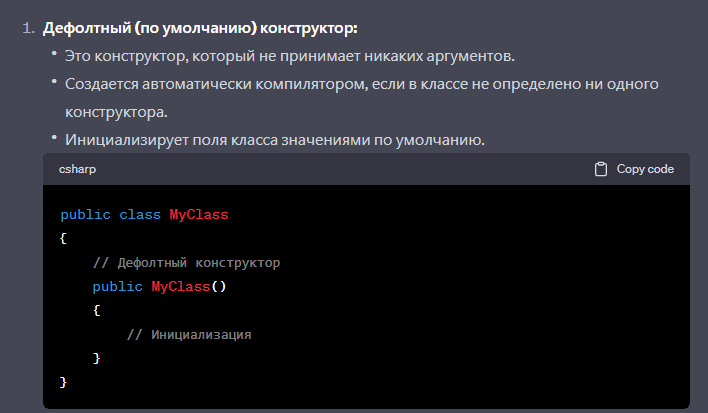
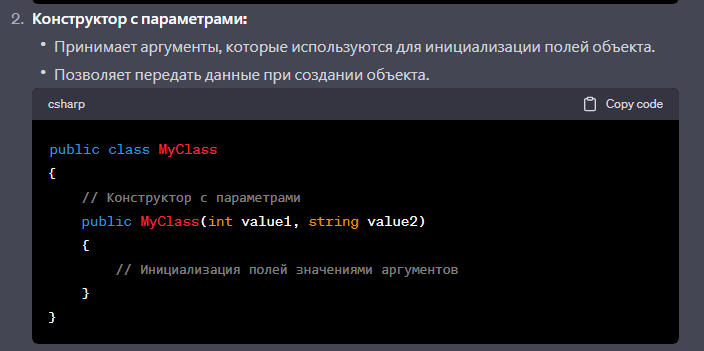
1. Все модификаторы доступа полностью

[C# и .NET | Модификаторы доступа (metanit.com)](https://metanit.com/sharp/tutorial/3.2.php)

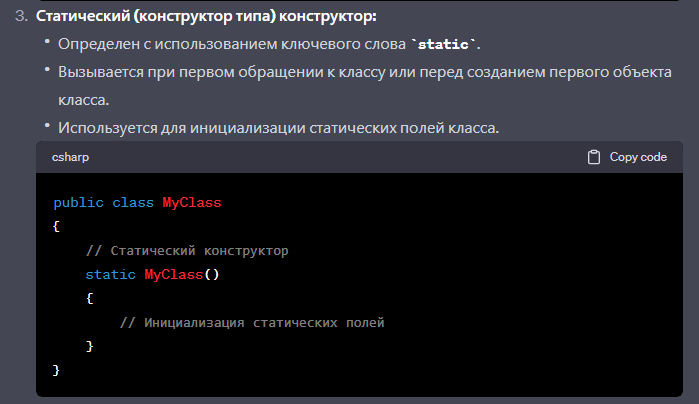
1. Виды конструктора (статичный, копирования, дефолтный, с параметрами, без параметров) и всё о них.
2. Дефолтный (по умолчанию) конструктор:



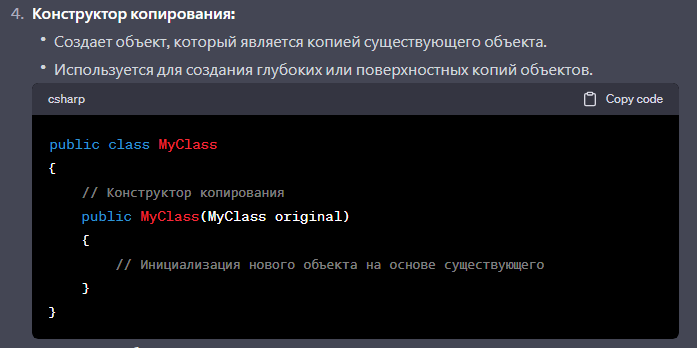
2. Конструктор с параметрами:



