1. Чем класс отличается от структуры?

Классы поддерживают наследование, а структуры –нет.

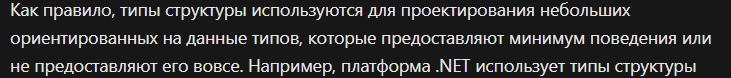
Класс – ссылочный тип данных (создаются в куче), а структуры – значимый тип данных (создаются в стеке).

Структуры, указываемые в списке параметров метода, передаются по значению (т.е. копируются), объекты классов – по ссылке. Примечание: структуру тоже можно передать по ссылке, используя модификаторы out и ref.

Чем больше будем использовать структуры вместо маленьких классов, тем менее затратные по ресурсам будет использование памяти.

Так же как и классы, структуры могут иметь поля, методы, конструкторы.

Классы могут иметь методы и свойства по умолчанию private, в то время как все члены структуры по умолчанию public.



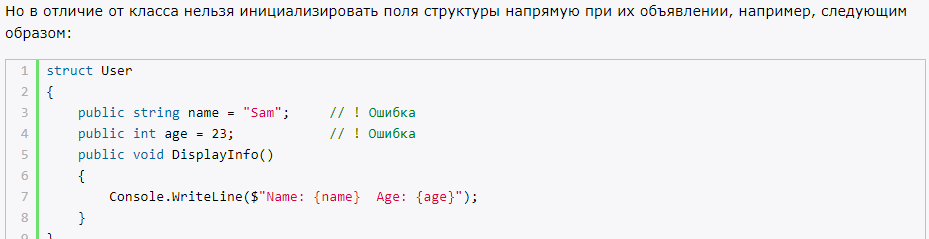
2. Что может и чего не может быть в структуре?

В структуре можно иметь следующие члены:

1. Поля (fields), которые представляют данные.
2. Методы (methods), которые выполняют операции над данными.
3. Свойства (properties), которые предоставляют доступ к данным.
4. Конструкторы (constructors), которые инициализируют структуру.
5. События (events), которые позволяют реагировать на действия.
6. Индексаторы (indexers), которые позволяют обращаться к данным, как к массиву.
7. Операторы преобразования (conversion operators), которые позволяют выполнять явные или неявные преобразования типов.

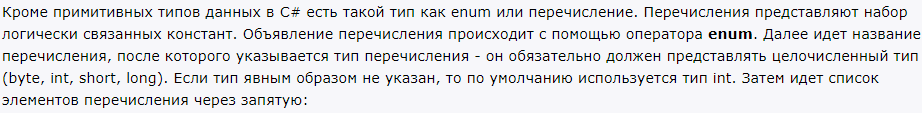
Нельзя в структуре:

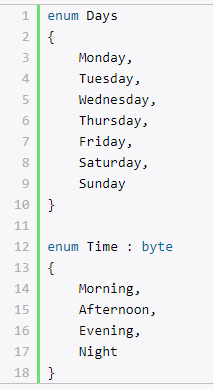
1. Определять виртуальные и абстрактные методы.
2. Использовать наследование от других структур или классов

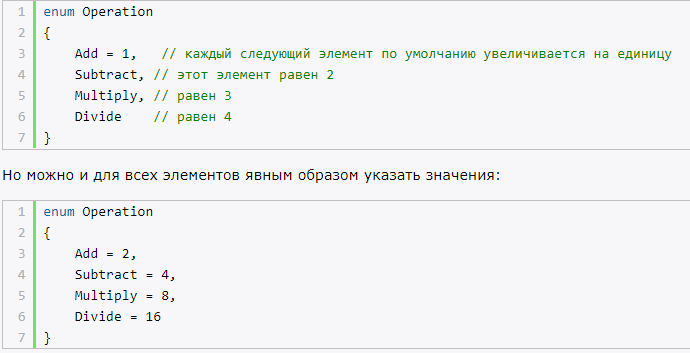


3. Что такое перечисление? Приведите пример определения и использования перечисления

Перечисление – тип данных, который представляет набор именованных целых констант. Они часто используются для создания семантических списков значений.







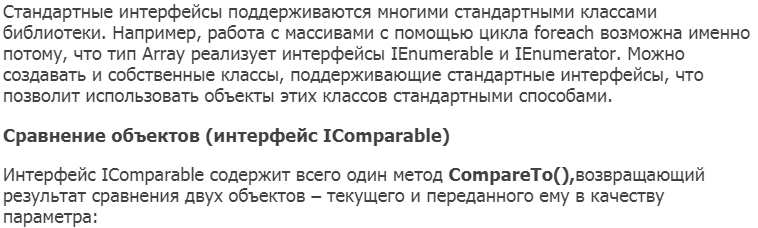
4. Перечислите и поясните стандартные интерфейсы .Net?

