**чЛАБОРАТОРНАЯ РАБОТА 6 СТРУКТУРЫ, ПЕРЕЧИСЛЕНИЯ**

1. Чем класс отличается от структуры? Что может и чего не может быть в структуре?

*Структура* – это составной объект (пользовательский тип данных), совокупность логически связанных данных различного типа, объединенных под одним идентификатором.

Структура относится к типу значения, класс – к ссылочному типу данных, т.е. структуры размещаются в стеке, а классы – в куче.  
Также структуры не поддерживают наследование.

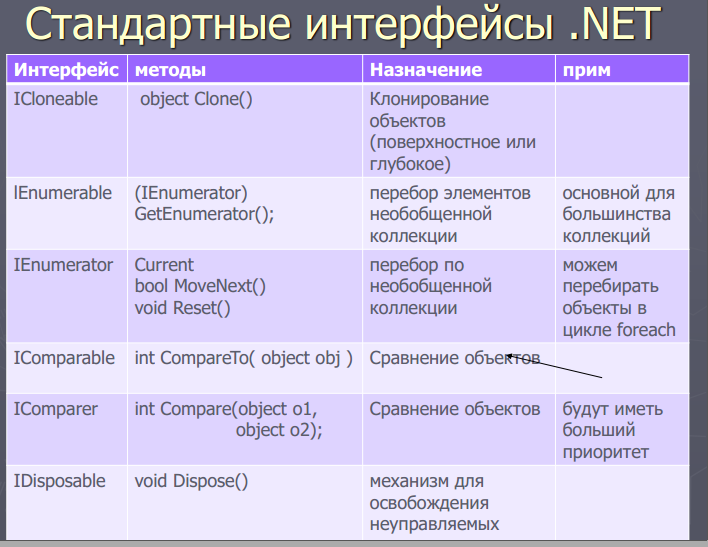
Может: структура этого же типа, указатель на такую же структуру, указатель на функцию, прототип функции, объединение, перечисление, реализовывать интерфейсы, значение NULL, объявление конструкторов с параметрами.

Не может: инициализированные при объявлении поля (если не константы и не статические), не может объявлять конструктор (деструктор) по умолчанию.

1. Что такое перечисление? Приведите пример определения и использования перечисления.

Перечисление -- тип данных, чьё множество значений представляет собой ограниченный список идентификаторов.

1. Перечислите и поясните стандартные интерфейсы .Net?



1. Что такое полиморфизм? Перечислите его формы. Приведите примеры.

Полиморфизм – возможность объектов с одинаковой спецификацией иметь различную реализацию. Способность объекта использовать методы производного класса, который не существует на момент создания базового.

Зачем в классе определяют виртуальные методы?

**Виртуальные** **методы** базового **класса** **определяют** интерфейс в любом производном **классе**

Как сделать запрет переопределения методов?

