# Section 4: Lists

## Abstract Data Types

- Does not dictate how the data is organized but dictates the operations you can perform on the data
- Concrete data structure is usually a concrete class
- ADT is usually an interface

## Array Lists

- Have an array backing, therefore it has the same problems as arrays for shifting, deleting, etc.
- Uses the **List** interface

**https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/ArrayList.html**

## Vectors

- Vector is thread safe, meaning its ok to use it from different threads without having to synchronize, whereas ArrayList is not
- If a thread safe implementation is not needed, it is recommended to use ArrayList
- Vector also implements list therefore, when using it all that needs to change is the ArrayList keyword, to Vector "new **ArrayList**<>()"

**https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Vector.html**

## Singly Linked Lists

- Contains a node that holds the current object, as well as a reference to the next object

## Doubly Linked Lists

- Contains a node that holds the current object, as well as a reference to the previous and to the next object

## Java JDK Linked List Class

**https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/LinkedList.html?is-external=true**