1. **Перечислите свойства внутренних и вложенных классов?**

Вложенные:

* Автоматически private
* могут иметь модификаторы [public](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/public), [protected](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/protected), [internal](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/internal), [protected internal](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/protected-internal), [private](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/private) или [private protected](https://learn.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/private-protected)
* может получить доступ к внешним членам тиипа, даже приватным

Внешние:

1. **Что такое статический класс и какие у него свойства?**

* прямой потомок System.Object
* экземпляры создавать запрещено (нельзя использовать new)
* не должен реализовывать никаких интерфейсов (не вызвать)
* нельзя использовать в качестве поля, параметра метода или локальной переменной
* от него запрещено наследовать
* все элементы такого класса должны явным образом объявляться с модификатором static
* может иметь статический конструктор
* компилятор не создает автоматически конструктор по умолчанию

1. **Каково назначение перегрузки операторов?**

Способ объявления новых операций для типа

1. **Как используется ключевое слово operator?**

Тип operator @ (список\_параметров-операндов)

{ тело функции }

Где @ - знак перегружаемого оператора

1. **Какие операции нельзя перегружать в C#?**

* [] (но есть индексатор)
* () (можно определить новые операторы преобразования)
* +=, -=, \*=, /=, %=, &=, |=, ^=, <<=, >>= (но получаем автоматически в случае перегрузки бинарной операции)
* &&, || (логическое и/или)
* =, ., ?:,??,->,=>,f(x),as,checked,unchecked,default,delegate,is,new,sizeof,typeof

1. **Можно ли перегрузкой отменить очередность выполнения операции?**
2. **Истинно ли следующее утверждение: операция >= может быть перегружена.**

да

1. **Сколько аргументов требуется для определения перегруженной унарной операции?**

0, если локальная; 1-если дружественная (находится не в классе, но имеет доступ к прайват полям)

1. **Можно ли перегружать операцию []?**

Нет, но есть индексатор

1. **Можно ли перегружать операцию ->?**

Нет

1. **Приведите пример оператора приведения типа**

implicit operator тип ( параметр )

// неявное преобразование

explicit operator тип ( параметр )

// явное преобразование

Где параметр – тип, который преобразуют; а тип – тип, в который преобразуют



12. Что такое метод расширения? Как и где его можно использовать?

13. Пусть дан фрагмент кода определения оператора преобразования типа. Определить форму преобразования.

public static implicit operator Point2D(Point3D a)

{/\* код\*/;}

14. Выберите верное утверждение. Метод расширения может:

1) получать доступ к public членам расширяемого класса

2) получать доступ к protected членам расширяемого класса

3) получать доступ к internal членам расширяемого класса

4) быть объявлен в любом классе

5) быть без параметров

15. Выберите из списка неверное правило перегрузки операторов для C#.

1) префиксные операции ++ и – – перегружаются парами

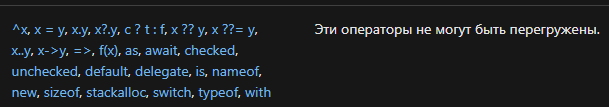
2) операции сравнения перегружаются парами: == и != ; < и >;<= и >=

3) перегруженные операции обязаны возвращать значения

4) должны объявляться как protected

5) true и false можно перегружать

5. Оператор равенства или тернарный оператор.



6. При перегрузке операторов также следует помнить, что мы не можем изменить приоритет оператора или его ассоциативность, мы не можем создать новый оператор или изменить логику операторов в типах, который есть по умолчанию в .NET.

9. a[i]

10. Нельзя

11. (int)

12. Методы расширения (extension methods) позволяют добавлять новые методы в уже существующие типы без создания нового производного класса. Эта функциональность бывает особенно полезна, когда нам хочется добавить в некоторый тип новый метод, но сам тип (класс или структуру) мы изменить не можем, поскольку у нас нет доступа к исходному коду. Либо если мы не можем использовать стандартный механизм наследования, например, если классы определенны с модификатором sealed.

13.

**List** C# — списки — представляют собой удивительно гибкий инструмент по работе с коллекциями. Одной из главных особенностей списков является возможность использовать любой тип данных.