APTs and structuring data/information

- Is an element in an array, Where is an element in an array?
  - DIY: use a loop
  - Use Collections, several options
  - Tradeoffs?

```java
public boolean contains(String[] list, String target){
    for (String s : list){
        if (s.equals(target)) return true;
        return false;
    }
}
```

```java
public boolean contains(String[] list, String target){
    return Arrays.asList(list).contains(target);
}
```

```java
public boolean contains(String[] list, String target){
    return new HashSet<String>(Arrays.asList(list)).contains(target);
}
```

APTs and Class/OO/Java tradeoffs

- If you search a list of elements once, you must “touch” every element, so why worry about doing anything else?
  - Can you skip an element when searching? Why?
  - What about a sorted list of elements?

- If you search repeatedly, it makes sense to organize data
  - Leverage or amortize the cost of the organization over several searches
  - Where do we store the organized data so that we can access it repeatedly?

Class State and helper methods

- Instance variables: initialized once and repeatedly accessed
  - Where to call new?
  - How to use helper methods — private or otherwise

```java
import java.util.*;
public class SearchForStuff {
    private HashSet<String> mySet;

    private boolean contains(String target){
        return mySet.contains(target);
    }

    public int findMePlease(String[] data, String[] query){
        mySet = new HashSet<String>(Arrays.asList(data));
        for(int k=0; k < query.length; k++){
            String[] all = query[k].split(" ");
        }
    }
}
```

Data and Information

Google Announces Plan To Destroy All Information It Can’t Index

How and why do we organize data? Differences between data and information?

- "We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandable as the world around us. We want the world to be as simple, clear, and understandab
Where is www.cs.dartmouth.edu?

traceroute www.cs.dartmouth.edu
traceroute to katahdin.cs.dartmouth.edu (129.170.213.101), 64 hops max,
1 lou (152.3.136.61) 2.566 ms
2 152.3.219.68 (152.3.219.69) 0.258 ms
3 teillap-roci.netcom.duke.edu (152.3.219.54) 0.336 ms
4 rlgh7600-gw-to-duke7600-gw.ncren.net (128.109.70.17) 194.752 ms
5 rlgh1-gw-to-rlgh7600-gw.ncren.net (128.109.70.37) 1.379 ms
6 rtg11-gw-to-rtgp-ooc48.ncren.net (128.109.52.1) 1.840 ms
7 rtg7600-gw2-to-rtg7600-gw.ncren.net (128.109.70.122) 1.647 ms
8 dep7600-gw2-to-rtg7600-gw.ncren.net (128.109.70.138) 2.273 ms
9 internet2-to-dep7600-gw2.ncren.net (198.86.17.66) 10.494 ms
10 gw-0-1-0-10.nyong.abilene.ucaid.edu (64.57.28.7) 24.058 ms
11 oz-0-0-0-0.rtr.newy.net.internet2.edu (64.57.28.10) 45.609 ms
12 nxo300gw-vl-nl-nxv.internet2.nox.org (192.5.89.221) 33.839 ms
13 ...
14 ...
15 border.ropewerry1-crt.dartmouth.edu (129.170.2.193) 59.991 ms
16 katahdin.cs.dartmouth.edu (129.170.213.101) 50.480 ms

John von Neumann

“Anyone who attempts to generate random numbers by deterministic means is, of course, living in a state of sin.”

“There’s no sense in being precise when you don’t even know what you’re talking about.”

“There are two kinds of people in the world: Johnny von Neumann and the rest of us.”

Eugene Wigner, Noble Physicist

From Google to Maps

- If we wanted to write a search engine we’d need to access lots of pages and keep lots of data
  - Given a word, on what pages does it appear?
  - This is a map of words->web pages

- In general a map associates a key with a value
  - Look up the key in the map, get the value
  - Google: key is word/words, value is list of web pages
  - DNS: Domain name is key, IP address is a value

- Interface issues
  - What if the key isn’t in the map, what value returned?
  - What if there’s already a value associated with a key?