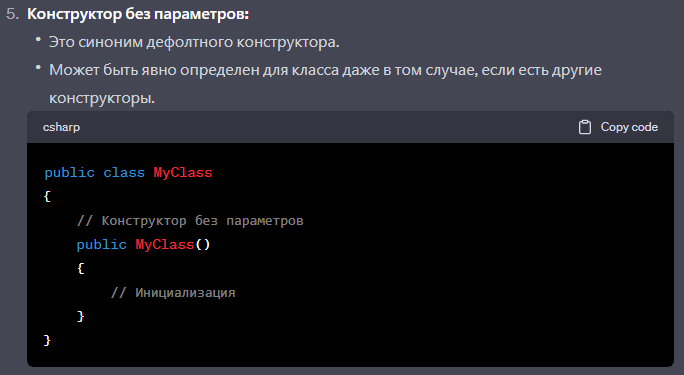
3. Статический (конструктор типа) конструктор:



4. Конструктор копирования:



5. Конструктор без параметров:



1. Работа с полями. Get и Set.

get и set - это ключевые слова в контексте использования свойств (properties) в языках программирования, таких как C# или Java. Они используются для определения доступа к значениям членов класса.

1. get:

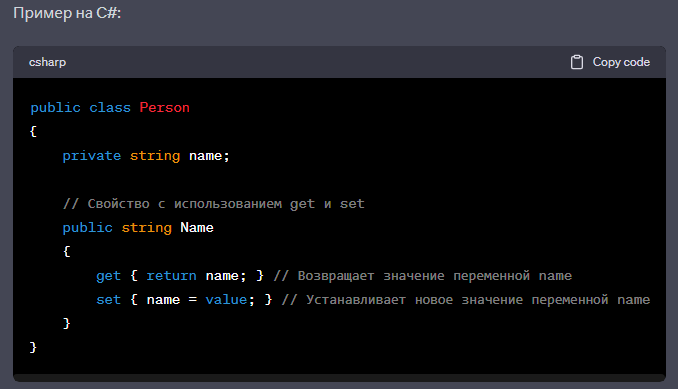
- get используется для получения значения переменной, объявленной в свойстве.

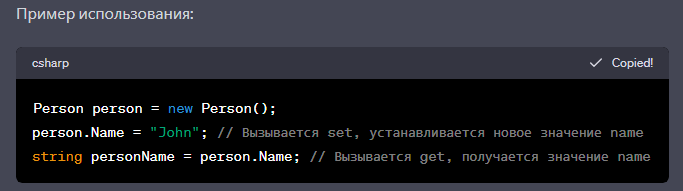
- Этот блок выполняется, когда код запрашивает значение свойства.

2. set:

- set используется для установки значения переменной, объявленной в свойстве.

- Этот блок выполняется, когда код пытается установить новое значение свойства.





Свойства с get и set обеспечивают контролируемый доступ к данным класса, позволяя добавлять логику проверки и изменения при чтении и записи значений переменных. Это является частью концепции инкапсуляции в объектно-ориентированном программировании.

1. Показать работу с конструкторами по своему варианту.
2. **По вопросам:**
3. Принципы ООП и всё о них (Виды полиморфизма: Статичный \*при компиляции\* и Динамичный \*при выполнении программы\*)

Статичный – Перегрузка методов, Перегрузка операторов

Динамический – Переопределение методов, наследование, реализация интерфейса

**1. Принципы ООП:**

- Инкапсуляция: Сокрытие деталей реализации объекта и предоставление интерфейса для взаимодействия с ним. Это позволяет изолировать внутреннюю реализацию от внешнего мира, обеспечивая более гибкое и безопасное программирование.

- Наследование: Механизм, позволяющий новому классу (подклассу) использовать свойства и методы существующего класса (суперкласса). Это способствует повторному использованию кода и иерархии классов.

