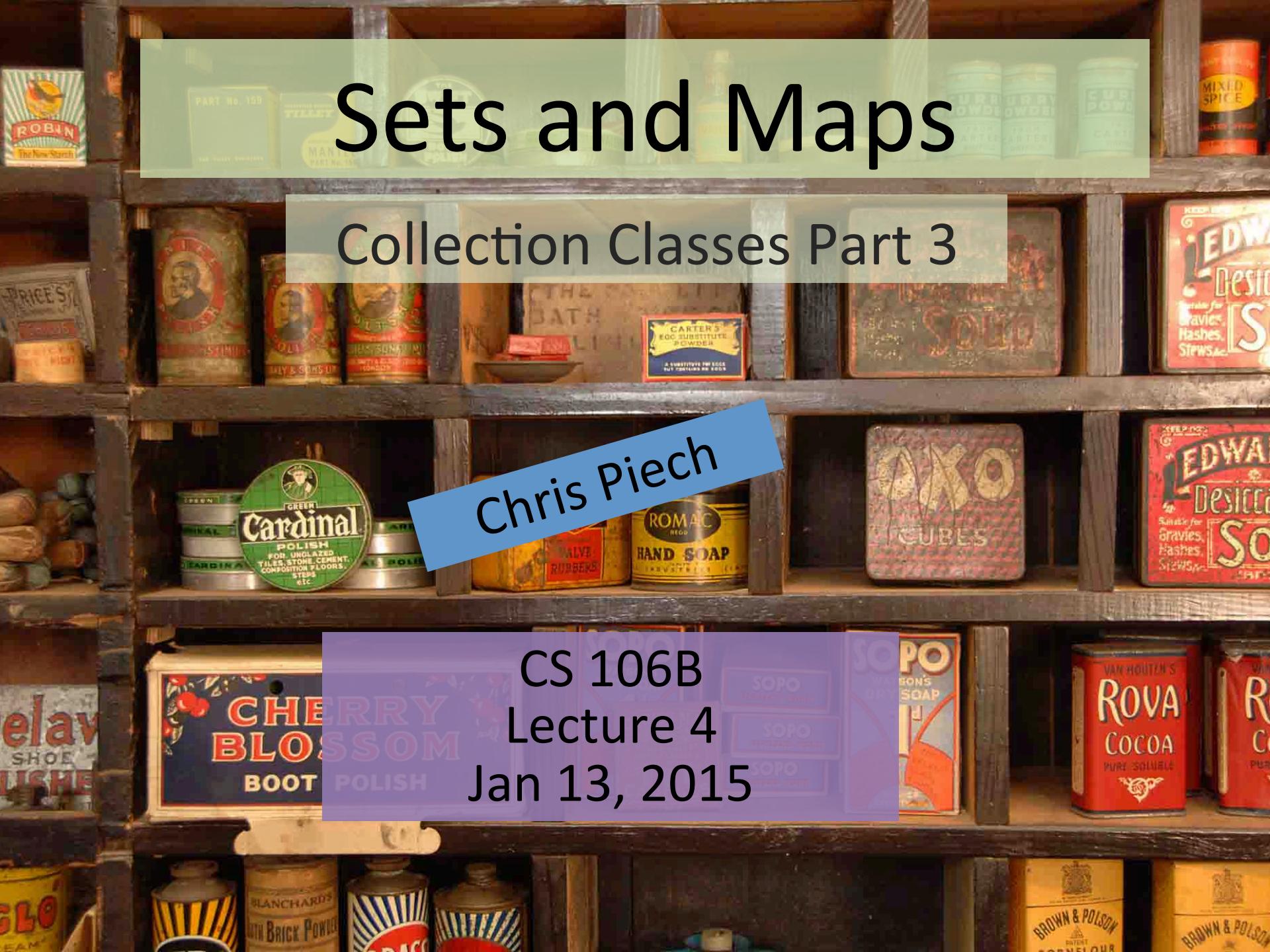


Sets and Maps

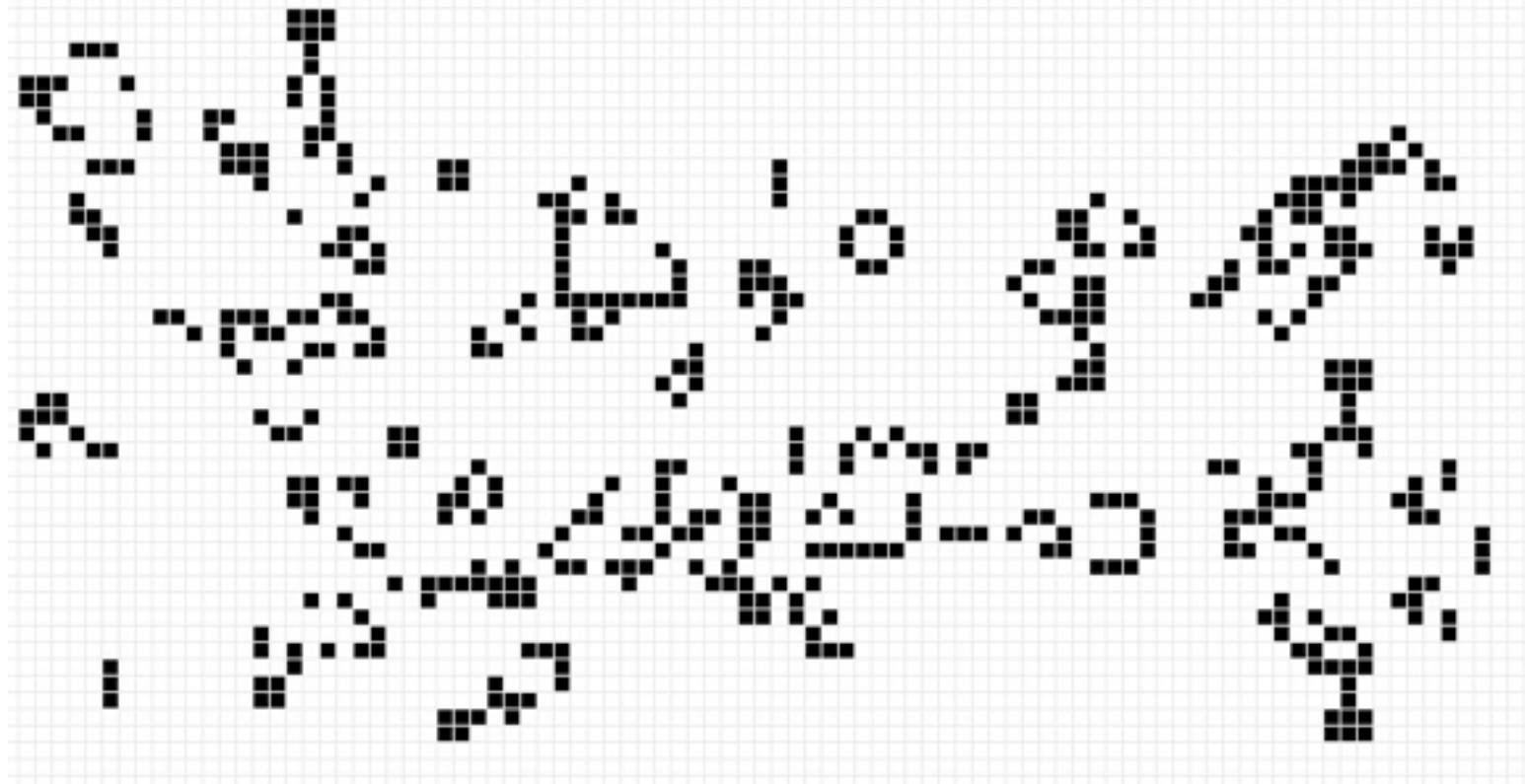
Collection Classes Part 3

Chris Piech

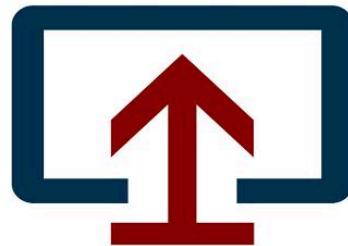
CS 106B
Lecture 4
Jan 13, 2015



Life is Due Friday



Join us at our upcoming info session to learn more about



DIVERSITYBASE

an organization dedicated to supporting people
from underrepresented backgrounds in Computer
Science with a focus on understanding the
