Lab 7: Equality

Software provided:

- Interfaces `ISelector`, `IComparator`, `IRange`

- Classes that implement lists of objects.

- Classes that represent three different kinds of audio recordings; interfaces that describe two slightly different kinds of stacks; and two different implementations of each kind of stack.

  File `Lab7.java`

The goals are

- Learn how to override the `equals` method that is defined in the `Object` class
- Learn how to implement an `equals` method that can compare two entirely different implementations of an abstract data type
- Learn how iterators make it easier to work with different implementations of an abstract data type

Details

- **Overriding the `equals` method**
  - Open the file `Lab7.java`.
  - Read the definitions of the `ATune`, `MIDI`, `MP3`, and `WAV` classes.
  - Add a `public boolean equals (Object x)` method to each of the `MIDI`, `MP3`, and `WAV` classes. Your methods will need to compare the `bits` arrays as well as the title and artist fields.

- **Comparing two different implementations**
  - Classes `ListStack0` and `ArrayStack0` both implement the `IStack0` interface. Study their code.

  - Two stacks are equal if they contain the same elements in the same order. This means that two instances of two different implementations of the `IStack0` interface may be equal. Think about the problem of comparing an instance of one of these classes to an instance of the other.
  - Hint: the `equals` method will need to cast its argument to a local variable of type `IStack0`, and use only the methods whose existence is guaranteed by the `IStack0` interface. Since the state of that stack will be changed by the only methods that allow its entire state to be examined, the `equals` method will have to keep track of enough information to restore the original state of that stack at the end of the comparison.
  - Add a `public boolean equals (Object x)` method to both the `ListStack0` and the `ArrayStack0` classes, making sure that an instance of one of these classes can compare itself to any object that implements the `IStack0` interface.

- **Using iterators to test for equality**
  - Study the code for the `equals` method that appears within the `AList` class in `AList.java`.
  - Back in `Lab7.java`, observe how the `IStack` interface extends the `IStack0` interface by adding an `iterator` method, and how the `ListStack` and `ArrayStack` classes extend `ListStack0` and `ArrayStack0` by adding this method.
  - Add a `public boolean equals (Object x)` method to both the `ListStack` and the `ArrayStack` classes, making sure that an instance of one of these classes can compare itself to any object that implements the `IStack` interface. (Hint: use iterators!)