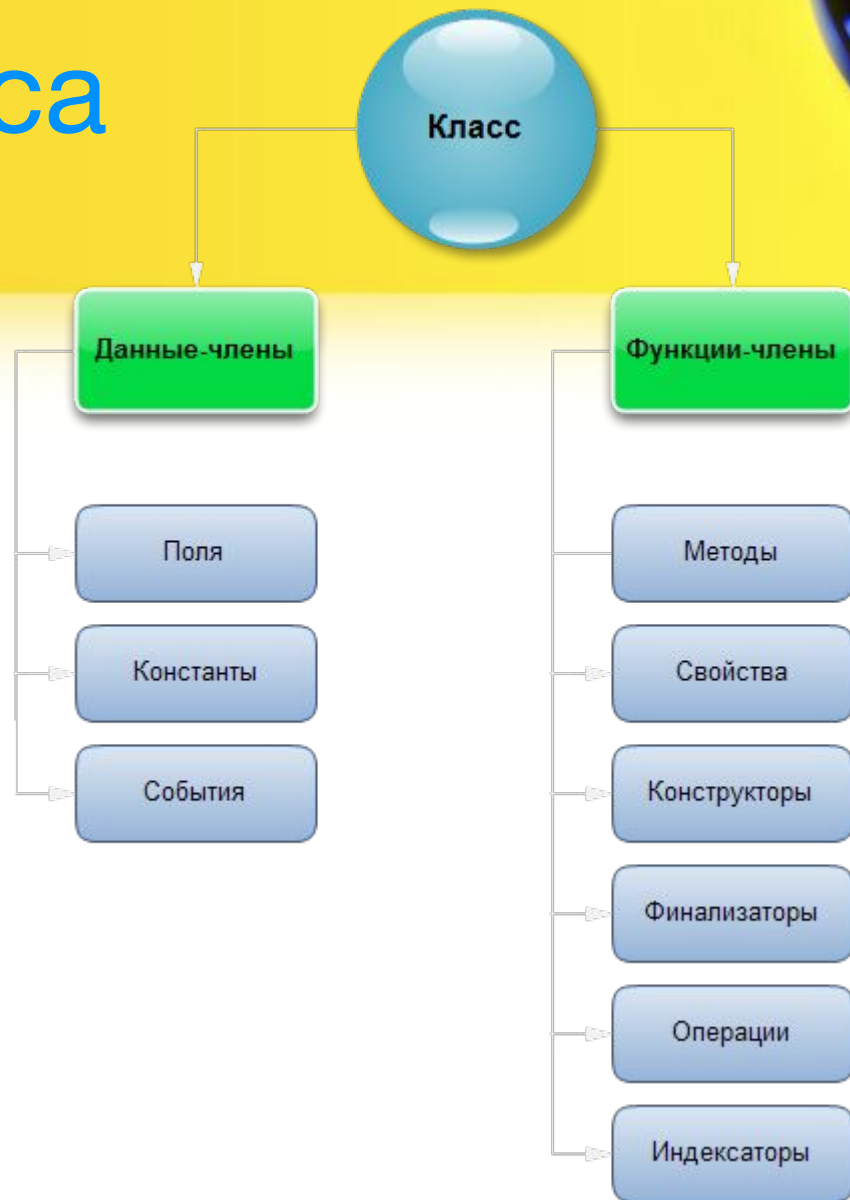
- `abstract` - абстрактный класс, служит только для порождения потомков.
- `sealed` - бесплодный класс, запрещает наследование от класса.
- `static` - статический класс, используется без создания объекта.



# Модификаторы доступа

- *public* – доступ к классу возможен из любого места одной сборки либо из другой сборки, на которую есть ссылка;
- *internal* – доступ к классу возможен только из сборки, в которой он объявлен.

# Тело класса





# Данные-члены класса

- *Данные-члены* — это те члены, которые содержат данные класса.