intersectionality of those backgrounds

THURSDAY, JANUARY 14TH at 6PM
OLD UNION 200

FOR MORE INFO, CONTACT:
PATRICIA PEROZO ([PPEROZO@STANFORD.EDU](mailto:pperozo@stanford.edu))

WE NEED TO GO

DEEPER

Interesting Puzzle

Counterfeiter



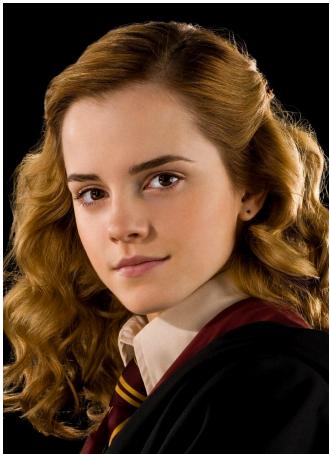
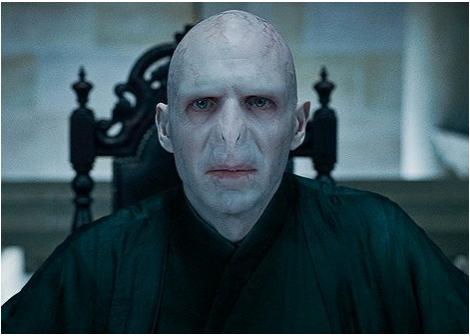
User



You (Distributor)

Interesting Puzzle

Counterfeiter



You (Distributor)

