Underneath the Hood with java.util

- Using an API. The client of a good library
  - Software design and engineering: techniques
  - Architect, engineer, scientist, creator

- Do you need to be able to implement: data structures or algorithms?
  - Learning Objectives of course ... CS problem
  - Not all languages have Java's libraries

- Knowledge of implementation facilitates ...
  - Debugging can depend on implementation

Two collections: arraylist and set

- Collection, List, Set.... Oh my!
  - https://docs.oracle.com/javase/8/docs/api/java/util/Collection.html

- What is a Java Interface?
  - Specifies names of methods and some behavior, but implementations can differ on how to implement the interface
  - Difference between add to list and add to set?
  - No syntactic difference, but semantic difference
  - Performance differences too

How do you implement ArrayList?

- We'll look at some example code for three classes
  https://git.cs.duke.edu/201fall16/building-arrays/tree/master/src

- SimpleStringArrayList – when out of room?
  - Throw Exception

- GrowableStringArrayList – when out of room?
  Make more room and copy old to new space
  - Grow by adding 1 entry? Grow by doubling

- ConformingArrayList
  - Toward using with Collections API

Start Simple!


- What happens with methods .add and .get?
  - If index is out of bounds? If no room to add?

- Why is MAX_SIZE static? Don't need to have separate copy in every object, shared across all
  - Static means it's for the class, not for the object
Making Array Grow

- When out of storage? Grow by creating more
  - What if computer runs out of storage?

- What’s going on in the body of checkSize()??
  - Why is this method private?
  - What does System.arraycopy do?
  - What happens to the old 'myStorage'?

- Subtle: why not call rangeCheck in .add(dex,"str")?
  - Very subtle!

Performance Analysis

- If array grows by +1 when out of storage, starting with "room for one" and adding 1024 elements
  - $1 + 2 + 3 + 4 + \ldots + 100 + 101 + \ldots + 1024 = \ldots$
  - Total storage for N? $\sum_{k=1}^{k} = N(N+1)/2$
  - Expression related to N is $N^2$

- If array grows by doubling, how much?
  - $1 + 2 + 4 + 8 + 16 + 32 + \ldots + 512 + 1024 = \ldots$
  - Total storage for N? Expression is $2^N - 1$

- What do these graphs look like?

Quadratic v Linear

- Linear is very fast. But it’s not fast enough!???

Helpful formulae

- We always mean base 2 unless otherwise stated
  - What is $\log(1024)$?
  - $\log(xy) = \log(x) + \log(y)$
  - $\log(2^n) = 2 \log n$

- Sums (also, use sigma notation when possible)
  - $1 + 2 + 4 + 8 + \ldots + 2^k = 2^{k+1} - 1 = \sum_{i=0}^{k} 2^i$
  - $1 + 2 + 3 + \ldots + n = n(n+1)/2 = \sum_{i=1}^{n} i$
  - $a + ar + ar^2 + \ldots + ar^{n-1} = a(r^n - 1)/(r-1) = \sum_{i=0}^{n-1} ar^i$
Questions about ArrayList


- You’ll need to look at code in gitlab, that may not support 200+ simultaneous views, .... Scale?

What is a Java Interface?

- Similar to a class, but a specification rather than an implementation
  ```java
  public class ArrayList implements List
  ```
  List is the interface, ArrayList is implementation
  List supplies method signatures, implementing classes supply ... implementations?

- The Set interface is realized by at least three classes
  Different performance characteristics
  Some different use-cases, e.g., order matters

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?

- Analytic peformance 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?

**Theory and Practice**

- In theory writing software is simple, in practice?
  - Deploying new grading/submission system for assignments in 201

- When you write solutions/programs in 201
  - Don't try for more than an hour when progress is minimal
  - Don't overestimate how much progress you're making