# Вопросы

1. Назовите принципы ООП. Поясните каждый из них.

* Инкапсуляция− механизм, связывающий вместе данные и код, обрабатывающий эти данные, и сохраняющий их от внешнего воздействия и ошибочного использования

**Абстракция** подразумевает разделение и независимое рассмотрение **интерфейса** и **реализации**

* способ выделить набор наиболее важных атрибутов и методов и исключить незначимые.

Наследование− процесс, благодаря которому один объект может наследовать (приобретать) свойства от другого объекта.

Полиморфизм− способность программы идентично

использовать объекты с одинаковым

интерфейсом без информации о конкретном

типе этого объекта. Способность вызывать метод потомка через

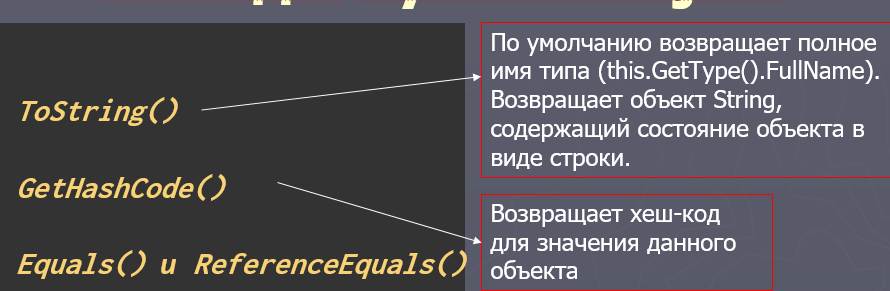
экземпляр предка

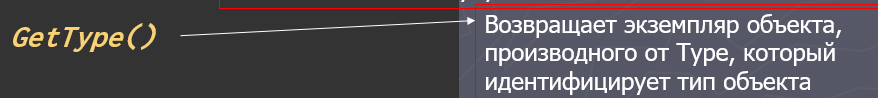
1. Назовите класс .NET, от которого наследуются все классы.

Класс object.

1. Охарактеризуйте открытые методы System.Object.

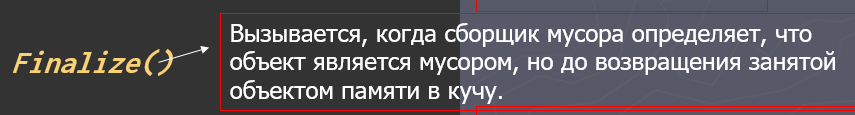
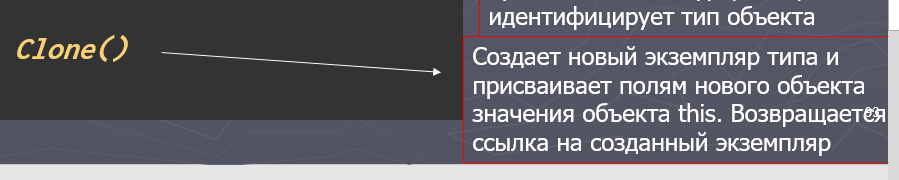
**Открытые методы** (или **public методы**) — это методы класса или объекта, которые доступны для вызова из любой части программы. Это означает, что если метод помечен как public, его можно использовать за пределами того класса, в котором он определён.





1. Охарактеризуйте закрытые методы System.Object.

**Закрытые методы** (или **private методы**) — это методы класса, которые доступны только внутри этого класса. Они **не могут быть вызваны из других классов** или объектов, даже если они созданы из этого же класса.

1. Приведите пример определения класса.

Public Class {}

1. Какие ключевые слова можно использовать при определении класса?