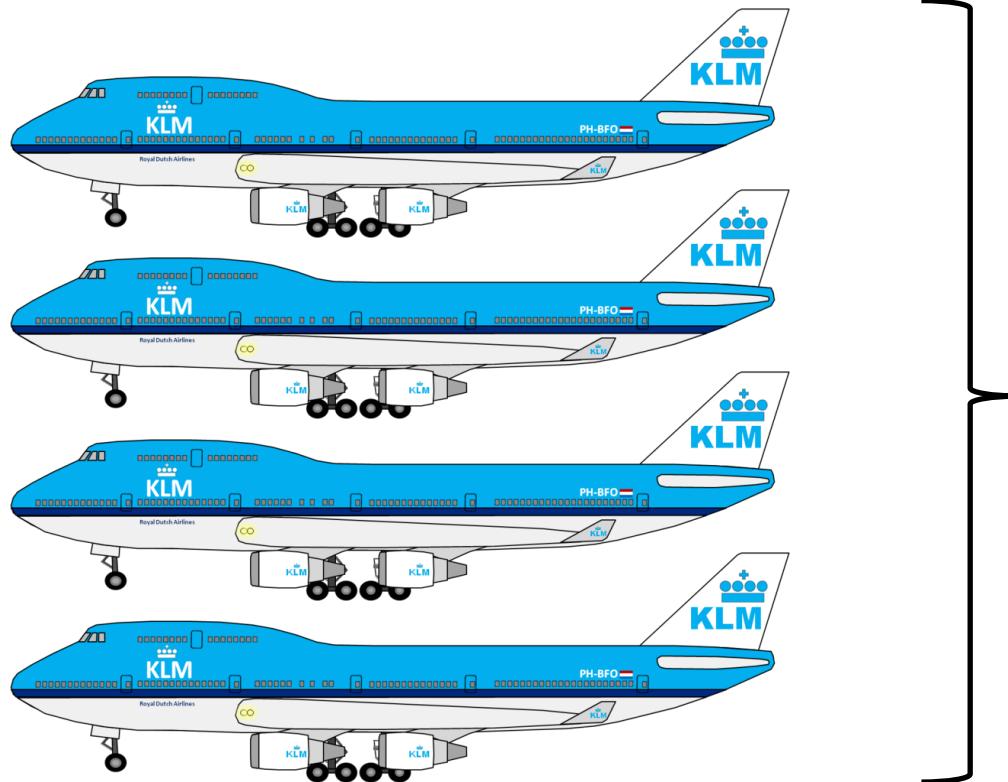


User



Fake Medicine is a Problem

700,000 deaths a year from fake malaria and tuberculosis drugs [1]

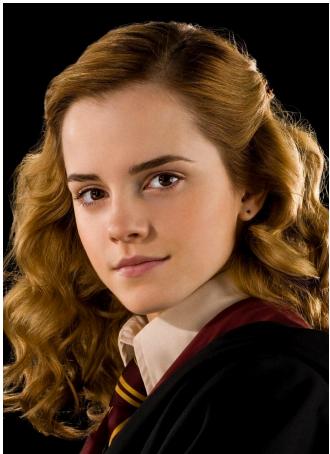
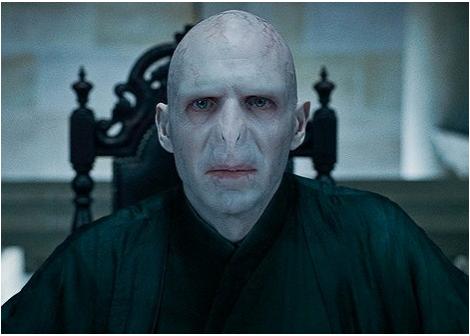


Equivalent of this
many crashes per
day

[1] <http://www.un.org/africarenewal/magazine/may-2013/counterfeit-drugs-raise-africa%20%99s-temperature>

Interesting Puzzle

Counterfeiter



You (Distributor)



User

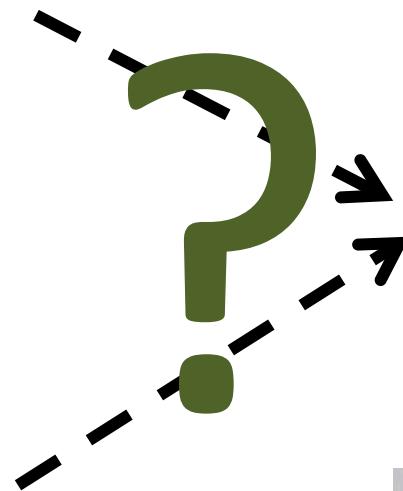


Real Life Challenge

Counterfeiter



You (Distributor)



User



Solution?

