1. \*\*Назовите принципы ООП. Поясните каждый из них.\*\*

\*\*Ответ\*\*:

- \*\*Инкапсуляция\*\*: объединение данных и методов в одном классе для защиты от несанкционированного доступа.

- \*\*Наследование\*\*: возможность создания нового класса на основе существующего, с перенятием его свойств и методов.

- \*\*Полиморфизм\*\*: способность объекта принимать множество форм, например, переопределение методов в наследниках.

- \*\*Абстракция\*\*: выделение ключевых характеристик объекта, сокрытие деталей его реализации.

2. \*\*Назовите класс .NET, от которого наследуются все классы.\*\*

\*\*Ответ\*\*: `System.Object`.

3. \*\*Охарактеризуйте открытые методы System.Object.\*\*

\*\*Ответ\*\*:

- `ToString()`: возвращает строковое представление объекта.

- `Equals()`: определяет, равны ли два объекта.

- `GetHashCode()`: возвращает хэш-код объекта.

- `GetType()`: возвращает тип текущего объекта.

4. \*\*Охарактеризуйте закрытые методы System.Object.\*\*

\*\*Ответ\*\*:

- `Finalize()`: вызывается перед уничтожением объекта для освобождения ресурсов.

- `MemberwiseClone()`: создает поверхностную копию текущего объекта.

5.Пример определения класса:

Класс — это пользовательский тип данных, который описывает объект, его состояние (поля) и поведение (методы). Определение класса начинается с ключевого слова class, за которым следует имя класса и его содержимое в виде полей, методов, свойств и конструкторов.

6. \*\*Какие ключевые слова можно использовать при определении класса?\*\*

\*\*Ответ\*\*: `public`, `private`, `protected`, `internal`, `abstract`, `sealed`, `static`, `partial`.

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

\*\*Ответ\*\*: Класс — это шаблон для создания объектов, а объект — это экземпляр класса.

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

\*\*Ответ\*\*: Конструктор — это метод, используемый для инициализации объекта. Он вызывается при создании нового объекта класса.

9. \*\*Перечислите свойства конструктора.\*\*

\*\*Ответ\*\*:

- Имя конструктора совпадает с именем класса.

- Конструкторы не возвращают значения.

- Может быть несколько конструкторов (перегрузка).

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

\*\*Ответ\*\*: Деструктор — это метод, вызываемый при уничтожении объекта для освобождения ресурсов. В C# используется редко, так как есть сборщик мусора.

11. \*\*Что такое this?\*\*

\*\*Ответ\*\*: `this` — это ссылка на текущий экземпляр класса, которая используется для обращения к его членам.

12. \*\*Что будет выведено в результате выполнения программы?\*\*

```csharp

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);

}

```

\*\*Ответ\*\*:

- 5 5

- 7 7

Поскольку `a` и `b` ссылаются на один и тот же объект.

13. \*\*Какие спецификаторы доступа для класса и методов класса существуют в C#?\*\*

\*\*Ответ\*\*: `public`, `private`, `protected`, `internal`, `protected internal`, `private protected`.

14. \*\*Опишите модификатор protected internal.\*\*

\*\*Ответ\*\*: Доступен внутри текущей сборки и в производных классах, даже если они находятся в другой сборке.

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

\*\*Ответ\*\*:

- `ref`: передает параметр по ссылке, требует инициализации до передачи.

- `out`: передает параметр по ссылке, не требует инициализации до передачи, но значение должно быть присвоено внутри метода.

1. Пример необязательных и именованных параметров метода:

Необязательные параметры — это параметры метода, которым можно задать значения по умолчанию, и эти значения будут использованы, если при вызове метода аргументы не переданы. Именованные параметры позволяют явно указывать имя параметра при вызове метода, что улучшает читабельность кода и позволяет передавать параметры в любом порядке.

1. Пример полей класса — статические, константные, только для чтения:

Статическое поле принадлежит классу, а не конкретному экземпляру, и его значение одинаково для всех объектов этого класса.

Константное поле задаётся при объявлении и не может быть изменено во время выполнения программы.

Поле "только для чтения" (readonly) может быть установлено либо при инициализации, либо в конструкторе, но его нельзя изменять после этого.

1. Пример определения свойств класса. Как свойства связаны с инкапсуляцией?

Свойства (properties) — это специальные методы доступа (геттеры и сеттеры), которые позволяют управлять доступом к полям класса. Они обеспечивают инкапсуляцию, так как через свойства можно ограничивать или контролировать доступ к полям: например, можно разрешить чтение, но запретить изменение значения поля напрямую.

19. \*\*Назовите явное имя параметра, передаваемого в метод set свойства класса.\*\*

\*\*Ответ\*\*: `value`.

20. \*\*Что такое автоматические свойства?\*\*

\*\*Ответ\*\*: Это свойства, для которых не нужно явно указывать поля. Компилятор автоматически создает закрытые поля для таких свойств.

```csharp

public string Name { get; set; }

```

21. \*\*Что такое индексаторы класса? Какие ограничения существуют на индексатор?\*\*

\*\*Ответ\*\*: Индексаторы позволяют обращаться к объекту класса как к массиву:

```csharp

public int this[int index] { get { return array[index]; } set { array[index] = value; } }

```

Ограничения: индексаторы не могут быть статическими и должны иметь хотя бы один параметр.

22. \*\*Что такое перегруженный метод?\*\*

\*\*Ответ\*\*: Это метод с тем же именем, но с разными параметрами (по количеству или типу).