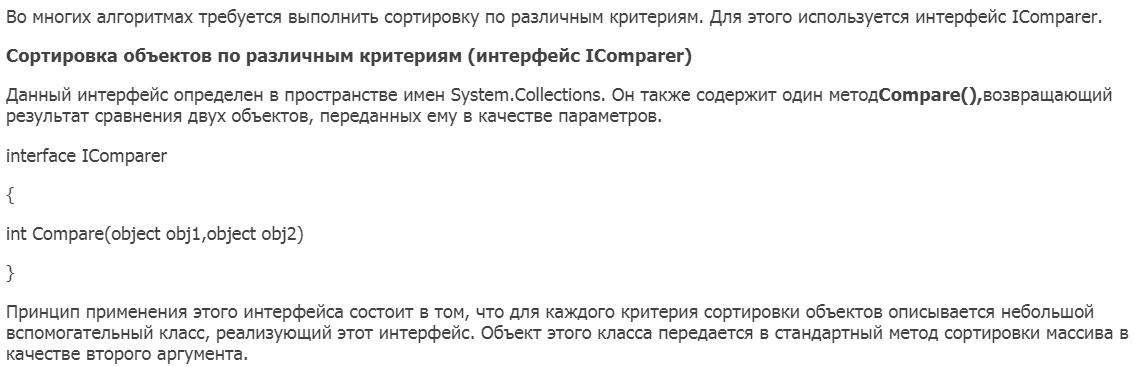
* IEnumerable: Позволяет перечислять элементы коллекции.
* IComparable: Позволяет сравнивать объекты для сортировки.
* IDisposable: Используется для освобождения ресурсов.
* IQueryable: Используется для создания запросов к источникам данных.
* IList: Предоставляет методы для работы с списками.
* IDictionary: Предоставляет методы для работы с словарями.
* INotifyPropertyChanged: Используется для уведомления об изменениях в свойствах объекта.

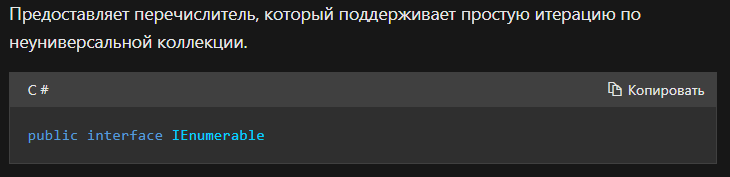
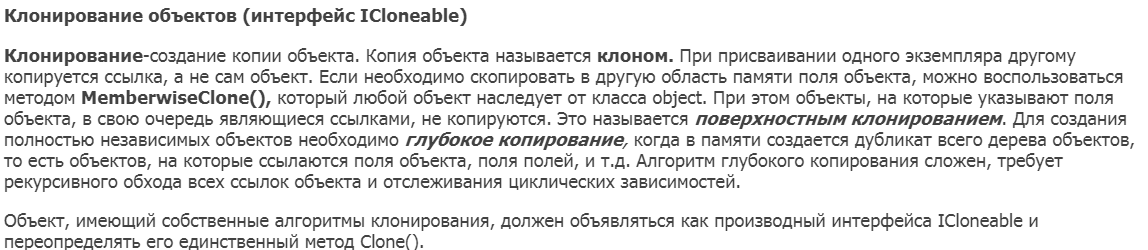
5. Как используется интерфейс IComparable?

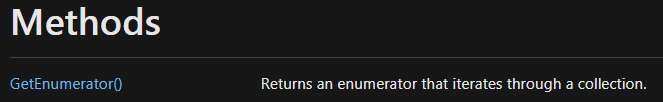
Интерфейс IComparable используется для сравнения объектов. Классы, реализующие этот интерфейс, должны определить метод CompareTo, который сравнивает текущий объект с другим объектом и возвращает результат сравнения.

6. Как используется интерфейс ICloneable?







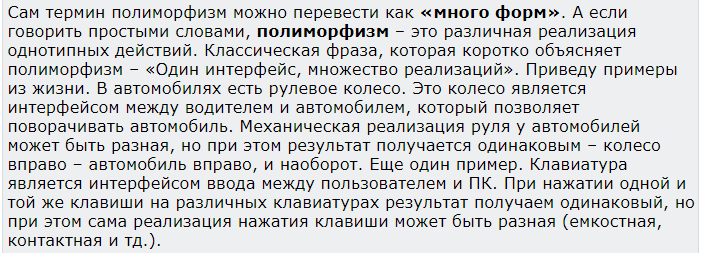


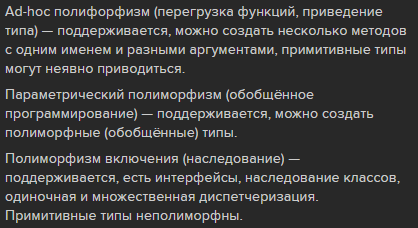
Интерфейс ICloneable используется для создания копии объекта. Классы, реализующие этот интерфейс, должны определить метод Clone, который возвращает копию текущего объекта.

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

Полиморфизм времени выполнения (runtime polymorphism): Он проявляется при использовании виртуальных методов и переопределения методов в производных классах. Пример: вызов виртуального метода Draw у разных объектов, которые могут иметь разную реализацию этого метода.

Полиморфизм времени компиляции (compile-time polymorphism): Он проявляется при перегрузке методов с разными параметрами. Пример: перегрузка метода Calculate с разными типами параметров.







8. Зачем в классе определяют виртуальные методы?

Чтобы их можно было **переопределить в производном классе**. Сам он реализован в базовом классе. Это полезно, когда вам нужно изменить специфическое поведение в производных классах, при этом сохраняя основное общее поведение из базового класса.

9. Как сделать запрет переопределения методов?

Добавить ключевое слово **sealed** в определении метода в базовом классе.