PFTD and PFTW

- Review hashing from intuitive viewpoint, from general viewpoint, from Java 8 viewpoint
  - Try to isolate dependencies on Java, but knowledge of Java is really important
- Understand from examples how to use Maps in Java
  - Collections Hierarchy
  - Java 8 opportunities
- Toward the Markov assignment
  - Part I will be Git specific

Key Ideas in Hashing

- Every object has its own idea of where it belongs
  - Ask not what you can do to an object, ...
  - Where do you belong? What's your number?
- In locker? A small ArrayList, ...
  - Why is it small?

Hashing details?

- Every Java object has a value, call .hashCode()
  - Should respect (at least some) fields
  - Must respect .equals() — if two objects are equals(), they must have same .hashCode()
  - Why is it ok for converse to be false?
- When in doubt? Convert to string, call .hashCode()
  - Need .toString() anyway
- Some details?

What is an ArrayList of ArrayLists?

- Think lockers, and in each locker there's a line of cubbies, an ArrayList
  - Easy to implement, performance of remove? ...
  - Searching in a bucket, or locker, that's long ...
  - Avoid ArrayList, use Linked List (low-level)
- Changes in Java 8 to make more efficient
  - Don't use low-level linked lists
  - Do use low-level trees
SimpleHashSet v ArraySet

- We'll look carefully at interfaces and client code
  - What changes when we change implementation in client/driver program, e.g., https://git.cs.duke.edu/201fall16/building-arrays/blob/master/src/SimpleSetBenchmark.java, see also SetDriver.java
  - Analytic performance on N words with U unique
    - For every word read ..., What do you do?
    - For ArraySet this is ..., NU which means ...
    - For HashSet this is ..., Small buckets means: N
    - If buckets aren't small? Disaster! Collisions

Questions about Sets


- Which method in the Set interface is hardest to implement? Why?
- What must we do to implement the Set interface?
  - https://git.cs.duke.edu/201fall16/building-arrays/blob/master/src/ConformingSimpleHashSet.java
  - https://git.cs.duke.edu/201fall16/building-arrays/blob/master/src/SimpleHashMultiSet.java

Maria Klawe

Chair of Computer Science at UBC, Dean of Engineering at Princeton, President of Harvey Mudd College, ACM Fellow,...

Klawe’s personal interests include painting, long distance running, hiking, kayaking, juggling and playing electric guitar. She describes herself as “crazy about mathematics” and enjoys playing video games.

“I personally believe that the most important thing we have to do today is use technology to address societal problems, especially in developing regions”
Map: store pairs of (key,value)

- Search engine: (K,V): (query, list of pages)
  - Key: word or phrase, value: list of web pages
  - This is a map: search query -> web pages
- DNS: (K,V): (domain name, IP address)
  - domain name, duke.edu, value: 152.3.189.29
  - This is a map: domain name -> IP address
- Color Name/RGB (K,V): (name, (r,g,b) triple)
  - Duke Blue: (0,0,156)
  - Dartmouth Green (0,105,62)

Simple Map Example: YAWTCW

```java
private Map<String, Integer> myMap;

public SimpleMapDemo()
    myMap = new HashMap<>();

public void processFile(File f) throws FNFE {
    Scanner scan = new Scanner(f);
    while (scan.hasNext()) {
        String s = scan.next().toLowerCase();
        if (! myMap.containsKey(s)) {
            myMap.put(s, 0);
        }
        myMap.put(s, myMap.get(s)+1);
    }
}
```

The java.util.Map interface, concepts

- Generic <Key,Value> or <K,V>
  - Map.Entry<K,V> has getters() for K and V
  - These work for all Map implementations!

<table>
<thead>
<tr>
<th>Method</th>
<th>return</th>
<th>purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map.size()</td>
<td>int</td>
<td># keys</td>
</tr>
<tr>
<td>Map.keySet()</td>
<td>Set&lt;K&gt;</td>
<td>Set of keys</td>
</tr>
<tr>
<td>Map.values()</td>
<td>Collection&lt;V&gt;</td>
<td>All values</td>
</tr>
<tr>
<td>Map.containsKey(K)</td>
<td>boolean</td>
<td>Is key in Map?</td>
</tr>
<tr>
<td>Map.put(K, V)</td>
<td>V (ignored)</td>
<td>Insert (K, V)</td>
</tr>
<tr>
<td>Map.entrySet()</td>
<td>Set&lt;Map.Entry&gt;</td>
<td>Get (K, V) pairs</td>
</tr>
<tr>
<td>Map.clear()</td>
<td>void</td>
<td>Remove all keys</td>
</tr>
</tbody>
</table>
Code examples

- See example on sorting key/value pairs:
  - Create list of `Map.Entry<K,V>` objects
  - Sort the list using `Comparator.comparing(...)`
  - This is new with Java 8

- See definitions of generic/collection variables
  - `HashMap<String,Integer> h = new HashMap<>();`
  - This is new in Java 8

- My goal: if it saves typing and concepts important?

KWIC Case Study

```
Arise, fair sun, and kill the envious moon, Who
I. Yet I should kill thee with much cherishing.
shortly, for one would kill the other. Thou! why,
those twenty could but kill one life. I beg
wherefore, villain, didst thou kill my cousin? That villain
mean, But 'banished' to kill me - 'banished'? O friar,
thy happy. Tybalt would kill thee, But thou slighest
cell there would she kill herself. Then gave I
heaven finds means to kill your joys with love!
```

- Keyword In Context
  - At one point this 100+ line program was worthy of a treatise. Memory and speed changed this

Key Word in Context Explained

- For every different word, store where it occurs
  - `love` is the 1st, 3rd, 50th, and 1237th word in the file

- This data is kept in a map, key is word, value is ??
  - How do we generate the data in the map?

- Keep a map of words and their indexes:
  - `the`: [0,3]
  - `fox`: [1,4,...]
  - `cried`: [2,...]

KWIC Questions

- Concentrate on high-level aspects of map

- How will we print every keyword in context, all keywords in alphabetical order
Luis von Ahn (Duke 2000)

I build systems that combine humans and computers to solve large-scale problems that neither can solve alone. I call this Human Computation, but others sometimes call it Crowdsourcing.