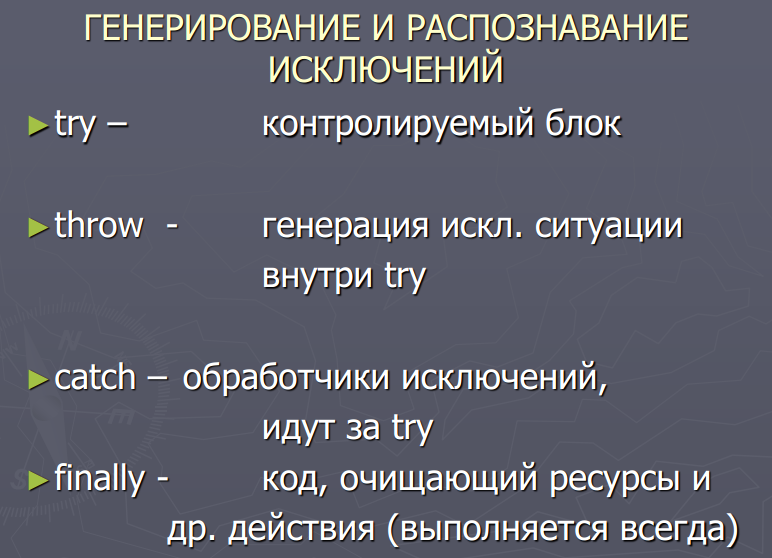
**Sealed**

**ЛАБОРАТОРНАЯ РАБОТА 7 ИСКЛЮЧЕНИЯ**

1. При использовании блока **try...catch..finally** вначале выполняются все инструкции в блоке **try**. Если в этом блоке не возникло исключений, то после его выполнения начинает выполняться блок **finally**. И затем конструкция try..catch..finally завершает свою работу.
2. Если же в блоке try вдруг возникает исключение, то обычный порядок выполнения останавливается, и среда CLR начинает искать блок **catch**, который может обработать данное исключение. Если нужный блок catch найден, то он выполняется, и после его завершения выполняется блок finally.
3. Если нужный блок catch не найден, то при возникновении исключения программа аварийно завершает свое выполнение.
4. Какое ключевое слово служит для обозначения блока кода, в котором можно генерировать исключение?

Throw

1. Какие ключевые слова используются для обработки и генерации исключений? Расскажите об механизме обработке исключения?

в к

* try-catch, try-finally, try-catch-finallyа

Когда внутри такого блока try возникает исключение, поток управления переходит к первому подходящему обработчику исключений в стеке вызовов. В C# ключевое слово catch обозначает обработчик исключений.

* Если для созданного исключения не существует обработчиков, выполнение программы прекращается с сообщением об ошибке.

1. Что будет, если в программе нет предложения catch, способного обработать  
   исключение?

При возникновении исключения среда CLR ищет блок catch, который может обработать данное исключение. Если такого блока не найдено, то пользователю отображается сообщение о необработанном исключении, а дальнейшее выполнение программы останавливается. И чтобы подобной остановки не произошло, и надо использовать блок try..catch.

1. Что такое фильтры исключения? Приведите пример.

Фильтры исключений — это механизм, который позволяет блоку catch объявить предусловия,

которым должно удовлетворять исключение, дабы быть пойманным данным блоком.

public void Main()

{

try

{

throw new Exception("E2");

}

catch(Exception ex) when(ex.Message == "E1")

{

Console.WriteLine("caught E1");

}

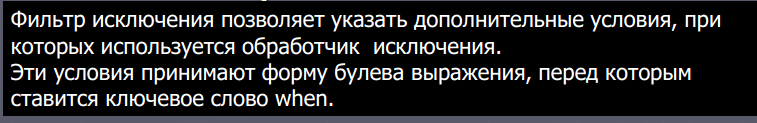
catch(Exception ex) when(ex.Message == "E2")

{

Console.WriteLine("caught E2");

}

}



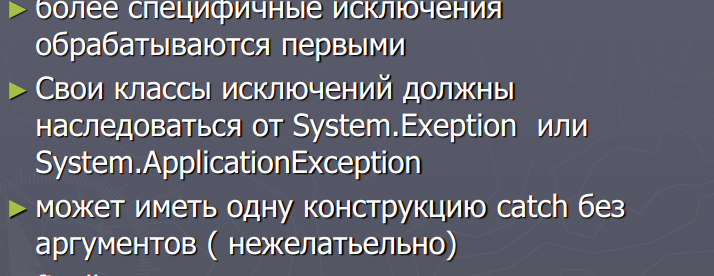
1. Могут ли исключения быть вложенными?

Могут.

1. Какой синтаксис нужно использовать в C# для отлова любого возможного исключения?

