Sorting
Mechanics of Sorting in Java

- Java has a built-in way to sort arrays
  \[\text{Arrays.sort(myArray);}\]

- Note that this is a static methods in the \text{Arrays} class, \textbf{NOT} an instance method!
  \[\text{NOT: myArray.sort();}\]
Sorting Example

- \textit{temperatureArray:}

\begin{center}
\begin{tabular}{cccccc}
51.7 & 33.9 & 21.6 & 62.1 & 59.0 & 44.5 \\
\end{tabular}
\end{center}

- After \textit{Arrays.sort(temperatureArray):}

\begin{center}
\begin{tabular}{cccccc}
21.6 & 33.9 & 44.5 & 51.7 & 59.0 & 62.1 \\
\end{tabular}
\end{center}
Types of things you can sort

- The `Arrays.sort()` method applies to arrays, but arrays have types.
  - E.g., `int[]`, `double[]`, `Object[]`, `Point[]`

- `Arrays.sort()` can sort:
  - any primitive array type
  - any Object array type that implements the `Comparable<T>` interface (to be explained later).
The Comparable<T> Interface

- The interface declares one public method:
  ```java
  public int compareTo(T other);
  ```

- What does it mean for a class to implement this interface?
  1. The class must declare that it implements it
     e.g.: `public class BankAccount implements Comparable<BankAccount>`
  2. The class must actually do the dirty work of implementing it
Example

public class BankAccount

    implements Comparable<BankAccount>
{

    private int balance = 0;

    ...

    public int compareTo(BankAccount other) {
        if(other==null) {
            return -1; // null objects less than
            // everything else
        }

        return other.balance - this.balance;
    }

}
Example, continued

```java
BankAccount ba1 = new BankAccount(100);
BankAccount ba2 = new BankAccount(10000);

System.out.println(ba1.compareTo(ba2));
System.out.println(ba2.compareTo(ba1));
```

Output:
99900
-99900