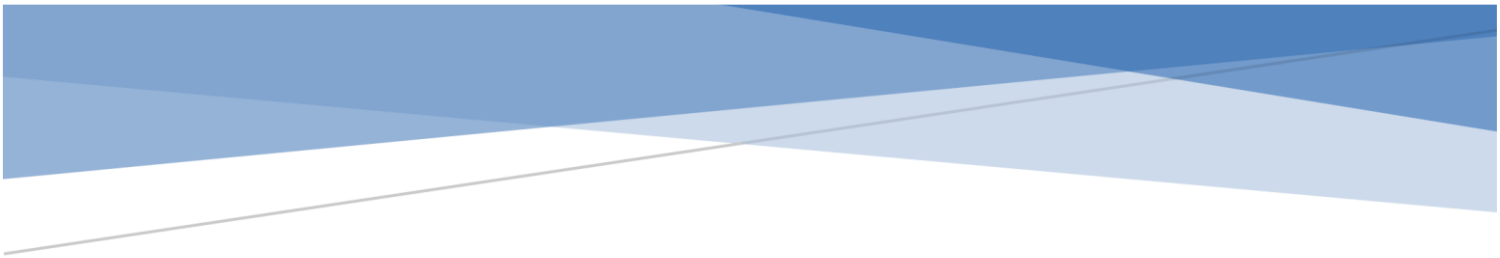

INTERFACES
ENUMS

Checklist

Olga Smolyakova

Ответьте на вопросы, приведенные ниже. Нарисуйте рисунки, позволяющие вам объяснить ответы.

Как вы думаете, что лучше использовать наследование или интерфейсы?	Полагаю, что всему свое место. В некоторых ситуациях лучше использовать наследование, а в некоторых интерфейсы. Если классы имеют схожую функциональность, но не связаны напрямую друг с другом в родственных отношениях, то интерфейс будет хорошим способом добавить эту функцию в классы без дублирования функций между ними.
Что (и с какими атрибутами) может содержать интерфейс.	Поля интерфейса по умолчанию являются final static. Все методы по умолчанию открыты (public). Ключевое слово interface используется для объявления интерфейса. Вместе с абстрактными методами интерфейс в Java может содержать константы, обычные методы, статические методы и вложенные типы.
Определите все возможности интерфейсной ссылки.	Интерфейсные ссылки должны ссылать на объекты классов, реализующих данный интерфейс. Через интерфейсную ссылку можно вызвать только методы определенные с интерфейсе.
Укажите, что Java не позволяет делать с интерфейсом.	В Java нельзя создать экземпляр интерфейса с помощью оператора new. Интерфейсные ссылки не могут ссылаться на объекты классов, которые не реализуют этот интерфейс. Через интерфейсную ссылку нельзя вызвать методы, которые не определены в интерфейсе.

Если Square – это интерфейс, то правомочна ли запись Square.class?	да
Расскажите, что такое поверхностное и глубокое копирование (клонирование) объектов? Для чего в Java используется интерфейс Clonable? Определите правила реализации этого интерфейса.	Поверхностное копирование копирует настолько малую часть информации, насколько это возможно. По умолчанию, клонирование в Java является поверхностным, т.е. Object class не знает о структуре класса, которого он копирует. При клонировании, JVM делает такие вещи: 1) если класс имеет только члены примитивных типов, то будет создана совершенно новая копия объекта и возвращена ссылка на этот объект. 2) если класс содержит не только члены примитивных типов, а и любого другого типа класса, тогда копируются ссылки на объекты этих классов. Следовательно, оба объекта будут иметь одинаковые ссылки. Глубокое копирование дублирует все. Глубокое копирование — это две коллекции, в одну из которых дублируются все элементы оригинальной коллекции. Мы хотим сделать копию, при которой внесение изменений в любой элемент копии не затронет оригинальную коллекцию. Класс Object определяет метод clone(), который создает копию объекта. Чтобы экземпляр класса можно было клонировать, необходимо переопределить этот метод и реализовать интерфейс Cloneable. Интерфейс Clonable - это интерфейс маркер, он не содержит ни методов, ни переменных. Интерфейсы маркер просто определяют поведение классов. Object.clone() выбрасывает исключение CloneNotSupportedException при попытке клонировать объект не реализующий интерфейс Cloneable. Метод clone() в родительском классе Object является protected, поэтому желательно переопределить его как public. Реализация по умолчанию метода Object.clone() выполняет неполное/поверхностное (shallow) копирование либо же нужно написать реализацию метода clone полностью.
Укажите, когда следует использовать интерфейс Comparable, а когда Comparator? Определите правила реализации этих интерфейсов.	Интерфейс Comparable содержит один единственный метод int compareTo(E item), который сравнивает текущий объект с объектом, переданным в качестве параметра. Если этот метод возвращает отрицательное число, то текущий объект будет располагаться перед тем, который передается через параметр. Если метод вернет положительное число, то, наоборот, после второго объекта. Если метод возвратит ноль, значит, оба объекта равны. Если в классе не был реализован Comparable, либо реализован, но нас не устраивает его функциональность, и мы хотим ее переопределить, то есть более гибкий способ, предполагающий применение интерфейса Comparator<E>. Интерфейс Comparator содержит ряд методов, ключевым из которых является метод compare(). Метод compare также возвращает числовое значение - если оно отрицательное, то объект a предшествует объекту b, иначе - наоборот. А если метод возвращает ноль, то объекты равны.

Что представляет собой перечисления в Java и чем по сути является каждый элемент перечисления.	Enum — специальный класс. Но он специально «заточен» на создание некоторого ограниченного круга значений. Перечисления не могут иметь наследников, сами в свою очередь наследуются от java.lang.Enum и реализуют java.lang.Comparable (могут быть сортированы) и java.io.Serializable. Каждый элемент перечисления является константой.

Как вы думаете, можно ли (а если можно, то нужно ли) в перечислении переопределять методы equals(), hashCode() и toString()	В перечислениях невозможно переопределить equals() и hashCode() потому, что они определены как final в классе Enum, который неявно является базовым классом всех типов enum. Сравнение бессмысленно, так как существует один экземпляр каждого значения enum. Возмодно переопределить toString(), так как по умолчанию он печатает то же самое , что и name().