**Iterator** is a behavioral design pattern that lets you traverse elements of an aggregate object without exposing its underlying representation (list, stack, tree, etc.).

**Pros and Cons**

* *Single Responsibility Principle*. Make code look cleaner by separating collection from bulky traversal algorithms.
* *Open/Closed Principle.* You can add new functionality with "open for extension, closed for modification".
* You can iterate over the same collection in parallel because each iterator object contains its own iteration state.
* For the same reason, you can delay an iteration and continue it when needed.
* Iterator provide a uniform interface for traversing different aggregate structures (that is, to support polymorphic iteration).
* Applying the pattern can be an overkill if your app only works with simple collections.
* Using an iterator may be less efficient than going through elements of some specialized collections directly

**How to Implement**

1. Declare the iterator interface. It must have a method for fetching the next element from a collection. But for the sake of convenience you can add a couple of other methods to provide different ways to traverse a collection, such as fetching the previous element, tracking the current position, and checking the end of the iteration.
2. Declare the collection interface and describe a method for fetching iterators. The return type should be equal to that of the iterator interface. You may declare similar methods if you plan to have several distinct groups of iterators.
3. Implement concrete iterator classes for the collections that you want to be traversable with iterators. An iterator object must be linked with a single collection instance. Usually, this link is established via the iterator’s constructor.
4. Implement the collection interface in your collection classes. The main idea is to provide the client with a shortcut for creating iterators, tailored for a particular collection class. The collection object must pass itself to the iterator’s constructor to establish a link between them.
5. Go over the client code to replace all of the collection traversal code with the use of iterators. The client fetches a new iterator object each time it needs to iterate over the collection elements.
6. Since each collection class has its own its underlying representation, we need to create appropriate iterators for each class. We could use a factory method to let client to ask the collection for the appropriate iterator