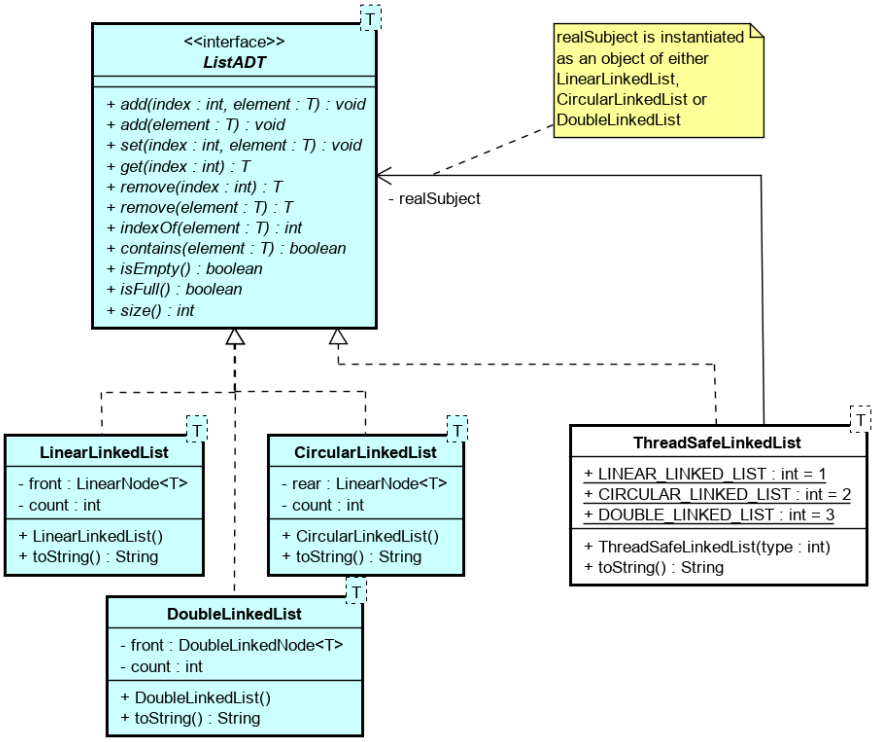
Exercise: ThreadSafeList – Proxy design pattern

Implement the proxy design pattern shown. Class `ThreadSafeLinkedList` implements `ListADT` and has an instance variable of type `ListADT`. In the constructor, the instance variable is instantiated as one of the three subclasses (real subject's) `LinearLinkedList`, `CircularLinkedList` or `DoubleLinkedList`. Because the implementation is thread safe, all methods are **synchronized**.

(A `LinearLinkedList` is most efficient when adding and removing from the front, like a *Stack*, while the other two also are efficient when adding at the rear end and removing from the front, like a *Queue*)

Interface `ListADT` and classes `LinearLinkedList`, `CircularLinkedList` and `DoubleLinkedList` (and used classes `LinearNode` and `doubleLinkedNode`) are all given here:

- MyCollection-1.1.jar: http://ict-engineering.dk/jar/MyCollection-1.1.jar
- javadoc for MyCollection: http://ict-engineering.dk/javadoc/MyCollection/



Note that the interface and all classes are generic, i.e. `ThreadSafeLinkedList` is declared this way:

```
public class ThreadSafeLinkedList<T> implements ListADT<T>
{
    // …
}
```

Test your solution in a class with a `main` method in which you create a list of `String`'s. Add a few strings and print out the full list.