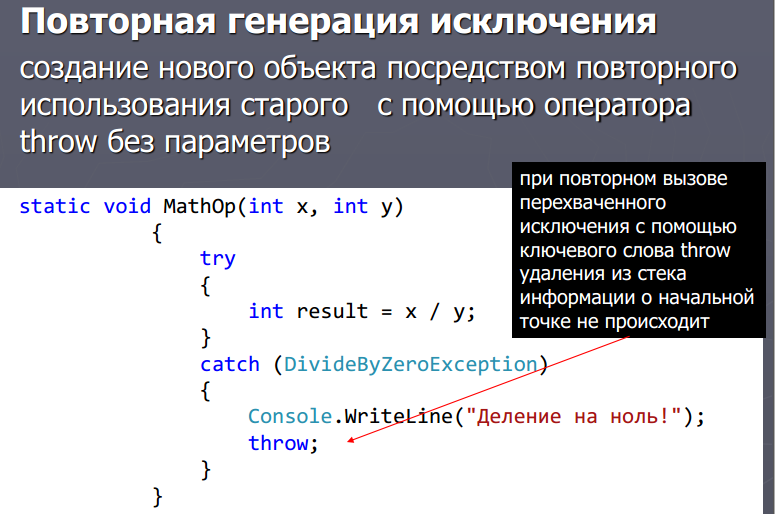
Блок catch, отлавливающий исключения типа System.Exception. Также можно просто использовать catch{}, не указывая тип исключения.

1. Чем следует руководствоваться при размещении обработчиков исключения?

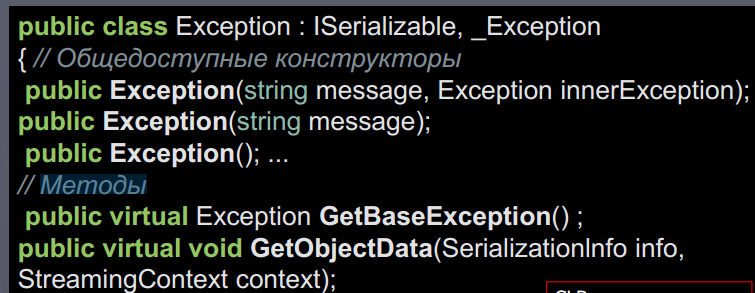


1. Как повторно сгенерировать то же самое исключение в блоке обработчике catch?

Throw без ничего.



1. Какие методы содержатся в классе Exception? Где и как их можно использовать?



1. При переопределении в производном классе возвращает исключение [Exception](https://msdn.microsoft.com/ru-ru/library/system.exception(v=vs.110).aspx), которое является корневой причиной одного или нескольких последующих исключений.

2. При переопределении в производном классе задает объект [SerializationInfo](https://msdn.microsoft.com/ru-ru/library/system.runtime.serialization.serializationinfo(v=vs.110).aspx) со сведениями об исключении.

**ЛАБОРАТОРНАЯ РАБОТА 8 ОБОБЩЕНИЯ (GENERIC)**

1. Что такое обобщение (generic)?

**Обобщение** — средство языка C#, позволяющее создавать программный код, содержащий единственное (типизированное) решение задачи для различных типов, с его последующим применением для любого конкретного типа (int, float, char и т.д.).

Термин обобщение означает параметризированный тип.

Обобщённое программирование (англ. generic programming) — парадигма программирования, заключающаяся в таком описании данных и алгоритмов, которое можно применять к различным типам данных, не меняя само это описание.

Свойства:

* Обобщённый тип может содержать другой обобщённый тип
* Обобщённые типы перегружаются на основе количества параметров
* Обобщёнными могут быть классы, структуры, интерфейсы, делегаты, методы
* Могут содержать статические типы
* Доступность конструируемых типов определяется на основе пересечения доступности обобщённого типа и типа в списке аргументов.
* Поддерживает механизм исключений

1. Как можно наложить определенное ограничение на параметр?

Для выхода из подобных ситуаций в C# предусмотрены ограниченные типы. Указывая параметр типа, можно наложить определенное ограничение на этот параметр. Это делается с помощью оператора **where** при указании параметра типа:

class имя\_класса<параметр\_типа> where параметр\_типа : ограничения { // ...

где ограничения указываются списком через запятую.

