1. Implement custom iterator

    a. Extend MyList interface (that was implemented during the previous hometask - https://docs.google.com/document/d/1g5GRLOU4XRDCIp50n_-Dmnok-2EdoDTIAQdVm6XyBLo/edit?usp=sharing - task #3) with Iterable interface like this:

    interface MyList extends Iterable<Object>

    b. In class DefaultMyList implements iterator() method

    ```
    public Iterator<Object> iterator() {
            return new IteratorImpl();
    }
    ```

    c. Implement inner class that would implement Iterator interface

    ```
    private class IteratorImpl implements Iterator<Object> {
    public boolean hasNext() {
    // returns true if the iteration has more elements
    // …
    }

    public Object next() {
    // returns the next element in the iteration
    // …
    }

    public void remove() {
    // removes from the underlying collection the last element returned by this iterator
    // …
    }
    }
    ```

    d. Remove method should throw IllegalStateException in case method remove was called without calling ‘next()’ method. Or in case it was called two times in a row.

    Tech note: to throw IllegalStateException wright the next code

    throw new IllegalStateException();

    e. In case there is no next element - NoSuchElementExcpetion should be thrown.

    Tech note: to throw NoSuchElementException wright the next code

throw new NoSuchElementException();

f. Iterator should work with Integers or String types

2. Implement custom list iterator

   a. Declare the interface of ListIterator like this

      interface ListIterator extends Iterator<Object> { // java.util.Iterator
      boolean hasPrevious(); // returns true if this list iterator has more elements when traversing the list in the reverse direction

      Object previous(); // returns the previous element in the list and moves the cursor position backwards

      void set(Object e); // replaces the last element returned by next or previous with the specified element

      void remove(); // removes from the list the last element that was returned by next or previous
      }

   b. Methods set() or remove() might be invoked only after the invocation of the next() or previous(). In other case - IllegalStateException is thrown

   c. Declare ListIterable interface

      interface ListIterable {
      ListIterator listIterator();
      }

   d. Add implementation of ListIterable to DefaultMyList class

      class DefaultMyList implements MyList, ListIterable {...}

   e. Add method to DefaultMyList class

      public ListIterator listIterator() {
      return new ListIteratorImpl();
      }

   f. Create inner Class ListIteratorImpl

      private class ListIteratorImpl extends IteratorImpl implements ListIterator {
      // IMPLEMENT ALL METHODS HERE

```
}
```

g. Implement all methods of custom ListIterator interface in the inner class.

h. Commit with changes to check only files that were changed (solution) - https://github.com/AndriiPiatakha/learnit_java_core/commit/a3fedfdc067b969bcfdd1159956d14ec0b3b0e6f