# Worksheet: Unmodifiable Collection Decorator

Besides the synchronization decorators, the collection framework also provides decorators that assert that a decorated collection cannot be changed. These decorators can be created by one of the following static methods declared in class `Collections`.

```
public static <T> Collection<T> unmodifiableCollection(Collection<? extends T> c)
public static <T> Set<T> unmodifiableSet(Set<? extends T> c)
public static <T> SortedSet<T> unmodifiableSortedSet(SortedSet<T> s)
public static <T> List<T> unmodifiableList(List<? extends T> list)
```

These methods return a "read-only" view of the collection that is passed when one of these methods is invoked. Read operations access the original collection, and operations that would change the collection throw an `UnsupportedOperationException`.

*Task:*

Implement the following method `unmodifiableCollection`:

```
public static <T> Collection<T> unmodifiableCollection(Collection<T> c)
```

This method should return a collection decorator that throws an `UnsupportedOperationException` whenever a method is called that could potentially change the collection. This also holds for operations invoked on the iterator returned by this decorator.

In this week's project, you will find in package `patterns.decorator.util` a program fragment and a test class.

*Remarks:*

- We intentionally simplified the signature of the method `unmodifiableCollection` (regarding generics), because the focus is on decorators and not on generics.
- In case you work with Eclipse: The code can be generated almost exclusively with operations from the Source-menu provided by Eclipse.