If I only had one variable type...
It would be a HashMap
HashMap
key -> value

Can be any type
A HashMap is a collection which allows you to look up values based on keys.
Name (String) -> Number (Integer)

![Contact List]

- Micah Cratty
- Waddie CrazyHorse
- Bryce Cronkite-Ratcliff
- Collin Cronkite-Ratcliff
- Ben Cunningham
- Lynn Cuthriell
- Waseem Daher
- Red Daly
- Richard Davis
- Philippe de Koning
- Hans Dejong

6701678
Keys (Glect) -> Sound (AudioFile)
Collections in CS106A

Aka Data Structures

- **Arrays**
  - Indexed list. Fixed size.

- **ArrayLists**
  - Indexed list. Changes size.

- **Matrices**
  - Indexed array of arrays. Fixed size.

- **HashMaps**
  - Lookup Values by Keys. Changes size.
Aside: All collections are Objects

Collection variables store addresses. Manage memory well.

```java
int[][] myMatrix = new int[5][3]
```

![Diagram of a 5x3 matrix with values 17, 12, 19, 25, 32, and 0s]

Piech, CS106A, Stanford University
End Review
Why is this so fast?

Google search for "mantis shrimp colors"

About 1,870,000 results (0.54 seconds)

Humans and many other primates have three; some birds and reptiles have four photoreceptors. Certain butterflies can even have six. But the mantis shrimp has 12 different types of photoreceptors in their eyes — and scientists haven't understood why until now. Jan 27, 2014

Study Offers Insights into Unique Color Vision of Mantis Shrimp ... www.sci-news.com/biology/science-color-vision-mantis-shrimp-01719.html
The internet is big

5 million articles and growing

Source: xkcd
Why is this so fast?

Google search:

```java
int hashFn(String key);
```

* Learn more in CS106B
A Basic Hash Function

```java
int basicStringHash(String key) {
    int preHash = 0;
    for(int i = 0; i < key.length(); i++){
        preHash += key.charAt(i);
    }
    return preHash % NUM_BUCKETS;
}
```

* Learn more in CS106B*
Why is this so fast?

Any HashMap action is a two step process. Regardless of how much data is in the HashMap.

Humans and many other primates have three; some birds and reptiles have four photoreceptors. Certain butterflies can even have six. But the mantis shrimp has 12 different types of photoreceptors in their eyes — and scientists haven't understood why until now. Jan 27, 2014.
HashMaps for search
Google Search

Key

word

“Shrimp”

Value

All websites with that word

[“Wikipedia.com/…”, “Allrecipes.com/…”, ...
]
Single Key with multiple Values?
Learn by Experimenting

6701678
Google Search

Key: search words

HashMap<String, ArrayList<String>>

Value: all websites with that word
HashMaps

ArrayLists

Arrays

mix 'em together

the possibilities are endless!
Search by name, email, work phone, or sunetid

maya

show more options
Section Leader Who

Chris Piech (cpiech)

Brahm Capoor (bcapoor)

Maya Ziv (mziv)

Gaby Candes (gcandes)

Andrew Tierno (atierno)

Christian Davis (cdavis)
Section Leader Who

The user enters a query

You give the results
**Section Leader Who**

**sls.txt:**
Has all section leaders in history, one per line.

**Example Name:**
Alisha Adam (springsky)
/* Print Query Responses
 * ---------------------
 * Print all students who have the query as either their sunetId or as one of their names
 * Example queries:
 * "Chris", "Capoor", "Maya", "mziv"
 */
private void printQueryResponses(String query) {
    // TODO: your code here
}
Approach #1: Needle in Haystack

Query: "Davis"

Chris Piech
cpiech

Brahm Capoor
bcapoor

Maya Ziv
mziv

Gaby Candes
gcandes

Andrew Tierno
atierno

Christian Davis
cdavis
Approach #1: Needle in Haystack

Query: "Davis"

Chris Piech
Brahm Capoor
Maya Ziv
Gaby Candes
Andrew Tierno
Christian Davis

cpiech
ccapoor
mziv
gcandes
atierno
cdavis
Approach #1: Needle in Haystack

Query: "Davis"

Chris Piech
cpiech

Brahm Capoor
capoor

Maya Ziv
mziv

Gaby Candes
gcandes

Andrew Tierno
atierno

Christian Davis
cdavis
Approach #1: Needle in Haystack

Query: "Davis"

Chris Piech
cpiech

Brahm Capoor
bcapoor

Maya Ziv
mziv

Gaby Candes
gcandes

Andrew Tierno
atierno

Christian Davis
cdavis
Approach #1: Needle in Haystack

Query: "Davis"

Chris Piech
,cpiech

Brahm Capoor
,bcapoor

Maya Ziv
,mziv

Gaby Candes
,gcandes

Andrew Tierno
,atierno

Christian Davis
,cdavis
Approach #1: Needle in Haystack

Query: "Davis"

- Chris Piech (cpiech)
- Brahman Capoor (bcapoor)
- Maya Ziv (mziv)
- Gaby Candes (gcandes)
- Andrew Tierno (atierno)
- Christian Davis (cdavis)
Approach #1: Needle in Haystack

Query: "Davis"

Chris Piech cpiech
Brahm Capoor bcapoor
Maya Ziv mziv
Gaby Candes gcandes
Andrew Tierno atierno
Christian Davis cdavis
Too Slow!
You design the solution

First, let's get our feet wet
Parse File Lines

```
index

0 1 2 3  ...  12  ...  17 18

"Alisha Adam (aadam)"

line.substring(0, 12);
line.substring(13, 18);
```
/* Print Query Responses */

