**Compsci 201**  
Maps and Midterms

- Java  
  - A simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, high performance, multi-threaded, and dynamic language.
- Just in Time Teaching  
  - Introduce concepts when needed, in context of solving problems. WOTO style

**Announcements**

- Exam 1 Friday, Feb 14!
- Assignment P2 due tomorrow, Feb 13  
  - Get it done early, great practice for exam  
  - Grace period is extended
- Assignment P3 will build on Assignment P2
- APT-Quiz coming next week  
  - Do by yourself
- Discussion 6 on Feb 17

**PFTDBE1**

- Maps: API and Problem Solving  
  - Keys and Values
- Toward Hashing DIYAD  
  - From locker analogies to code
- Midterm details and review  
  - What to do, bring, think about
Go over – WOTO from last time


Problems and Solutions

• String that occurs most in a list of strings?
  • CountingStringsBenchmark.java, two ideas
    • See also CountingStringsFile for same ideas
      • https://coursework.cs.duke.edu/201spring20/classcode
  • Parallel arrays: word[k] occurs count[k] times
  • Use ArrayLists: 2 “the”, 3 “fat”, 4 “fox”

<table>
<thead>
<tr>
<th></th>
<th>the</th>
<th>fox</th>
<th>cried</th>
<th>fat</th>
<th>tears</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

How does the code work?

• Process each string s
  • First time words.add(s), counter.add(1)
  • Otherwise, increment count corresponding to s
  • c[s] += 1

```
public static String parallelArrays(List<String> list) {
  ArrayList<String> words = new ArrayList<>();
  ArrayList<Integer> counter = new ArrayList<>();

  for(String w : list) {
    int index = words.indexOf(w);
    if (index == -1) {
      words.add(w);
      counter.add(1);
    } else {
      counter.set(index, counter.get(index) + 1);
    }
  }
}
```

Tracking N strings

• Complexity of search? O(M) for M different words
  • Complexity of words.indexOf(…) is O(M)
  • what about all calls? 1 + 2 + … N is N(N+1)/2

```
public static String parallelArrays(List<String> list) {
  ArrayList<String> words = new ArrayList<>();
  ArrayList<Integer> counter = new ArrayList<>();

  for(String w : list) {
    int index = words.indexOf(w);
    if (index == -1) {
      words.add(w);
      counter.add(1);
    } else {
      counter.set(index, counter.get(index) + 1);
    }
  }
}
```
Understanding O-notation

- This is an upper bound and in the limit
  - Coefficients don’t matter, \textit{order of growth}
  - \(N + N + N + N = 4N\) is \(O(N)\) — why?
  - \(100*N*N\) is \(O(N^2)\) — why?
  - \(O(1)\) means independent of \(N\), constant time

- In analyzing code and code fragments
  - Account for each statement
  - How many times is each statement executed?

CountingStringsFile.java

- Generate an ArrayList of Strings
  - Find the word that occurs the most often
    - See three different methods

\begin{verbatim}
LIVE C O D I N G
\end{verbatim}

O(N²) too slow, solution?

- Rather than parallel arrays, where search is \(O(N)\)
  - Use hashing, where search is \(O(1)\) — wow!
  - (String,Integer) stored together in \texttt{map}

\[ \begin{array}{ll}
10000 & 0.389 \text{niojlkmp:1} \\
20000 & 0.672 \text{lnaji:2} \\
30000 & 1.409 \text{lnaji:2} \\
40000 & 4.064 \text{lnaji:2} \\
50000 & 5.140 \text{lnaji:2} \\
60000 & 6.410 \text{lnaji:2} \\
70000 & 8.474 \text{lnaji:2} \\
80000 & 11.768 \text{lnaji:2} \\
90000 & 16.839 \text{lnaji:2} \\
100000 & 22.001 \text{lnaji:2} \\
\end{array} \]
Map conceptually (key,value)

- Search engine: (K,V) is (query, list of web pages)
  - Key is word or phrase, Value: list of pages
  - Maps query to list of web pages/URLs
- Internet: URL -> IP address
- Color Name/RGB triple: (K,V) is (name, (r,g,b))
  - Duke Blue maps to (0,48, 135)
  - NCSU Wolfpack red maps to (204, 0, 0)
  - Purdue Boilermakers gold maps to (194,142, 12)

Map: Keys and Values

- I'm looking for the value associated with a key
  - The key is a string, a Point, almost anything
- Given a food, find calories and protein
  - Key: food, Value: (calorie, protein) pair