**class**: Основное ключевое слово для объявления класса.

**public, private, internal, protected**: Модификаторы доступа, определяющие видимость класса и его членов.

**public** – класс доступен из любого кода

**private** – класс доступен только внутри содержащего его класса

**internal** – класс доступен только в пределах текущей сборки

**protected** – доступ к классу возможен только в наследуемых классах (для членов класса).

**abstract**: (класс обязан иметь производные классы),. Абстрактные классы служат в качестве базовых классов и не могут быть напрямую созданы (т.е. нельзя создать экземпляр абстрактного класса). Вместо этого они предназначены для наследования, чтобы дочерние классы могли реализовать или расширить их функциональность.

**static**: Указывает, что класс статический и не может иметь экземпляров.

**sealed**: Указывает, что класс не может быть унаследован.

**partial** – позволяет разделять реализацию класса на несколько файлов.

1. В чем отличие между объектом и классом?

 **Класс** — это шаблон или чертеж, который описывает свойства (атрибуты) и поведение (методы) объектов. Он определяет, какие данные и функции будут у создаваемых объектов.

 **Объект** — это конкретный экземпляр класса, который имеет свои уникальные значения атрибутов. Например, если класс — это "Автомобиль", то объектом может быть конкретный автомобиль с определенными характеристиками.

1. Что такое конструктор? Когда вызывается конструктор?

Конструкторы — это специальные методы, позволяющие корректно инициализировать новый экземпляр типа.

. Конструктор вызывается автоматически при создании экземпляра класса.

1. Перечислите свойства конструктора?

имя такое же как и имя типа (класса )

2) не имеет возвращаемого значения

* 3) не наследуются
* 4) нельзя применять модификаторы virtual, new, override, sealed и abstract
* 5) для класса без явно заданных конструкторов компилятор создает конструктор по умолчанию (без параметров)
* 6) для статических классов компилятор не создает конструктор по умолчанию
* 7) может определяться несколько конструкторов, сигнатуры и уровни доступа к конструкторам обязательно должны отличаться
* 8) можно явно заставлять один конструктор вызывать другой конструктор посредством зарезервированного слова this:

1. Что такое деструктор (destructor) ?

* Деструктор — это специальный метод, который вызываться непосредственно перед окончательным уничтожением объекта системой "сборки мусора", чтобы гарантировать четкое окончание срока действия объекта.

1. Что такое this?

* Это ключевое слово указывает на нынешний объект класса. в любой нестатический метод автоматически передается скрытый параметр this

1. Что будет выведено в результате выполнения

class A {

private int \_num;

public A(int num) { Num = num; }

public int Num { get { return \_num; } set { \_num = value; } }

}

static void Main(string[] args)

{

A a = new A(5);

A b = a;

Console.WriteLine(a.Num + " " +b.Num);

a.Num = 7;

Console.WriteLine(a.Num + " " + b.Num);

}

Выведет 5 5 и 7 5

1. Какие спецификаторы доступа для класса и методов класса существуют в C#?

 p**ublic**: доступен из любого места.

 **private**: доступен только внутри своего класса.

 **protected**: доступен внутри своего класса и в производных классах.

 **internal**: доступен только внутри текущей сборки.

1. Опишите модификатор protected internal.

protected internal комбинирует два уровня доступа: доступен как в производных классах (protected), так и внутри текущей сборки (internal). Это позволяет использовать члены класса как для наследующих классов, так и для классов в одной сборке.

1. Зачем и как используются ref и out параметры функции?

 **ref**: позволяет передать параметр по ссылке, и изменения внутри метода будут отражены на оригинальном аргументе. Аргумент должен быть инициализирован перед передачей.

 **out**: также передает параметр по ссылке, но не требует предварительной инициализации. Метод должен присвоить значение выходному параметру перед завершением.

1. Приведите пример необязательных и именованных параметров метода.