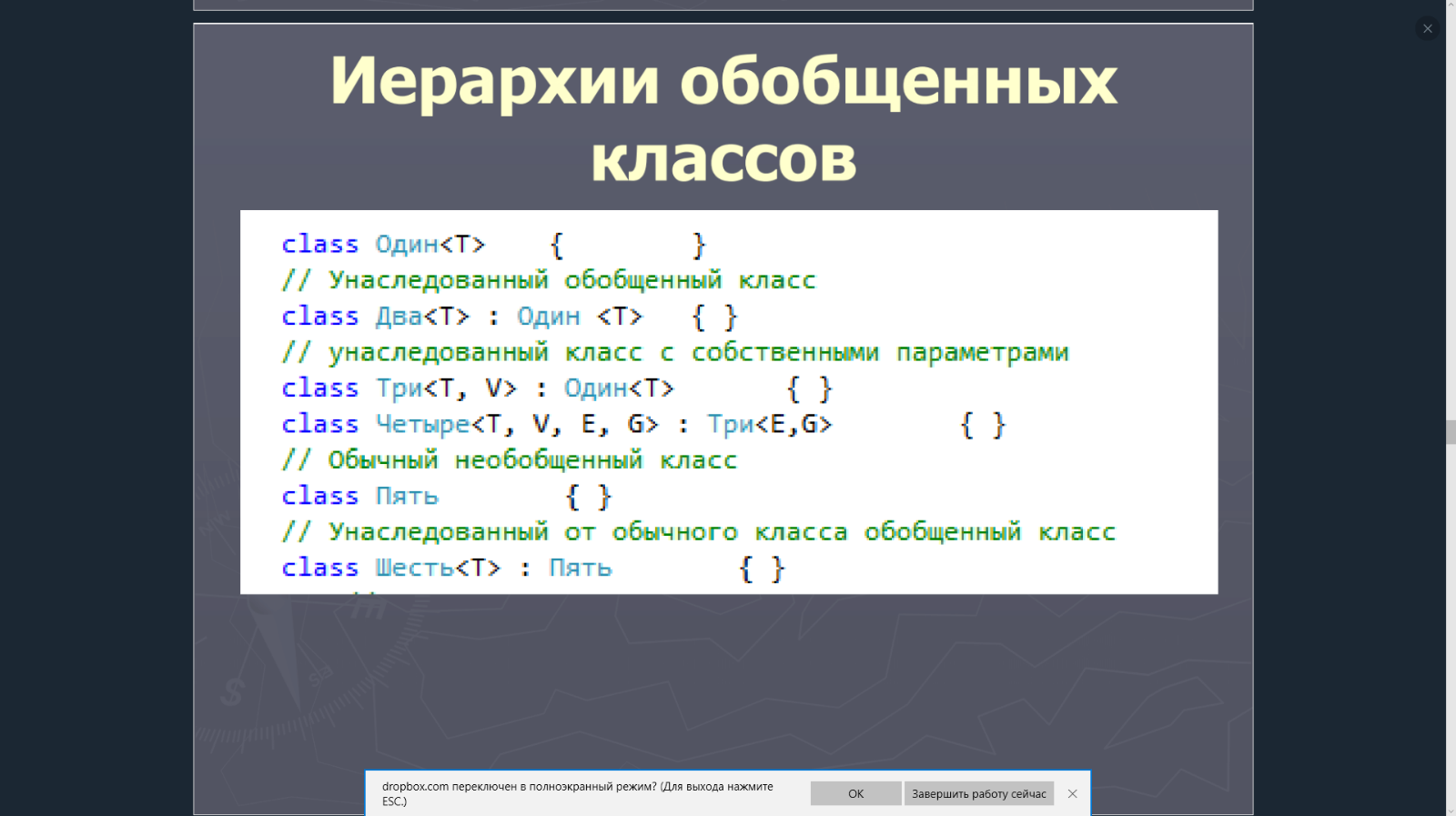
1. Как можно наложить несколько ограничений на параметр?

Указать несколько через запятую.

1. Перечислите все существующие ограничения на типы данных обобщения?

* Ограничение на интерфейс (ограничение сужает перечень типов, которые можно передать в обобщённом аргументе, и расширяет возможности по работе с этими типами.
* Ограничение типа значения (гарантирует компилятору, что указанный аргумент будет иметь значимый тип (с поддержкой null не подходят под это)).
* Ограничение на базовый класс.
* Ограничение ссылочного типа.
* Ограничение на конструктор.
* Ограничение на связь параметров.

