High-level Java concepts: interfaces and maps
- What is a map, what is an interface, how do we use them
- How are interfaces useful, what are they used for
- What is a map, how is it implemented, how like array?

From Hangman to APTs to Jotto
- What's hard about Java, what's hard about classes
- What do you want to know about Java? What is it good to know?
- How do we get the programs done, how much time?

1-2-3hour rule: when to give up, when to persevere

You concentrate on algorithms and data structures
- Not on rendering fonts, interacting with users
- This is important! But not what this course is about

We build GUIs or views that facilitate projects
- You write the brains, we build the skull/manikin
- Our GUI communicates with your code and your code communicates with the GUI
  - Requires following conventions in interacting code

GUI libraries can be similar across languages, but...
- Deeper Java specific details than ArrayList or Map

You read file, find word and it's context, print
- Can find all words, but how do we get context?
- Loop and look: create context line for each occurrence

Dissect and inspect KWIC code to understand conventions
- Understand Model and View interaction
- Facilitates doing Jotto and Markov assignments

Review some basic coding idioms and ideas
- Avoiding recomputing same value, readability, modifiability, ...

Coping with Java idioms: how do we pass a List, String, Integer, all in the same parameter?
- What does inheritance and Object provide?

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, thou happy. Tybalt would kill thee, But thou slieast
call there would she kill herself. Then gave I heaven finds means to kill your joys with love!

 KWIC: Key word in Context

Read file, find word and it's context, print
- Can find all words, but how do we get context?
- Loop and look: create context line for each occurrence
- See KWICModel.java

Use KWIC example to motivate study
MVC Example, key-word-in-context

- User loads file
  - Where? Communicate to?
  - What changes in model?
  - What happens in view?

- User chooses word
  - Process in Model
  - Alternatives?
  - Generate context, display
  - How to show in any view?

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?

- Given a word, how do we find its context? How do we format?
  - All words are in an array, in order
  - Memory concerns?
  - Original KWIC paper by Parnas as comparison

Two Maps: by name and by concept

- Keep an array of all words, in order as they occur

<table>
<thead>
<tr>
<th>the</th>
<th>dead</th>
<th>cried</th>
<th>the</th>
<th>dead</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

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

- A map stores (key,value) pairs, conceptually
  - Look up a key, find it's value
  - What are the maps above?

Code Interlude

- Examine KWICModel.process
  - Called when user enters word, parameter is the word
  - If file already read, we don’t need map, where is this?
  - Error checking? When and when happens
  - How does Model communicate back to View?

- Examine KWICModel.justify
  - What is method doing
  - What is parameter, where was it constructed, issues?
  - What about 'magic numbers', e.g., 30?
  - What about comments, should we add some?
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

- Map (aka table, hash) associates keys with values: pairs
  - Insert (key, value) into map, iterate over keys or pairs
  - Retrieve value associated with a key, remove pair

Maps, another point of view

- An array is a map, consider array \texttt{arr}
  - The key is an index, say \(i\), the value is \(arr[i]\)
  - Values stored sequentially/consecutively, not so good if the keys/indexes are 1, 100, and 1000, great if 0, 1, 2, 3, 4, 5

- Time/space trade-offs in map implementations, we’ll see more of this later
  - TreeMap: most operations take time \(\log(N)\) for \(N\)-elements
  - HashMap: most operations are constant time on average
    - Time for insert, get, ... doesn’t depend on \(N\) (wow!)
  - But! Elements in TreeMap are in order and TreeMap usually uses less memory than HashMap

Map (foreshadowing or preview)

- Any kind of Object can be key in a HashMap
  - But, performance might be terrible if hashValue isn’t calculated well
  - Every object has a different number associated with it, we don’t want every object to be associated with 37, we want things spread out

- Only Comparable object can be key in TreeMap
  - Basically compare for less than, equal, or greater
  - Some objects are naturally comparable: String, Integer
  - Sometimes we want to change how objects are compared
  - Sometimes we want to invent Comparable things

KWIC main program/class

```java
public class KWICMain {
    public static void main(String[] args) {
        KWICModel model = new KWICModel();
        SimpleViewer view = new SimpleViewer("Compsci 100 KWIC", "word");
        view.setModel(model);
    }
}
```

- What changes in above for Jotto assignment?
  - How can view communicate with any model?
  - View doesn’t change, model does!
Model View Controller, MVC

- Gui is the View and often the controller
  - Separate user-interaction from updates to data
- User loads file, chooses word, ...
  - Model notified, computes, updates view
- Model has all the state and knows when it changes
  - Communicates changes to views (via controller)
  - Must be initialized, updated, etc.

- Very common Design Pattern
  - Capture common solutions to problems in a context
  - Iterator, Composite, Decorator seen in Compsci 100

Eugene (Gene) Myers

- Lead computer scientist/software engineer at Celera Genomics, then at Berkely, now at Janelia Farms Research Institute (HHMI)

"What really astounds me is the architecture of life. The system is extremely complex. It's like it was designed." ... "There's a huge intelligence there."

- BLAST and WG-Shotgun

Methods, Interfaces, Inheritance

- A method by any other name would smell as sweet
  - Method in OO languages, functions, procedures in others
  - Parameters and return value: communication
    - Do objects or methods communicate?: OO v procedural

- Static: Math.sqrt, Character.toUpperCase
  - Don’t belong to an object, invoked via class (clue above?)
  - Java API helpful here

- interfaces are conceptual (lower-case i)
  - Interfaces are part of Java: Map, List, Set (Hash, Array, Tree)

What does Object-Oriented mean?

- Very common method of organizing code
  - Design classes, which encapsulate state and behavior
  - Some classes can be similar to, but different from their parent class: inheritance
    - Super class, subclass
  - Inherit behavior, use as is or modify and use or both

- Hard to design a hierarchy of classes, but important
  - More of this in Compsci 108 or on-the-job training
  - We solve simple problems, don't design re-usable libraries

- Simple doesn't mean it's not hard/difficult