/* Print all students who have the query as either their sunetId or as one of their names */

/* Example queries: */

"Chris", "Capoor", "Maya", "mziv"

*/

private void printQueryResponses(String query) {

    // TODO: your code here
}

Database HashMap

Key | Value
--- | ---
piech | ["Chris Piech (cpiech)""]
cpiech | ["Chris Piech (cpiech)""]
chris | ["Chris Collins (ccollins)"
"Chris Koenig (ckoenig)"
"Chris Piech (cpiech)"
"Chris Wang (cwang)"
]
wang | ["Anna Wang (awang)"
"Avery Wang (awang)"
"Chris Wang (cwang)"
"Lisa Wang (lwang)"
... ]
### Approach #2: HashMap

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
</table>

**Database HashMap**

**How do we build the hashmap?**
Database HashMap

Originally no "chris" key...
Approach #2: HashMap

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
</table>

When we see the **first** “Chris”
Approach #2: HashMap

When we see the first "Chris"
Approach #2: HashMap

Database HashMap

When we see the second “Chris”
Approach #2: HashMap

When we see the thrid "Chris"
### Approach #2: HashMap

**Database HashMap**

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
</table>
| chris | [ "Chris Collins (ccollins)"
    "Chris Koenig (ckoenig)"
    "Chris Piech (cpiech)"
    "Chris Wang (cwang)"
] |

When we see the **fourth** “Chris”
Know how to do this

Imagine a String -> String hashmap:

```java
HashMap<String, String> original = new HashMap<String, String>();
```
# Original

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Brahm”</td>
<td>“Dog”</td>
</tr>
<tr>
<td>“Maya”</td>
<td>“Cat”</td>
</tr>
<tr>
<td>“Gabby”</td>
<td>“Dog”</td>
</tr>
<tr>
<td>“Chris”</td>
<td>“Zebra”</td>
</tr>
<tr>
<td>Key</td>
<td>Value</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>&quot;Dog&quot;</td>
<td>[&quot;Brahm&quot;, &quot;Gabby&quot;]</td>
</tr>
<tr>
<td>&quot;Cat&quot;</td>
<td>[Maya]</td>
</tr>
<tr>
<td>&quot;Zebra&quot;</td>
<td>[Chris]</td>
</tr>
</tbody>
</table>
/**
 * Reverse Map:
 * Perhaps the best hashmap practice...
 * Try and work through it before next weeks section
 */

private HashMap<String, ArrayList<String>> reverse(HashMap<String, String> original) {
    // TODO: your code here
    return null;
}
What Collection?

Tic Tac Toe board

char[3][3]
What Collection?

Election Vote Counter

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Votes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mahatir</td>
<td>10,000</td>
</tr>
<tr>
<td>Najib</td>
<td>5,000</td>
</tr>
<tr>
<td>Anwar</td>
<td>15,000</td>
</tr>
<tr>
<td>Azizah</td>
<td>0</td>
</tr>
</tbody>
</table>

HashMap<String, Integer>
What Collection?

Wikipedia

HashMap<String, ArrayList<String>>
What Collection?

3D Brain Scan

Human Brain - 3D Scan
Model (quads) + lighting setup + materials (mental ray)

white matter
grey matter
ventricles

double[][][][]]
What Collection?

Google Images

HashMap<String, ArrayList<GImage>>
What Collection?

Shazam
Shazam Query

Wang, A. An Industrial-Strength Audio Search Algorithm
Shazam

You Can Call Me Al – Paul Simon. 7s,
You Can Call Me Al – Paul Simon. 43s,
All Right Now – Police. 18s
HashMap NotePair -> Song List

![Graph showing frequency over time](image)

- Dirty Paws (94s)
- All Right (18s)
- You Can Call (3s)
- You Can Call (43s)
- Riptide (3s)

Votes
HashMap NotePair -> Song List

The figure shows a scatter plot of frequency (Hz) over time (s), with dots indicating the distribution of frequencies at different times. The x-axis represents time in seconds, ranging from 0 to 10, while the y-axis represents frequency. The graph also includes a line chart with votes for different songs: "Dirty Paws (94s)", "All Right (18s)", "You Can Call (7s)", "You Can Call (43s)", and "Riptide (3s)". The votes are indicated by blue bars on the line chart.
HashMap NotePair -> Song List

![Graph showing frequency vs time with notes on votes for different songs]

Voting options include:
- Dirty Paws (94s)
- All Right (18s)
- You Can Call (3s)
- You Can Call (43s)
- Riptide (3s)
HashMap NotePair -> Song List

The top diagram shows a scatter plot with time (s) on the x-axis and frequency (Hz) on the y-axis. The data points are represented by blue dots.

The bottom diagram is a bar chart with votes on the y-axis and song titles on the x-axis. The song titles include 'Dirty Paws (94s)', 'All Right (18s)', 'You Can Call (3s)', 'Me At', 'You Can Call (43s)', and 'Riptide (3s). The bars represent the votes for each song title.
HashMap NotePair -> Song List

The diagram shows two visualizations:

1. A scatter plot with time on the x-axis and frequency (Hz) on the y-axis. The plot contains numerous data points scattered across the graph, indicating variations in frequency over time.

2. A bar chart with 'Votes' on the y-axis and various song names on the x-axis. The bar chart displays the number of votes for each song. The bar corresponding to 'Me At You Can Call (43s)' is significantly taller than the others, indicating it received the highest number of votes.