- Полиморфизм: Возможность объектов разных типов обрабатываться единообразно. Полиморфизм может проявляться как перегрузка методов (одинаковые методы с разными параметрами), так и виртуальные методы.

* 1. **Назовите класс .NET, от которого наследуются все классы.**

В .NET все классы наследуются от класса System.Object.

6. Какие ключевые слова можно использовать при определении класса?

Ключевые слова при определении класса: abstract, static, sealed (без детей), partial (частичный)

6. Ключевые слова при определении класса:

- class: Используется для создания класса.

- public, private, protected: Определяют уровень доступа к членам класса.

- static: Обозначает статические члены класса.

* **sealed**: Класс не может быть производным (не имеет потомков).
* **partial**: Класс с разделенной реализацией, разделен на несколько частей в разных файлах.

**7. В чем отличие между объектом и классом?**

**Отличие между объектом и классом:**

- Класс: Это шаблон или чертеж, описывающий структуру и поведение объекта.

- Объект: Экземпляр класса, созданный на основе его определения. Реальное воплощение класса в программе.

**8. Что такое конструктор? Когда вызывается конструктор?**

Конструктор это метод, который инициализирует объект. Он вызывается при создании экземпляра класса.

Вызывается с помощью оператора new.

**9. Перечислите свойства конструктора?**

- Имя:

- Нет возвращаемого значения:

- Вызывается при создании объекта:

- Может иметь параметры:

**10. Что такое деструктор (destructor) ?**

- Специальный метод, используемый для освобождения ресурсов перед уничтожением объекта.

- Имя деструктора совпадает с именем класса, предваряется символом ~.

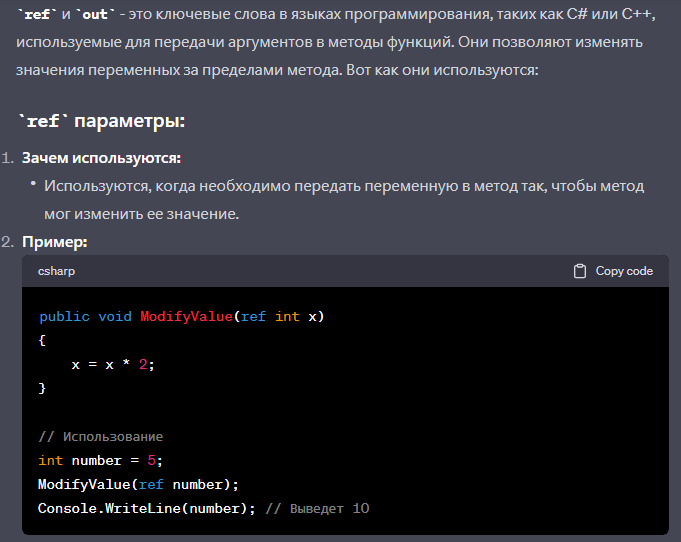
**11. Что такое this?**

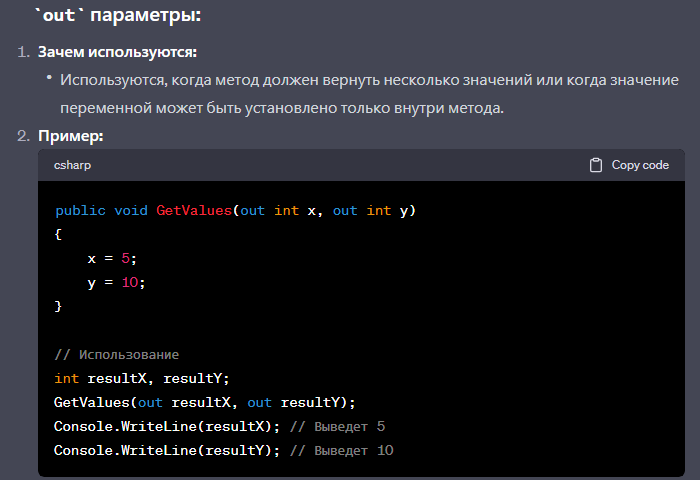
Это ключевое слово, которое ссылается на текущий экземпляр объекта.