Именованные: позволяет указать имя того параметра, которому присваивается его значение

( в конструкторах, индексаторах или делегатах.)

static void RedrawButton(int color ,

int type = 2 ,

int size = 4)

{ }

static void Main(string[] args)

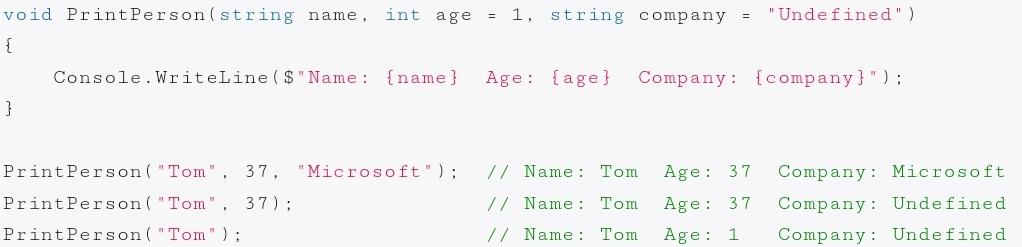
{

RedrawButton(243,size:45);

}

Необязательные: позволяет определить используемое по умолчанию значение для параметра метода

* можно применять в конструкторах, индексаторах



1. Приведите пример полей класса – статические, константные, только для чтения.

public class MyClass

{

// Статическое поле

public static int staticField = 10;

// Константное поле

public const double Pi = 3.14159;

// Поле только для чтения

public readonly string name;

// Конструктор

public MyClass(string name)

{

this.name = name; // Инициализация поля readonly

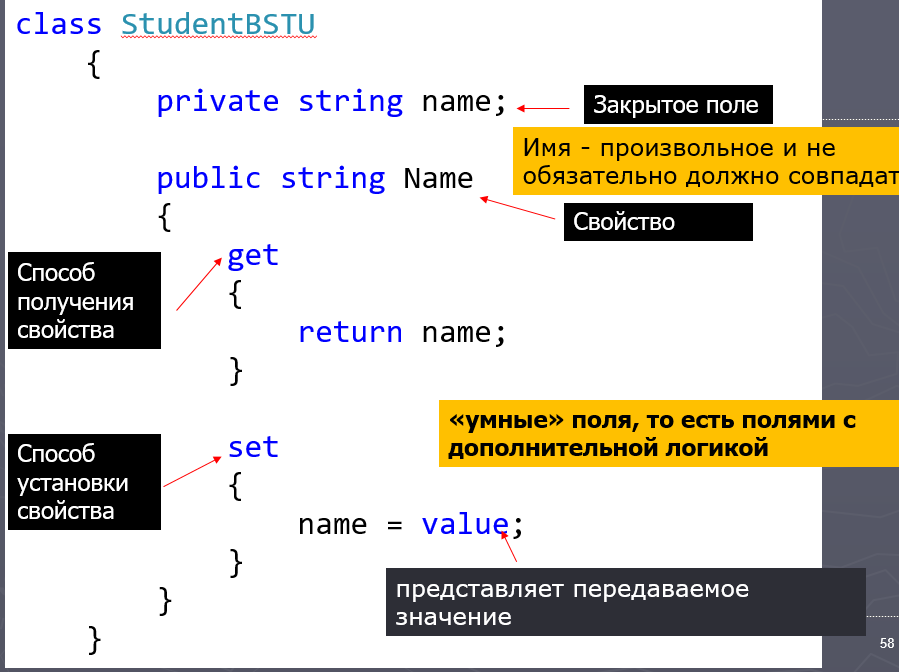
}

}

1. Приведите пример определения свойств класса. Как свойства связаны с инкапуляцией?

**Свойства – специальные методы класса, служат для организации доступа к полям класса.**

**Как правило, свойство связано с закрытым полем класса и определяет методы его получения и установки (предоставляет инкапсуляцию).**

****

1. Назовите явное имя параметра, передаваемого в метод set свойства класса?

Имя параметра, передаваемого в метод set, всегда **value**. Оно является неявным и используется для присваивания значения полю.

1. Что такое автоматические свойства?

компилятор автоматически реализует методы для правильного возвращения значения из поля и назначения значения полю

1. Что такое индексаторы класса? Какие ограничения существуют на индексатор?

Позволяют индексировать объекты таким же способом, как массив или коллекцию. Индексаторы позволяют приложению обращаться с объектом класса так, как будто он является массивом.