Friday

Collections

Vector

Grid

Map

Stack

Queue

Set

Collections

Vector

Grid

Map

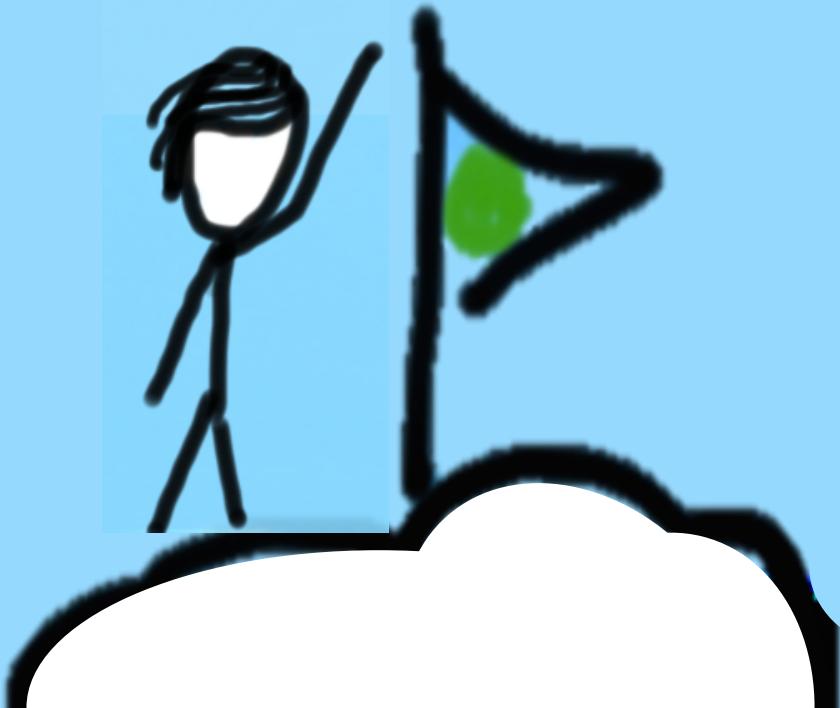
Stack

Queue

Set

Today's Goals

1. Learn how to use Sets
2. Learn how to use Maps



See a previously
never seen before
computer program

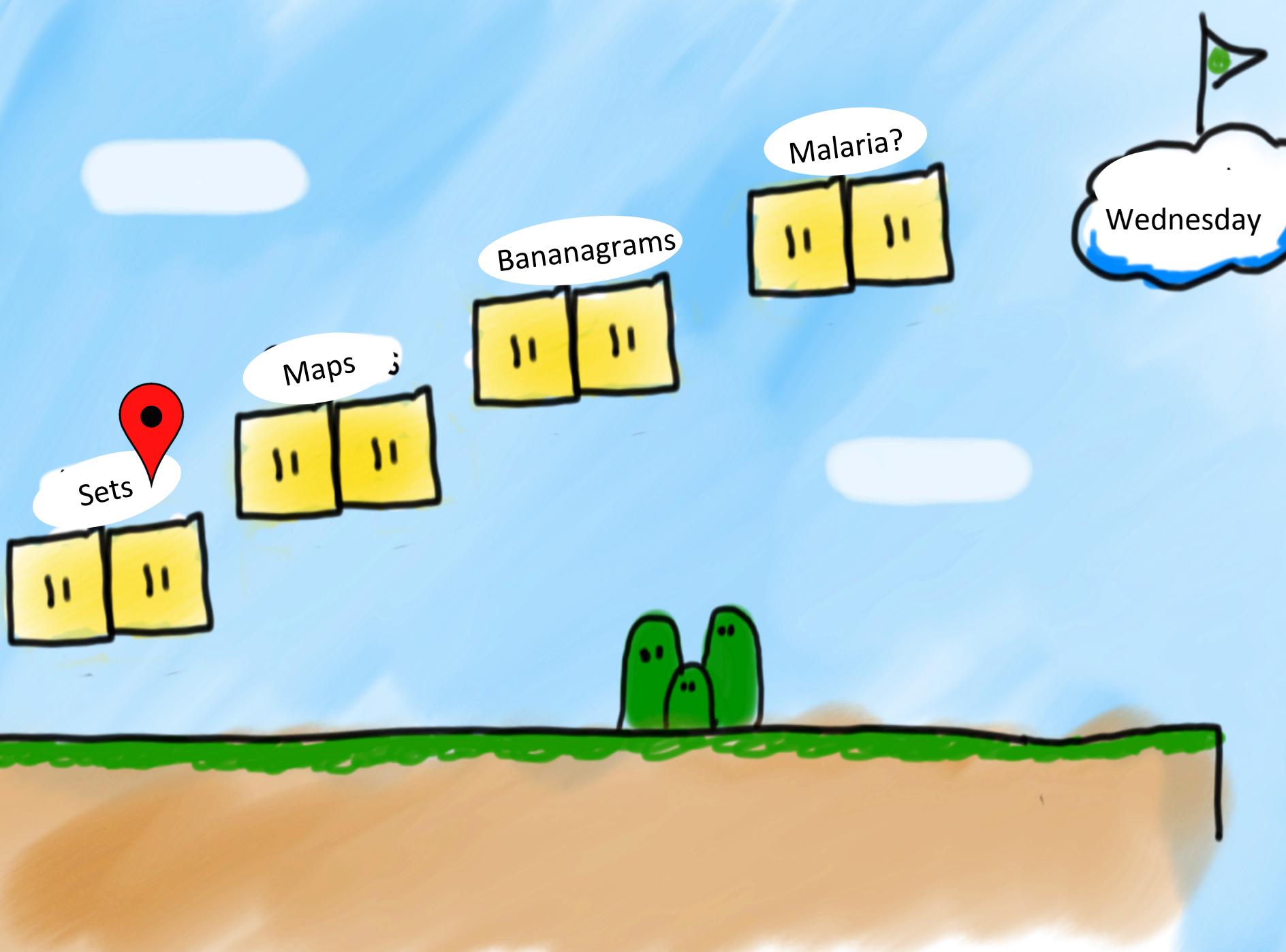
Sets

Maps ;

Bananagrams

Malaria?

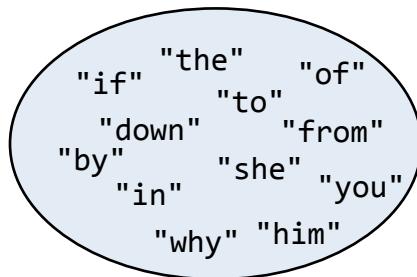
Wednesday



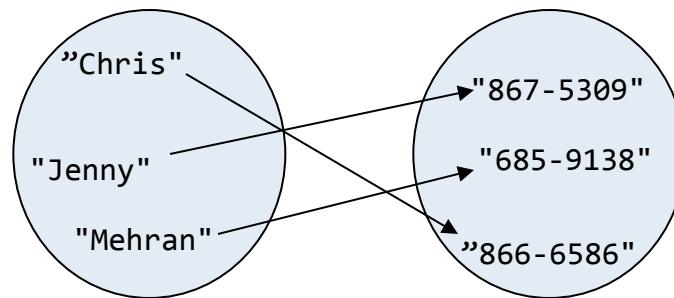
Sets and Maps

Set: Collection of elements with no duplicates;
fast for searching.