# Данные-члены класса

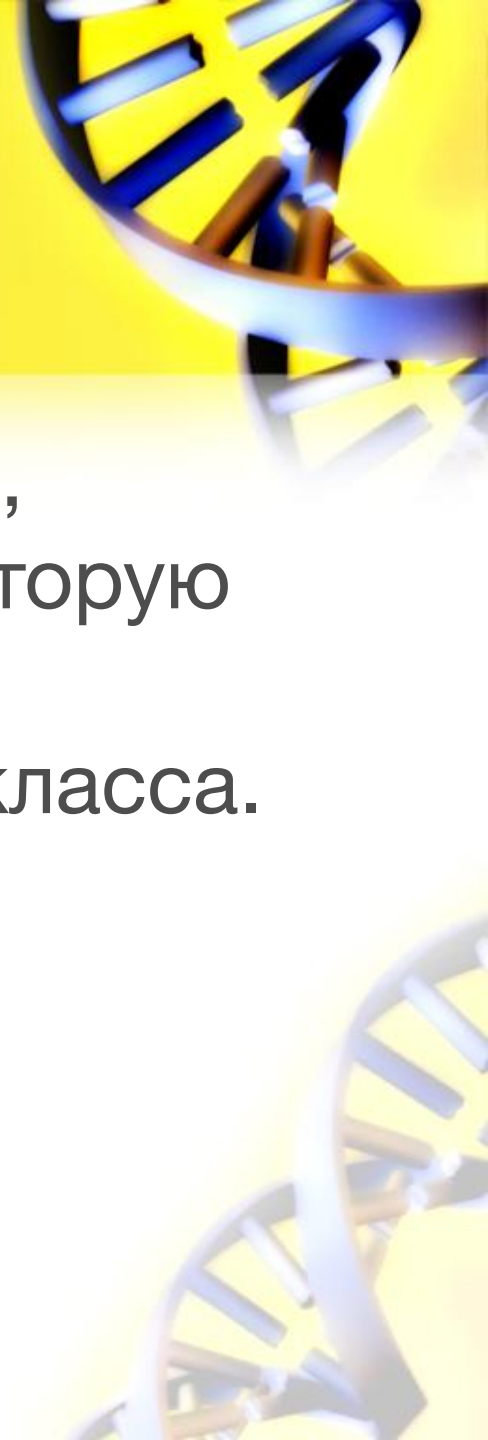
- **Поля (*field*)** Это любые переменные, ассоциированные с классом.
- **Константы** могут быть ассоциированы с классом тем же способом, что и переменные. Константа объявляется с помощью ключевого слова **const**. Если она объявлена как **public**, то в этом случае становится доступной извне класса.

# Данные-члены класса

- **События** Это члены класса, позволяющие объекту уведомлять вызывающий код о том, что случилось нечто достойное упоминания, например, изменение свойства класса либо некоторое взаимодействие с пользователем. Клиент может иметь код, известный как обработчик событий, реагирующий на них.

# Функции члены класса

- *Функции-члены* — это члены, которые обеспечивают некоторую функциональность для манипулирования данными класса.



# Функции члены класса

- **Методы (*method*)** Это функции, ассоциированные с определенным классом.

```
class Printer
{
    public void Print(){
        //код метода    }  }
```

# Функции члены класса

**Свойства (*property*)** Это наборы функций, которые могут быть доступны клиенту таким же способом, как общедоступные поля класса. В C# предусмотрен специальный синтаксис для реализации чтения и записи свойств для классов, поэтому писать собственные методы с именами, начинающимися на Set и Get, не понадобится.

# Функции члены класса

```
private string text;  
    public string Text  
    {  
        set { text = value; }  
        get { return text; }  
    }
```

# Функции члены класса

- **Конструкторы (*constructor*)** Это специальные функции, вызываемые автоматически при инициализации объекта. Их имена совпадают с именами классов, которым они принадлежат, и они не имеют типа возврата. Конструкторы полезны для инициализации полей класса.