Interface at work: MapDemo.java

- Key is a string, Value is # occurrences
  - Code below shows how Map interface/classes work
- What clues are there for prototype of map.get and map.put?
  - What if a key is not in map, what value returned?
  - What kind of objects can be put in a map?

```java
for(String s : list) {
    s = s.toLowerCase();
    Integer count = map.get(s);
    if (count == null){
        map.put(s,1);
    }
    else{
        map.put(s,count+1);
    }
}'''

CPS 100
What can an Object do (to itself)?

  - Look at java.lang.Object
  - What is this class? What is its purpose?

- `toString()`
  - Used to print (System.out.println) an object
  - overriding `toString()` useful in new classes
  - String concatenation: `String s = "value " + x;`
  - Default is basically a pointer-value

What else can you do to an Object?

- `equals(Object o)`
  - Determines if guts of two objects are the same, must override, e.g., for using `a.indexOf(o)` in ArrayList `a`
  - Default is `==`, pointer equality

- `hashCode()`
  - Hashes object (guts) to value for efficient lookup

- If you’re implementing a new class, to play nice with others you must
  - Override `equals` and `hashCode`
  - Ensure that equal objects return same `hashCode` value

Objects and values

- Primitive variables are boxes
  - think memory location with value
- Object variables are labels that are put on boxes

```java
String s = new String("genome");
String t = new String("genome");
if (s == t) { they label the same box }
if (s.equals(t)) { contents of boxes the same }
```

Objects, values, classes

- For primitive types: int, char, double, boolean
  - Variables have names and are themselves boxes (metaphorically)
  - Two int variables assigned 17 are equal with `==`

- For object types: String, ArrayList, others
  - Variables have names and are labels for boxes
  - If no box assigned, created, then label applied to `null`
  - Can assign label to existing box (via another label)
  - Can create new box using built-in `new`

- Object types are references or pointers or labels to storage
Anatomy of a class

```java
public class Foo {
    private int mySize;
    private String myName;
    public Foo() {
        // what's needed?
    }
    public int getSize() {
        return mySize;
    }
    public double getArea() {
        double x;
        x = Math.sqrt(mySize);
        return x;
    }
}
```

- What values for vars (variables) and ivars (instance variables)?

David Parnas

"For much of my life, I have been a software voyeur, peeking furtively at other people's dirty code. Occasionally, I find a real jewel, a well-structured program written in a consistent style, free of kludges, developed so that each component is simple and organized, and designed so that the product is easy to change."

Parnas on re-invention

"We must not forget that the wheel is reinvented so often because it is a very good idea; I've learned to worry more about the soundness of ideas that were invented only once."

David Parnas (entry in Wikipedia)

- **Module Design**: Parnas wrote about the criteria for designing modules, in other words, the criteria for grouping functions together. This was a key predecessor to designing objects, and today's object-oriented design.
- **Social Responsibility**: Parnas also took a key stand against the Strategic Defense Initiative (SDI) in the mid 1980s, arguing that it would be impossible to write an application that was free enough from errors to be safely deployed.
- **Professionalism**: Parnas became one of the first software engineers to earn a professional engineering license in Canada. He believes that software engineering is a branch of traditional engineering.
Tomato and Tomato, how to code

- `java.util.Collection` and `java.util.Collections`:
  - one is an interface
    - `add()`, `addAll()`, `remove()`, `removeAll()`, `clear()`
    - `toArray()`, `size()`, `iterator()`
  - one is a collection of static methods
    - `sort()`, `shuffle()`, `reverse()`, `max()`
    - `frequency()`, `indexOfSubList()`

- `java.util.Arrays`:
  - Also a collection of static methods
    - `sort()`, `fill()`, `binarySearch()`, `asList()`

Kinds of maps, sets, lists, ...

- We'll study performance of maps and sets (and lists)
  - Difference between array and ArrayList (LinkedList)
    - `size`, `performance`, `operations`
  - Difference between TreeSet and HashSet (and Map)
    - Performance of `get` and `put`
    - Comparable v. Hashable, issues?

- How to convert between array and List?
  - `Arrays.asList(...)`
  - `list.toArray(...)`
    - Which is static method, what are parameters?