Map: Key/value pairs;
Use a key to find its associated value.



set



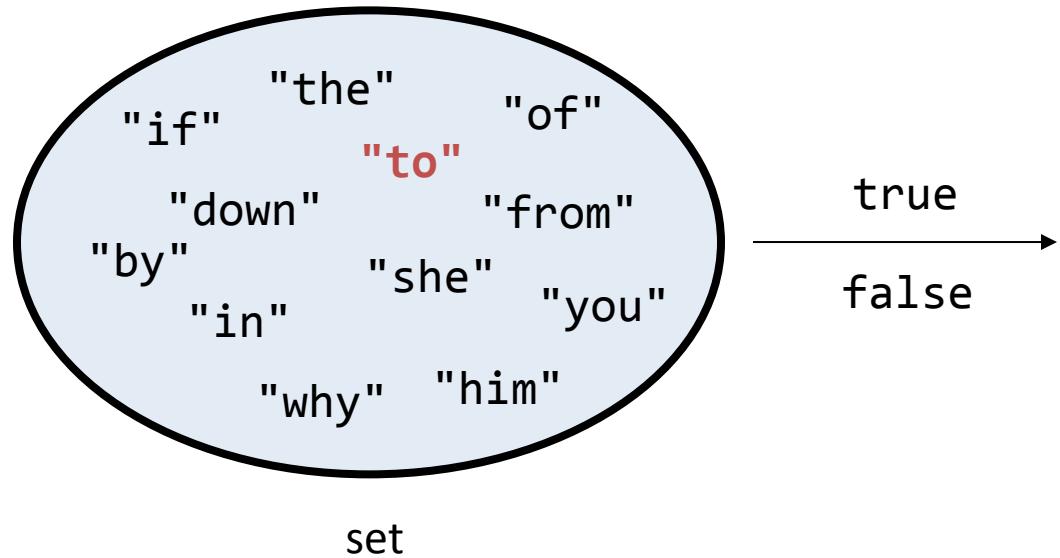
map

Problem #1: Count Unique Words



Sets

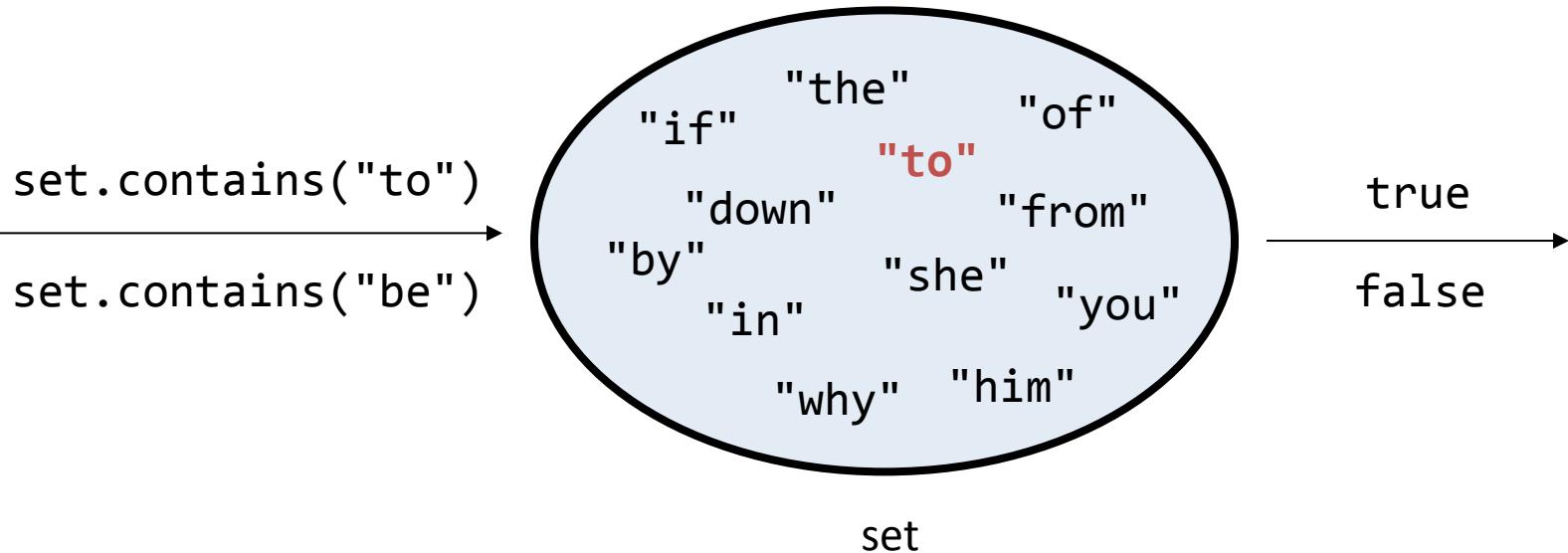
```
set.contains("to")  
----->  
set.contains("be")
```



Sets

Set: A collection of unique values (no duplicates allowed)

- add, remove, search (contains) **Fast**
- We don't think of a set as having any indexes



Looping Over a Set

```
for (type name : collection) {  
    statements;  
}
```

Looping Over a Set

```
for (type name : collection) {  
    statements;  
}
```

The "for-each" loop provides a clean syntax for looping over the elements of a Set, Vector, or most other collections.