Ограничения на индексаторы:

1. значение, выдаваемое индексатором, нельзя передавать методу в качестве параметра ref или out
2. индексатор не может быть объявлен как static
3. Что такое перегруженный метод?

**Перегруженный метод** — это метод с тем же именем, но с разными наборами параметров (различия в типах, количестве или порядке параметров). Перегрузка позволяет вызывать одну и ту же функцию для разных типов входных данных.

1. Что такое partial класс и какие его преимущества?

**Partial-класс** в C# позволяет разделить реализацию класса на несколько файлов. Все части класса, отмеченные ключевым словом partial, компилируются в один класс.

1. Что такое анонимный тип в C#?

**Анонимные типы** в C# позволяют создавать объекты без явного объявления класса. Они используются, когда нужно временно создать объект для хранения данных без необходимости объявлять отдельный тип.

1. Для чего делают статические классы?

Статические классы предназначены для хранения статических членов и не могут быть инстанцированы. Их используют для группировки методов и свойств, которые не требуют создания экземпляра класса.

1. В чем отличие статического поля от экземплярного?

**Статическое поле** принадлежит самому классу и общее для всех его экземпляров. Значение статического поля сохраняется между вызовами методов и не изменяется при создании новых объектов класса.

**Экземплярное поле** принадлежит конкретному объекту класса, и каждый объект имеет свое собственное значение этого поля.

1. Поясните работу статических конструкторов.

не должны иметь модификатор доступа и не принимают параметров

**нельзя** использовать ключевое слово **this** для ссылки на текущий объект класса и можно обращаться только к статическим членам класса

нельзя вызвать в программе вручную.

**Они выполняются автоматически при самом первом создании объекта данного класса или при первом обращении к его статическим членам**

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

**Поверхностное копирование** создает новый объект, но копирует только значения полей, которые являются примитивными типами (например, int, bool). Если поле является ссылкой на другой объект, копируется только ссылка, а не сам объект. Это означает, что изменения в вложенном объекте будут видны в обеих копиях.

**Глубокое копирование** создает новый объект, а также рекурсивно создает копии всех вложенных объектов, на которые ссылается оригинал. Это гарантирует, что изменения в копии не повлияют на оригинал, и наоборот.

1. В чем разница между равенством и тождеством объектов?

**Равенство объектов** означает сравнение содержимого объектов, то есть их данных или свойств, чтобы понять, эквивалентны ли они по значениям.

**Тождество объектов** проверяет, ссылаются ли две переменные на один и тот же объект в памяти. Это сравнение указывает, являются ли две переменные ссылками на один и тот же экземпляр объекта.

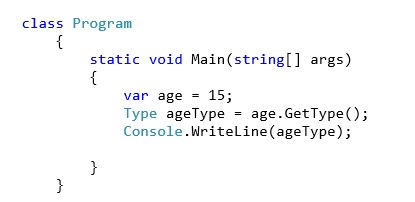
1. Что такое частичные классы и частичные методы?

**Частичные классы** и **частичные методы** — это механизм, используемый в некоторых языках программирования, таких как C#, для разделения реализации одного класса или метода на несколько файлов. Это полезно в больших проектах для улучшения организации кода.

* **Частичные классы (partial classes)** позволяют разделить определение класса на несколько файлов. Все части класса объединяются компилятором в один класс во время компиляции.

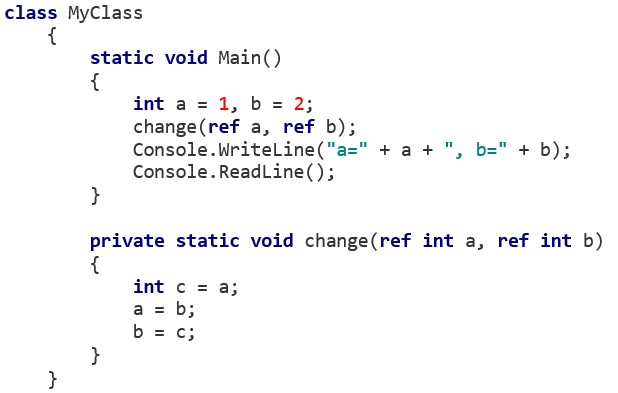
**Частичные методы (partial methods)** позволяют определить метод в одном файле и, при необходимости, реализовать его в другом.