Map Code in Java

- jshell

```
jshell> map
map = {cat=0}
jshell> map.put("dog", 5)
5
map = {dog=5}
jshell> map.put("fish", 5)
5
map = {cat=0, fish=5, dog=5}
jshell> map.get("dog")
5
```

A Rose by Any Other Name…
Examining Map Code in CountingStringsBenchmark.java

- First time key is seen, set value to zero. Why?
  - `map.get(key)` return?
  - `map.put(key, value)` does?
  - `map.putIfAbsent(key, value)` does?

```java
public static String mapping(List<String> list) {
    Map<String, Integer> map = new HashMap<>();
    System.gc();
    for (String w : list) {
        map.putIfAbsent(w, 0);
        map.put(w, map.get(w) + 1);
    }
    int max = Collections.max(map.values());
    String maxString = null;
    for (String word : map.keySet()) {
        if (map.get(word) == max) {
            maxString = word;
            break;
        }
    }
    return maxString + "=" + max;
}
```

Building Map `<String, Integer> as <Key, Value>`

- For each string `s`, create `<S, 0>` initially
  - We are going to increment the value, start at 0
  - Notice line 65: analogous to `map[w] += 1`
  - That syntax doesn't work in Java

Map concepts, HashMap concepts

- Keys should be immutable, cannot change
  - If you change a key, you change its `hashCode`, so where does it go? What Bucket?
- Keys unique, there's a `KeySet`!
- HashMap: `key` uses `.hashCode()`, `value` anything
  - How big is the set of lockers? Can it change?
  - Big enough, but can grow if needed
The java.util.Map interface, concepts

- **HashMap <Key,Value> or <K,V**

<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.get(K)</td>
<td>V</td>
<td>get value</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>
<tr>
<td>map.putIfAbsent(K,V)</td>
<td>V (ignored)</td>
<td>Insert if not there</td>
</tr>
</tbody>
</table>

HashMap Internals

- What does map.get(key) actually do?
  - Find h = key.hashCode()
  - Find the h\(^{th}\) bucket/locker/location of map/table
  - Actually use `Math.abs(h) % (# buckets)`

- Look at all the values in that bucket/locker
  - Could be ArrayList or LinkedList or …
  - Traverse searching for .equals(key)

- What is best case? Average case? Worst Case

Toward Diyad for HashMap

- We saw synthetic workload in previous program
  - Reading words from file, similar program
    - [https://coursework.cs.duke.edu/201spring20/classcode/blob/master/src/CountingStringsFile.java](https://coursework.cs.duke.edu/201spring20/classcode/blob/master/src/CountingStringsFile.java)

- How does HashMap work?
  - Compare parallel arrays, HashMap as before
  - Add method to illustrate how HashMap works

CountingStringsFile.java

- Method parallelArraysMax(list) – previously saw
- Method hashMapMax(list) – same map code
- Method hashMax(list) – version how hashmap works
Not Ideal Design: Pair as pojo

- **Private:** *plain old java object*, only used here
  - Only uses one field for `.equals` and `.hashCode`
  - Code ensures no two Pairs have same string
- **Class is private**
  - Restricted use
  - No getter/setter
  - Access myCount

```java
public class Pair {
    String myString;
    int myCount;

    Pair(String s) {
        myString = s;
        myCount = 1;
    }

    @Override
    public int hashCode() { return myString.hashCode(); }

    @Override
    public boolean equals(Object o) {
        Pair p = (Pair) o;
        return p.myString.equals(myString);
    }
}
```

How to use Pair?

- 5,000 lockers. Each locker contains an ArrayList
  - Create Pair
  - Find locker
  - Look in list

```java
public String hashMax(List<String> list) {
    ArrayList<ArrayList< Pair>> hash = new ArrayList<>();
    for(int k=0; k < TABLE_SIZE; k++) {
        hash.add(new ArrayList<>());
    }

    for(String s : list) {
        Pair p = new Pair(s);
        int hval = Math.abs(p.hashCode()) % hash.size();
        int index = hash.get(hval).indexOf(p);
        if (index == -1) {
            hash.get(hval).add(p);
        } else {
            hash.get(hval).get(index).myCount += 1;
        }
    }
    return hashMax;
}
```
How do you read this line?

hash.get(hval).get(index).myCount += 1;

Analysis and Experiments

• Does code depend on # lockers/size of table?
  • Change HTABLE_SIZE and see

• Can different Pair objects be in same locker?
  • Yes, two different strings can have same hashCode()
  • p.equals(q) is false
    • but p.hashCode() == q.hashCode()