Sets have no indexes;
can't use a normal for loop and get each element [i]

```
for (int i = 0; i < set.size(); i++) {  
    do something with set[i];  
}  
} // does not compile
```

The Set Essentials

`set.size()`

Returns the number of elements in the set.

`set.add(value)`

Adds a new value to the set (ignores it if the value already was in the set).

`Set.contains(value)`

Return true if the value is in the set.

```
Set<string> friends;
friends.add("waddie");
friends.add("megan");
cout << friends.contains("voldemort") << endl;
for(string person : friends) {
    cout << person << endl;
}
```

Types of Sets

Set

Iterate over
elements in
sorted order

Really Fast

Implemented using a
binary search tree

HashSet

Element order
is jumbled

Really,
Ridiculously
Fast

Implemented using a
hash table



Set Operands

s1 == s2	true if the sets contain exactly the same elements
s1 != s2	true if the sets don't contain the same elements
s1 + s2	returns the union of s1 and s2 (elements from either)
s1 * s2	returns intersection of s1 and s2 (elements in both)
s1 - s2	returns difference of s1, s2 (elements in s1 but not s2)

Set Operands

```
int main() {
    Set<string> allStudents = loadStanford();

    Set<string> left;
    for(string student : allStudents){
        if(audienceLeft(student)) {
            left.add(student);
        }
    }

    Set<string> hats;
    for(string student : allStudents){
        if(isWearingHat(student)) {
            hats.add(student);
        }
    }

    // more code coming...
}
```

Set Operands

```
int main() {  
    Set<string> allStudents = loadStanford();  
  
    Set<string> left;  
    for(string student : allStudents){  
        if(audienceLeft(student)) {  
            left.add(student);  
        }  
    }  
  
    Set<string> hats;  
    for(string student : allStudents){  
        if(isWearingHat(student)) {  
            hats.add(student);  
        }  
    }  
  
    // more code coming...  
}
```

Students on the audience left of the stage

Set Operands

```
int main() {  
    Set<string> allStudents = loadStanford();  
  
    Set<string> left;  
    for(string student : allStudents){  
        if(audienceLeft(student)) {  
            left.add(student);  
        }  
    }  
}
```

```
Set<string> hats;  
for(string student : allStudents){  
    if(isWearingHat(student)) {  
        hats.add(student);  
    }  
}
```

Students wearing hats

```
// more code coming...
```

```
}
```

Set Operands

allStudents, hats, left

Set Operands

```
allStudents, hats, left
```

```
stand(allStudents);
```


Set Operands

```
allStudents, hats, left
```

```
stand(left);
```


Set Operands

```
allStudents, hats, left
```

```
Set<string> combine;  
combine = left + hats;
```



Elements in
either set

Set Operands

```
allStudents, hats, left
```

```
Set<string> combine;  
combine = left + hats;  
stand(combine);
```


Set Operands

```
allStudents, hats, left
```

```
Set<string> combine;  
combine = left * hats;
```