Используется для разрешения конфликтов идентификаторов между параметрами метода/конструктора и полями/свойствами класса.

**15. Зачем и как используются ref и out параметры функции?**

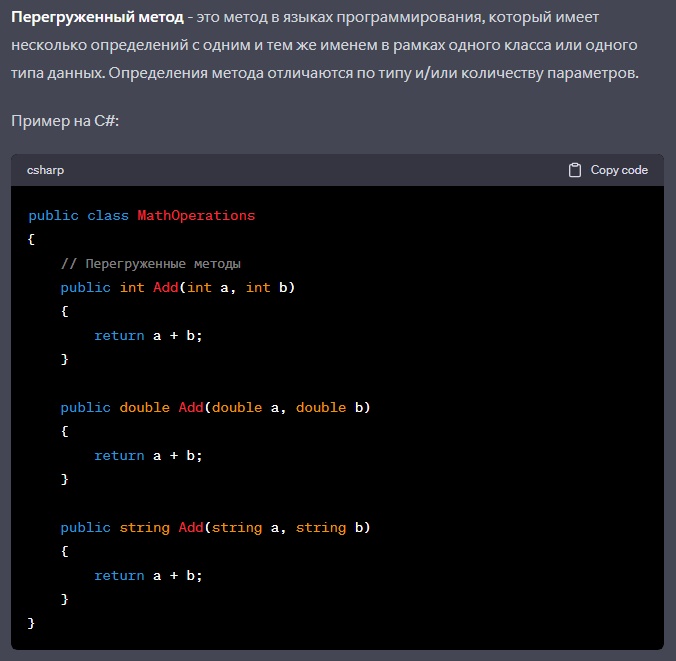
**ref** и **out** параметры функции используются для передачи параметров по ссылке. **ref** используется, когда параметр должен быть инициализирован перед передачей, а **out** используется, когда параметр должен быть инициализирован внутри метода.





**22. Что такое перегруженный метод?**

Перегруженный метод - это метод с тем же именем в одном классе, но с разными параметрами (различной сигнатурой). Перегруженные методы позволяют использовать одно и то же имя метода для разных операций.



**23. Что такое partial класс и какие его преимущества?**

Partial класс (частичный класс) - это класс, который можно определить в нескольких файлах. Все части класса объединяются компилятором в один класс. Это удобно, когда несколько разработчиков работают над одним классом

**24. Что такое анонимный тип в C#?**

Анонимный тип (anonymous type) - это тип данных в C#, который не имеет имени и определяется автоматически на основе инициализации. Он часто используется для создания объектов с набором свойств, не требующих дополнительных объявлений

**28. Какая разница между поверхностным (shallow) и глубоким (deep) копированием?**

1. Разница между **поверхностным** (shallow) и **глубоким** (deep) копированием:

**Поверхностное** копирование копирует только **ссылки** на объекты, но не их содержимое.

**Глубокое** копирование создает копии **объектов** и их содержимости, что может потребовать рекурсивного копирования подобных структур.

**29. В чем разница между равенством и тождеством объектов?**

Равенство (equality) проверяет, что два объекта имеют одинаковое состояние, но не обязательно ссылаются на один и тот же объект в памяти. Тождество (identity) проверяет, что две ссылки указывают на один и тот же объект в памяти. В C#, метод **Equals** используется для проверки равенства объектов, а оператор **==** проверяет тождество (для ссылочных типов).



**Вопрос 32 (посмотреть) а=2 b=1**

**Вопрос 34 (кол-во элементов) (от 0 до 2 параметров)**