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

\*\*Ответ\*\*: Класс, определённый в нескольких файлах с использованием ключевого слова `partial`. Удобен при работе в команде и при автоматической генерации кода.

24. \*\*Что такое анонимный тип в C#?\*\*

\*\*Ответ\*\*: Это тип, который создается без явного указания имени класса. Используется для временных объектов:

```csharp

var obj = new { Name = "John", Age = 30 };

```

25. \*\*Для чего делают статические классы?\*\*

\*\*Ответ\*\*: Статические классы используются для организации методов и данных, которые не относятся к экземплярам класса, например, математические операции (`Math`).

26. \*\*В чем отличие статического поля от экземплярного?\*\*

\*\*Ответ\*\*: Статическое поле принадлежит классу и является общим для всех экземпляров, тогда как экземплярное поле уникально для каждого объекта.

27. \*\*Поясните работу статических конструкторов.\*\*

\*\*Ответ\*\*: Статические конструкторы инициализируют статические поля и вызываются один раз при первом обращении к классу или его статическим членам.

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

\*\*Ответ\*\*:

- \*\*Поверхностное копирование\*\*: копирует только значения полей. Если поле — ссылка, копируется только ссылка, а не сам объект.

- \*\*Глубокое копирование\*\*: создаёт копию объекта, а также копирует все вложенные объекты.

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

\*\*Ответ\*\*:

- \*\*Равенство\*\*: два объекта считаются равными, если их значения совпадают (`Equals`).

- \*\*Тождество\*\*: два объекта считаются идентичными, если они ссылаются на одну и ту же область памяти (`ReferenceEquals`).

30. \*\*Что такое частичные классы и частичные методы?\*\*

\*\*Ответ\*\*: Частичные классы и методы позволяют разделять их определение между несколькими файлами. Частичные методы могут быть определены в одном месте и реализованы в другом или не реализованы вовсе, без влияния на компиляцию.

### 31. Что будет выведено на консоль в результате выполнения следующего кода:

csharp

Копировать код

class Program

{

static void Main(string[] args)

{

var age = 15;

Type ageType = age.GetType();

Console.WriteLine(ageType);

}

}

**Ответ:** На консоль будет выведено System.Int32, так как переменная age имеет значение 15, а это тип данных int. В языке C# тип int соответствует типу System.Int32.

### 32. Что будет выведено на консоль в результате выполнения следующего кода:

csharp

Копировать код

class MyClass

{

static void Main()

{

int a = 1, b = 2;

change(ref a, ref b);

Console.WriteLine("a=" + a + ", b=" + b);

Console.ReadLine();

}

private static void change(ref int a, ref int b)

{

int c = a;

a = b;

b = c;

}

}

**Ответ:** Будет выведено: a=2, b=1.  
В методе change значения переменных a и b меняются местами через временную переменную c.

### 33. Какой из конструкторов задан неверно?

csharp

Копировать код

internal class A

{

public A() { } //1

public int A() { } //2

public A(int someI) { } //3

public A(A somA) { } //4

}

**Ответ:** Неверным является конструктор под номером **2**, так как он имеет возвращаемый тип int, что делает его методом, а не конструктором. Конструкторы не должны возвращать тип данных.

### 34. Сколько аргументов может быть задано при вызове конструктора данного класса?

csharp

Копировать код

class Motorcycle

{

private int driverIntensity;

private string driverName;

public Motorcycle(int intensity = 0, string name = "")

{

if (intensity > 10)

{

intensity = 10;

}

driverIntensity = intensity;

driverName = name;

}

}

**Ответ:** Можно задать 0, 1 или 2 аргумента при вызове конструктора. Это связано с тем, что оба параметра конструктора имеют значения по умолчанию: intensity = 0 и name = "".

### 35. Почему не удается создать объект класса A?

csharp

Копировать код

internal class A

{

A() {} //1

A(String st) {} //2

A(int a) {} //3

public A(int a, int b) {} //4

}static void Main(string[] args)

{

A obj = new A(5);

}

**Ответ:** Ошибка возникает, потому что конструктор A(int a) является internal, и попытка создать объект класса через этот конструктор из статического метода Main невозможна. Необходимо использовать публичный конструктор или сделать конструктор A(int a) публичным.

### 36. Что будет выведено в консоль при выполнении данной программы?

csharp

Копировать код

class A

{

static A() { Console.WriteLine("A static "); }

public A() { Console.WriteLine("A "); }

}

class Program

{

static void Main()

{

new A();

}

}

**Ответ:** Будет выведено:

css

Копировать код

A staticA

Статический конструктор вызывается один раз при первом обращении к классу. После этого вызывается обычный (нестатический) конструктор.

1. Какая строка приведенного далее класса вызовет ошибку компиляции?

class Points

{

public readonly int a = -10;

public static readonly Int32 b = new Int32();

public static string c = "New";

private int d;

public Points()

{

c = "Method"; //1

a = 20; //2

b = 30; //3

}

}

Ответ: Ошибка компиляции возникнет на строке a = 20; //2.

**Причина**: Поле a объявлено как readonly, что означает, что его значение можно присвоить только один раз: либо при объявлении, либо в конструкторе. В данном случае значение поля a уже задано при объявлении (a = -10), поэтому попытка изменить его в конструкторе приведет к ошибке компиляции.

Строка b = 30; //3 также вызвала бы ошибку, так как b — это статическое поле readonly, и его можно изменить только в статическом конструкторе или при инициализации.