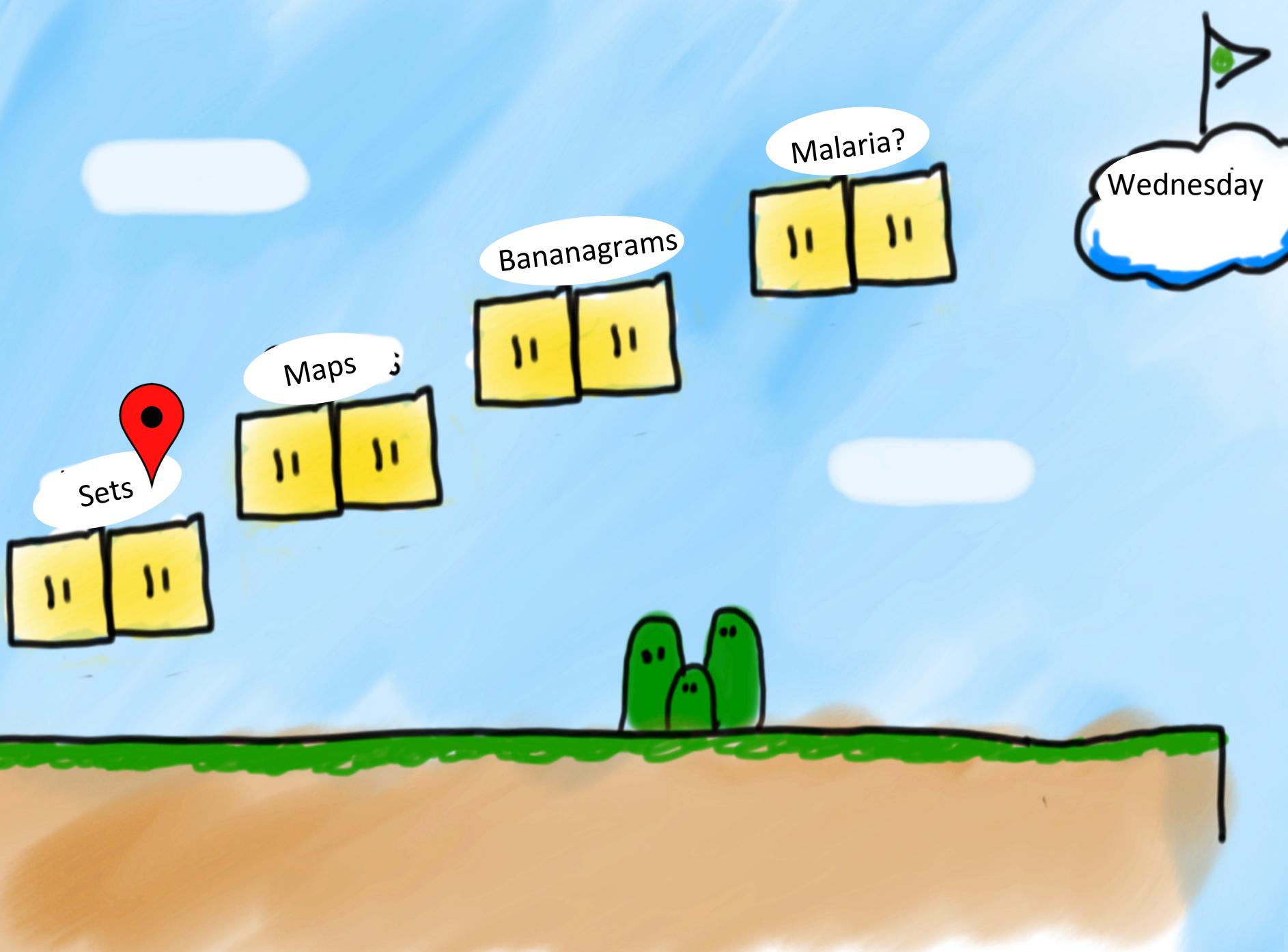
Elements in
both sets

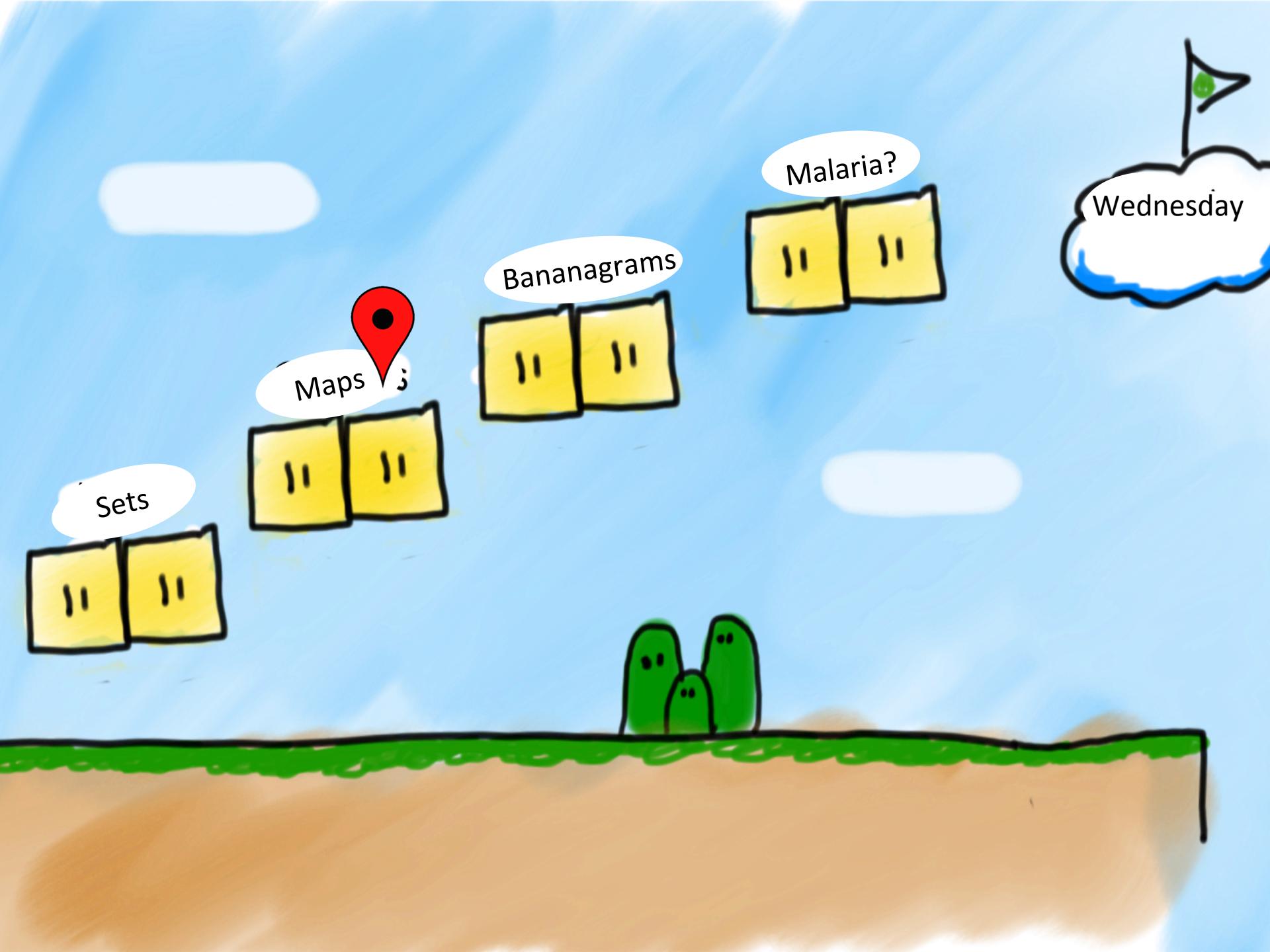
Set Operands

```
allStudents, hats, left
```

```
Set<string> combine;  
combine = left * hats;  
stand(combine);
```