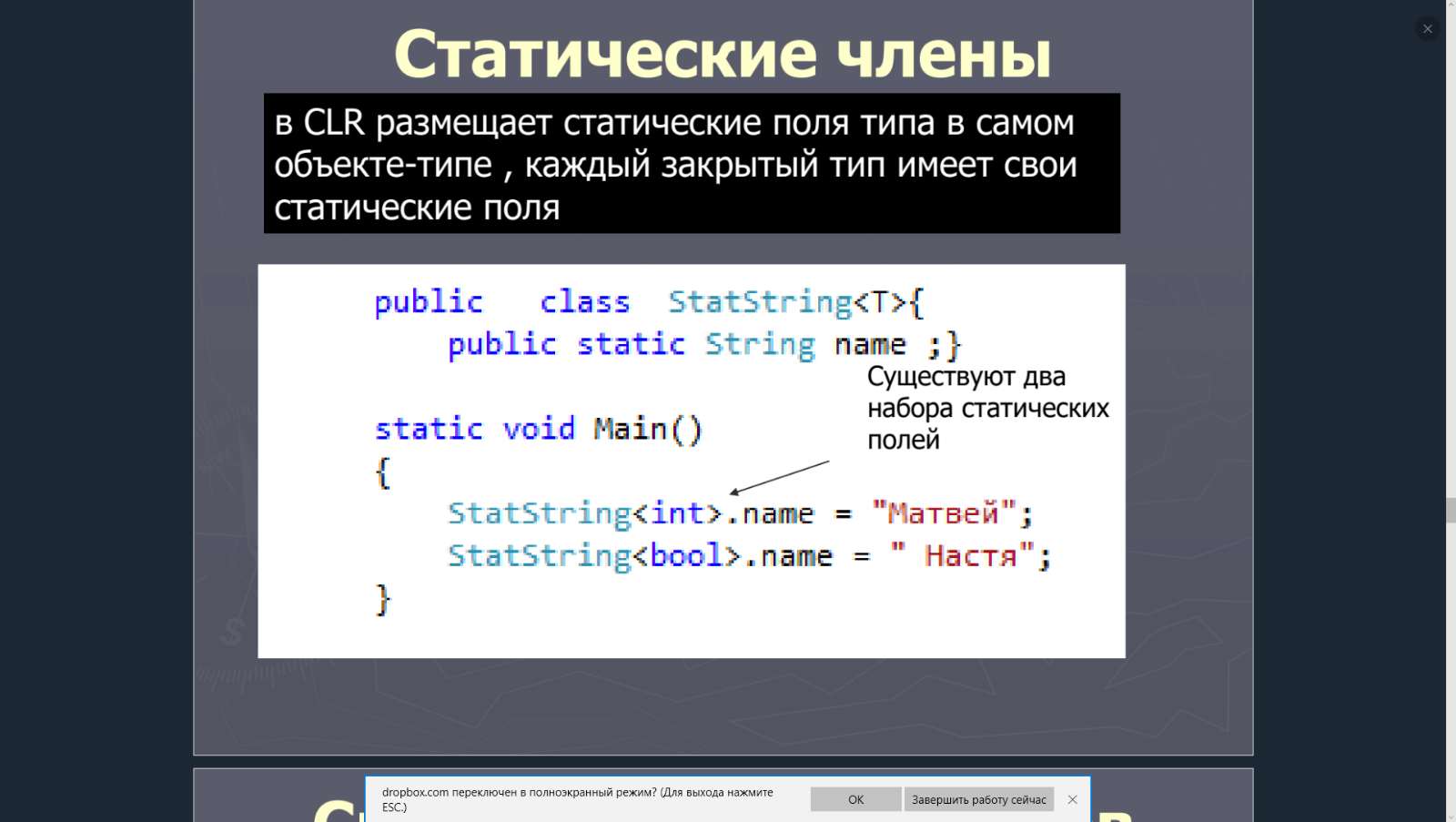
1. Приведите примеры, когда обобщенный класс может действовать как базовый или производный класс.



1. В каких случаях в обобщениях может использоваться оператор default?