# Функции члены класса

```
class Printer
{
    |
    ссылка 0
    public void Print();

    byte[] info;
    ссылка 0
    public Printer(byte[] info)
    {
        this.info = info;
    }
}
```

# Функции члены класса

- **Финализаторы (*finalizer*)**

Вызываются, когда среда CLR определяет, что объект больше не нужен. Они имеют то же имя, что и класс, но с предшествующим символом тильды. Предсказать точно, когда будет вызван финализатор, невозможно.

# Функции члены класса

- **Операции (*operator*)**
- Это простейшие действия вроде + или -. Когда вы складываете два целых числа, то, строго говоря, применяете операцию + к целым. Однако C# позволяет указать, как существующие операции будут работать с пользовательскими классами (так называемая перегрузка операции).

# Функции члены класса

- **Индексаторы (*indexer*)**
- Позволяют индексировать объекты таким же способом, как массив или коллекцию.

В общем случае

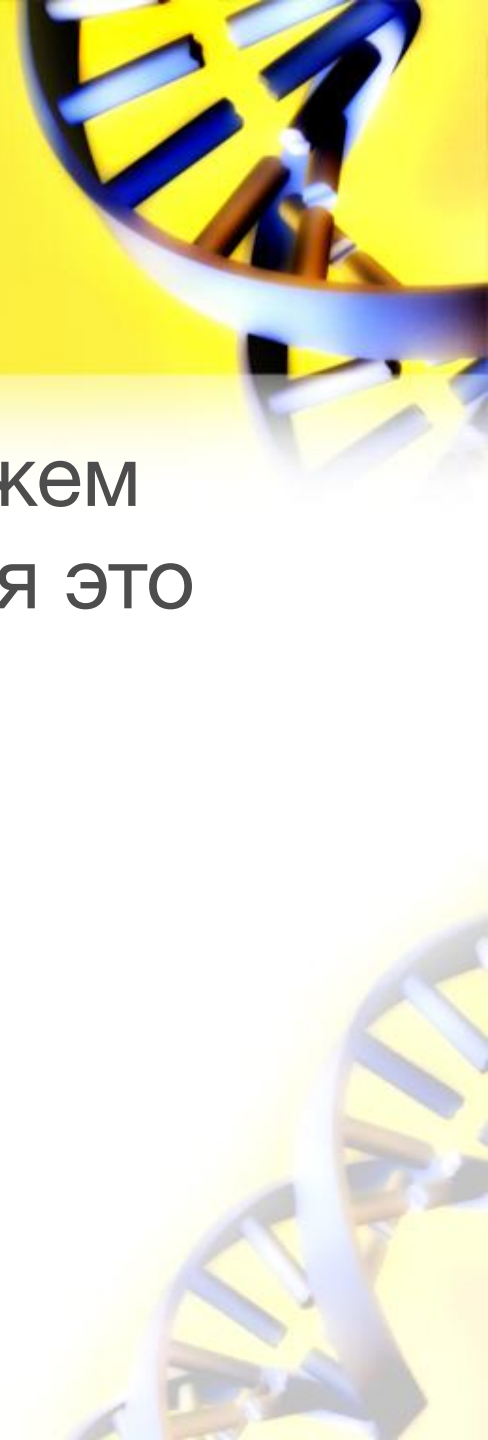
```
public тип возвращаемого значения this[int  
    index] {  
    // set и get методы }
```

# Функции члены класса

```
class TempRecord {  
    private float[] temps = new float[10] { 56.2F,  
        56.7F, 56.5F, 56.9F, 58.8F, 61.3F, 65.9F,  
        62.1F, 59.2F, 57.5F };  
    public int Length { get { return temps.Length;  
    } }  
    public float this[int index] {  
        get { return temps[index]; }  
        set { temps[index] = value; } } }  
}
```

# Создание объектов

Объявив класс, мы теперь можем создавать объекты. Делается это при помощи ключевого слова *new* и имени класса:



# Создание объектов

```
namespace HelloWorld
{
    class Student
    {
        private string firstName;
        private string lastName;
        private int age;
        public string group;
    }
    class Program
    {
        static void Main(string[] args)
        {
            Student student1 = new Student(); //создание объекта
            student1 класса Student
            Student student2 = new Student();
        }
    }
}
```