"He's off the map!"
- Stan (Mark Ruffalo) *Eternal Sunshine of the Spotless Mind*

**Data Structures**
- More than arrays and lists
- Write a program to count the frequency of all the words in a file.

---

**Performance using ArrayList**

<table>
<thead>
<tr>
<th>Title</th>
<th>Size (kb)</th>
<th>Total Words</th>
<th>Distinct Words</th>
<th>Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>small sample</td>
<td>0.6</td>
<td>89</td>
<td>25</td>
<td>0.001</td>
</tr>
<tr>
<td>2BR02B</td>
<td>34</td>
<td>5,638</td>
<td>1,975</td>
<td>0.051</td>
</tr>
<tr>
<td>Alice in Wonderland</td>
<td>120</td>
<td>29,460</td>
<td>6,017</td>
<td>0.741</td>
</tr>
<tr>
<td>Adventures of Sherlock Holmes</td>
<td>581</td>
<td>107,533</td>
<td>15,213</td>
<td>4.144</td>
</tr>
<tr>
<td>2008 CIA Factbook</td>
<td>10,030</td>
<td>1,330,100</td>
<td>74,042</td>
<td>173.000</td>
</tr>
</tbody>
</table>

---

**Order?**
- Express change in size as factor of previous file

<table>
<thead>
<tr>
<th>Title</th>
<th>Size</th>
<th>Total Words</th>
<th>Distinct Words</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>small sample</td>
<td>0.6x</td>
<td>89</td>
<td>25</td>
<td>0.001x</td>
</tr>
<tr>
<td>2BR02B</td>
<td>57x</td>
<td>63x</td>
<td>79x</td>
<td>51x</td>
</tr>
<tr>
<td>Alice in Wonderland</td>
<td>3.5x</td>
<td>5.2x</td>
<td>3.0x</td>
<td>14.5x</td>
</tr>
<tr>
<td>Adventures of Sherlock Holmes</td>
<td>4.8x</td>
<td>3.7x</td>
<td>2.5x</td>
<td>6.0x</td>
</tr>
<tr>
<td>2008 CIA Factbook</td>
<td>17x</td>
<td>12.3x</td>
<td>5x</td>
<td>42x</td>
</tr>
</tbody>
</table>

O(Total Words * Distinct Words) ??
Clicker Question

- Given 3 minutes for the 2008 CIA Factbook with 1,330,100 total words and 74,042 distinct words, how long for 1,000x total words and 100x distinct words?
  
  A. an hour  
  B. a day  
  C. a week  
  D. a month  
  E. half a year

A Faster Way - Maps

- Also known as:
  - table, search table, dictionary, associative array, or associative container

- A data structure optimized for a very specific kind of search / access

- In a map we access by asking "give me the value associated with this key."

- Recall, in the ArrayList example we did not count the number of occurrences of each word

Why So Slow??

- Write a contains method for an array based list

  ```java
  public boolean contains(E target) {
      // implementation
  }
  ```

Keys and Values

- Dictionary Analogy:
  - The key in a dictionary is a word: foo
  - The value in a dictionary is the definition: First on the standard list of metasyntactic variables used in syntax examples

- A key and its associated value form a pair that is stored in a map

- To retrieve a value the key for that value must be supplied
  - A List can be viewed as a Map with integer keys
More on Keys and Values

- Keys must be unique, meaning a given key can only represent one value
  - but one value may be represented by multiple keys
  - like synonyms in the dictionary.
  
Example:
- factor: n. See coefficient of X
- factor is a key associated with the same value (definition) as the key coefficient of X

The Map Interface in Java

- void clear()
  - Removes all mappings from this map (optional operation).
- boolean containsKey(Object key)
  - Returns true if this map contains a mapping for the specified key.
- boolean containsValue(Object value)
  - Returns true if this map maps one or more keys to the specified value.
- Set<K> keySet()
  - Returns a Set view of the keys contained in this map.

The Map Interface Continued

- V get(Object key)
  - Returns the value to which this map maps the specified key.
- boolean isEmpty()
  - Returns true if this map contains no key-value mappings.
- V put(K key, V value)
  - Associates the specified value with the specified key in this map

The Map Interface Continued

- V remove(Object key)
  - Removes the mapping for this key from this map if it is present
- int size()
  - Returns the number of key-value mappings in this map.
- Collection<V> values()
  - Returns a collection view of the values contained in this map.
### Results with HashMap

<table>
<thead>
<tr>
<th>Title</th>
<th>Size (kb)</th>
<th>Total Words</th>
<th>Distinct Words</th>
<th>Time List</th>
<th>Time Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>small sample</td>
<td>0.6</td>
<td>89</td>
<td>25</td>
<td>0.001</td>
<td>0.0008</td>
</tr>
<tr>
<td>2BR02B</td>
<td>34</td>
<td>5,638</td>
<td>1,975</td>
<td>0.051</td>
<td>0.0140</td>
</tr>
<tr>
<td>Alice in Wonderland</td>
<td>120</td>
<td>29,460</td>
<td>6,017</td>
<td>0.741</td>
<td>0.0720</td>
</tr>
<tr>
<td>Adventures of Sherlock Holmes</td>
<td>581</td>
<td>107,533</td>
<td>15,213</td>
<td>4.144</td>
<td>0.2500</td>
</tr>
<tr>
<td>2008 CIA Factbook</td>
<td>10,030</td>
<td>1,330,100</td>
<td>74,042</td>
<td>173.000</td>
<td>4.0000</td>
</tr>
</tbody>
</table>

### Order?

<table>
<thead>
<tr>
<th>Title</th>
<th>Size</th>
<th>Total Words</th>
<th>Distinct Words</th>
<th>Time List</th>
<th>Time Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>small sample</td>
<td>0.6x</td>
<td>89x</td>
<td>25x</td>
<td>0.001x</td>
<td>0.0008x</td>
</tr>
<tr>
<td>2BR02B</td>
<td>57x</td>
<td>63x</td>
<td>79x</td>
<td>51x</td>
<td>18x</td>
</tr>
<tr>
<td>Alice in Wonderland</td>
<td>3.5x</td>
<td>5.2x</td>
<td>3.0x</td>
<td>14.5x</td>
<td>5x</td>
</tr>
<tr>
<td>Adventures of Sherlock Holmes</td>
<td>4.8x</td>
<td>3.7x</td>
<td>2.5x</td>
<td>5.6x</td>
<td>3.5x</td>
</tr>
<tr>
<td>2008 CIA Factbook</td>
<td>17x</td>
<td>12.3x</td>
<td>5x</td>
<td>42x</td>
<td>16x</td>
</tr>
</tbody>
</table>

O(Total Words)?