При использовании переменных обобщённого типа в качестве операндов (поддержка обобщений в среде CLR).

1. Поясните как использовать статические переменные в обобщенных классах.



1. В чем отличие обобщенных классов от обобщенных структур?

Структуры не наследуются.

1. Какие классы для работы с файлами вы знаете? Приведите пример.

File, FileStream, BinaryFormatter.

**ЛАБОРАТОРНАЯ РАБОТА 9 ДЕЛЕГАТЫ , СОБЫТИЯ**

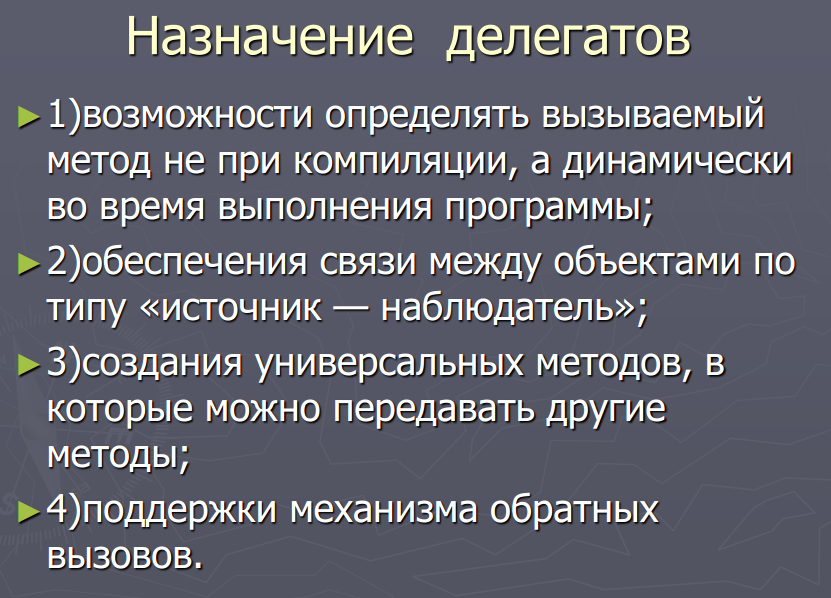
1. Что такое делегат? Как определить делегат?

***Делегат*** представляет собой объект, который может ссылаться на метод

*delegate возвращаемый\_тип имя (список\_параметров);*

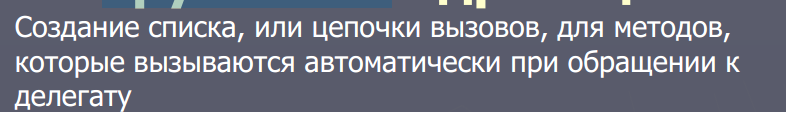
1. Назначение делегатов

То есть делегаты - это указатели на методы и с помощью делегатов мы можем вызвать данные методы



1. Какие есть способы присваивания делегату адреса метода?

Через конструктор

1. Чтобы вызвать метод, на который указывает делегат, надо использовать его метод**Invoke**. используется для синхронного вызова каждого из методов, поддерживаемых объектом делегата
2. Что такое групповая адресация делегата?
3. Как создать событие?

*event делегат\_события имя\_события;*

1. Как события связаны с делегатами?

*Событие* — это именованный делегат, при вызове которого, будут запущены все подписавшиеся на момент вызова события методы заданной сигнатуры.

1. Что такое лямбда-выражения?

Лямбда-выражения позволяют создать методы, которые могут возвращать некоторое значение и которые можно передать в качестве параметров в другие методы.

1. Что такое ковариантность и контравариантность делегатов?

В частности, ***ковариантность*** позволяет присвоить делегату метод, возвращаемым типом которого служит класс, производный от класса, указываемого в возвращаемом типе делегата. А ***контравариантность*** позволяет присвоить делегату метод, типом параметра которого служит класс, являющийся базовым для класса, указываемого в объявлении делегата.

1. Поясните разницу между встроенными делегатами Action и Func.

Делегат Action является обобщенным, принимает параметры и возвращает значение void.  **Func** возвращает результат действия и может принимать параметры.