1. Что будет выведено на консоль результате выполнения следующего кода:



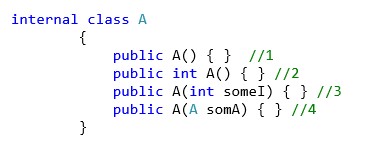
System.Int32`.

1. Что будет выведено на консоль результате выполнения следующего кода:



a=2, b=1

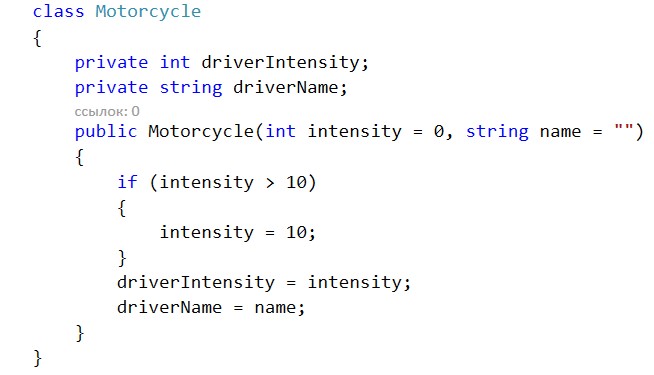
1. Пусть задан следующий класс.

 Какой из конструкторов задан неверно?

2й

Конструкторы не должны иметь возвращаемого типа, даже void. В отличие от обычных методов, они предназначены исключительно для инициализации объекта и не возвращают значений.

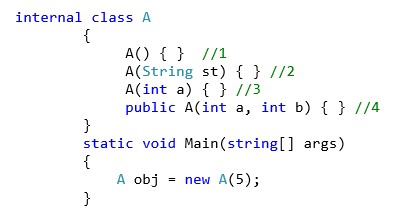
1. Пусть задан следующий класс.



Сколько аргументов может быть задано при вызове конструктора данного класса?

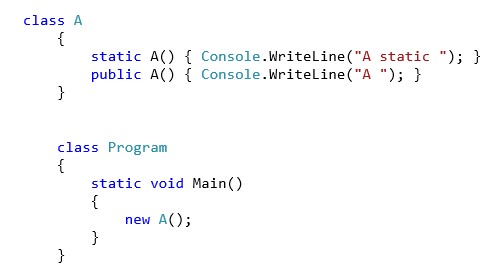
Можно задать 0 1 или 2 так как есть значения по умолчанию

1. Почему не удается создать объект класса A?



Все конструкторы по умолчанию стоят приват

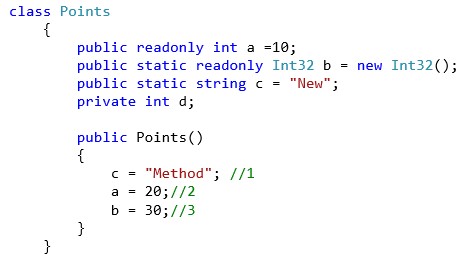
1. Что будет выведено в консоль при выполнении данной программы?



A static

A

1. Какая строка приведенного далее класса вызовет ошибку компиляции?



Поля с модификатором static readonly можно инициализировать только в момент их объявления или в статическом конструкторе класса, но не в обычном конструкторе. Поле a объявлено как public readonly int a = 10;. Поля с модификатором readonly можно инициализировать только в момент их объявления или в конструкторе. Однако, если поле уже инициализировано (как в этом случае, значением 10), его нельзя изменить в конструкторе.