WOTO


Barbara Liskov

• Turing Award Winner in 2008 for contributions to practical and theoretical foundations of programming language and system design, especially related to data abstraction, fault tolerance, and distributed computing.
• Developed CLU programming language

The advice I give people in general is that you should figure out what you like to do, and what you can do well—and the two are not all that dissimilar, because you don’t typically like doing something if you don’t do it well. ... So you should instead watch—be aware of what you’re doing, and what the opportunities are, and step into what seems right, and see where it takes you.
Exam 1

• Review syllabus for policies
  • Missing Exam 1 – Fill out form on webpage
  • Bring 1 page of notes, front and back, 8.5x11 inches, name and netid on it, MUST TURN IN

• Exam covers all topics through today
  • Arrays, ArrayLists, HashSets, HashMaps, Classes, etc
  • Mix of read code, short answer, write code
  • Problems have recommended Time to take

• Map questions will be primarily reading
  • You should be able to update a map and basic map methods

Lists, and Sets, and … Oh My!

• First step: get all words, store in a list and a set
  • Don't need both, nod to efficiency
  • For each loop? Easier if index not needed

Maps on APTs

• [https://www2.cs.duke.edu/csed/newapt/bigword.html](https://www2.cs.duke.edu/csed/newapt/bigword.html)

• Before you knew about maps …
  • Count each word, maximal value? Done
  • How do we get each word in each string?
    • Call `s.split(" ")`
  • How do we find out how many occurrences?
    • Helper method or `Collections.frequency(...)`

• All words, one word; one loop, two loops

BigWord APT

Problem Statement

In days of yore, ska BG (Before Google), search engines ranked webpages in part by the number of occurrences of a word on the page. You should write method `most` to determine and return the word that occurs most often in an array of sentences. This most frequently occurring word will be unique --- that is you don't need to worry about two words both occurring more often than any other word. The word returned should be all lower-case regardless of the case of letters in sentences.

Each string in sentences represents several words, each word is delimited by spaces from other words. Words should be considered the same without respect to case, so BIG is the same word as big, for example.

<table>
<thead>
<tr>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>public class BigWord {</td>
</tr>
<tr>
<td>public String most(String[] sentences) {</td>
</tr>
<tr>
<td>// you write code here</td>
</tr>
<tr>
<td>}</td>
</tr>
<tr>
<td>}</td>
</tr>
</tbody>
</table>

Examples

1. sentences = {"one fish two", "fish red fish blue", "fish this fish is black"}
Returns: fish

   The word "fish" occurs five times, which is more than any other word.
Finding maximal # occurrences

- Can we substitute list for set in code below?
  - N words in list, M words in set
  - Code below is O(MN), if list used? O(N^2)

```java
int max = 0;
String ms = "",;
for(String s : set) {
  int count = Collections.frequency(list, s);
  if (count > max) {
    max = count;
    ms = s;
  }
}
return ms;
```

Investigate Map Solution

- One pass over the data instead of many passes
  - Understand all map methods
  - Why is line 39 never executed? Still needed?

```java
public String most(String[] sentences) {
  Map<String, Integer> map = new HashMap<>();
  for(String one : sentences) {
    for(String s : one.toLowerCase().split(" ")) {
      map.putIfAbsent(s, 0);
      map.put(s, map.get(s) + 1);
    }
  }
  int mx = Collections.max(map.values());
  for(String key : map.keySet()) {
    if (map.get(key) == mx) return key;
  }
  return "never";
}
```

APT Quiz next week

- You’ve practiced programming on APTs and assignments. Typically you don’t write the code with paper/pencil
  - Limitations of exams: not easy to “write” code

- We use APT quizzes to verify: can you solve a problem by programming
  - Have you understood and mastered the Java concepts we’ve studied

APT Quiz Details

- You’ll get a “practice” quiz as a prelude to and as part of discussion section
  - You should work to do these on your own before discussion
  - You should get answers to questions in discussion

- You CANNOT, CANNOT, CANNOT collaborate on the quiz. We run reasonably sophisticated similarity detection software
Quiz Logistics

- You can start the quiz anytime between Thursday and Monday (Feb. 20-24)
  - Do not start until you have two consecutive hours to complete the quiz

- You must track time yourself. As soon as you access the quiz, your time starts
  - We will only count code you submit before time is up, even though you can keep submitting.
  You should not keep submitting

APT Quiz

- We expect that everyone will get the first problem
  - Sometimes we are wrong. But it’s designed to be straightforward. If you’ve done the APTs? You’ll succeed

- We expect everyone will know how to solve the other problems, but sometimes coding and debugging is not easy
  - There is a time limit, if stuck? Try next problem