Set Operands



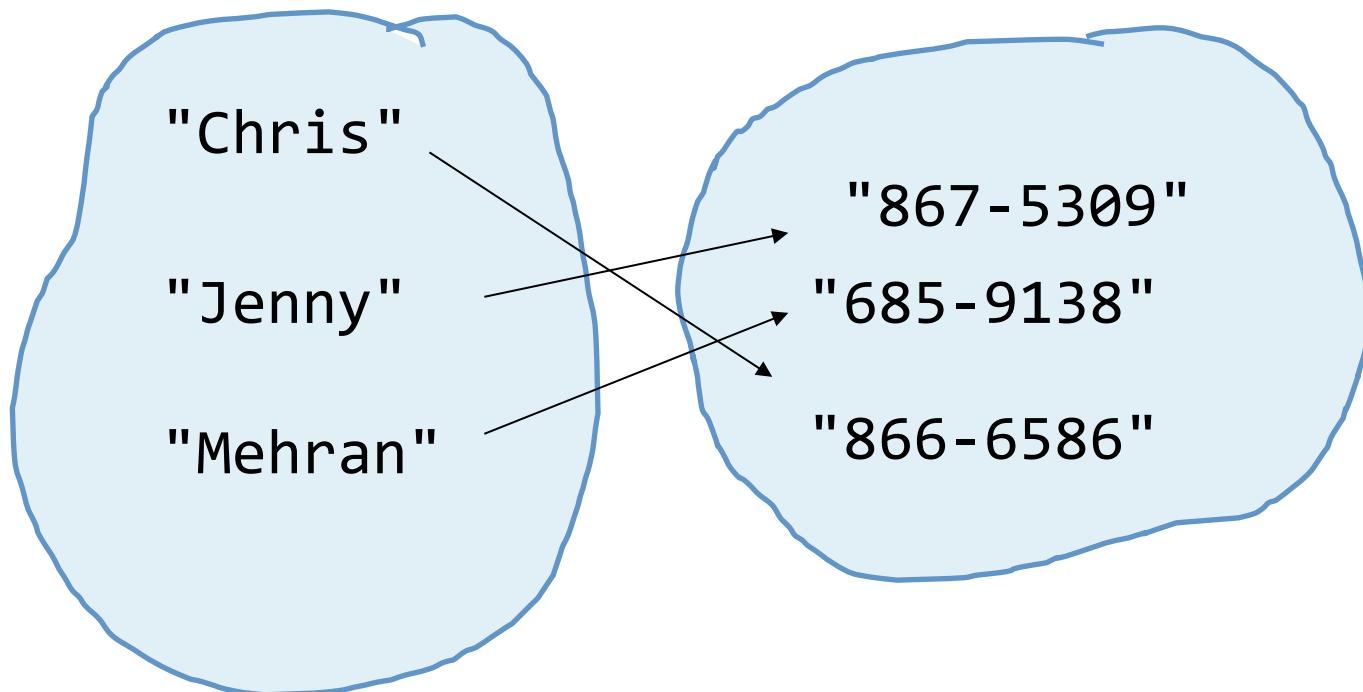


Maps

map: A collection of pairs (k, v), sometimes called key/value pairs, where v can be found quickly if you know k .

a.k.a. dictionary, associative array, hash

a generalization of an array, where the "indexes" need not be ints.

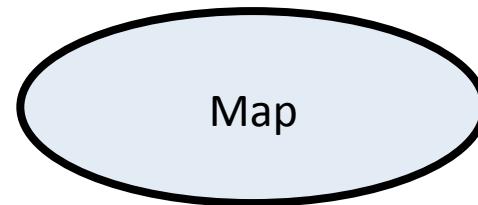
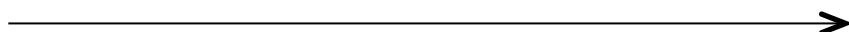


Using Maps

A map allows you to get from one half of a pair to the other.

Store an association from "Suzy" to "206-685-2181".

```
//      key          value
// m["Suzy"] = "206-685-2181";
m.put("Suzy", "206-685-2181");
```

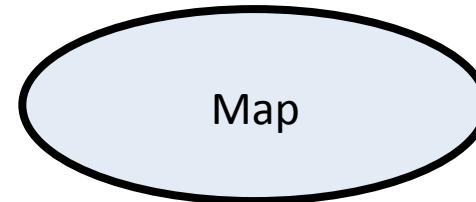
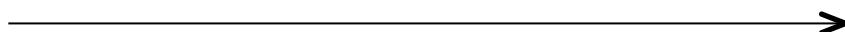


Using Maps

A map allows you to get from one half of a pair to the other.

Store an association from "Suzy" to "206-685-2181".

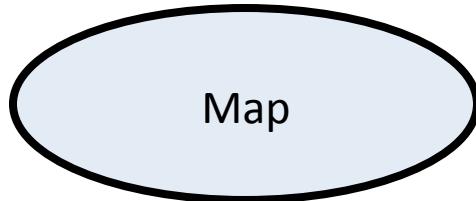
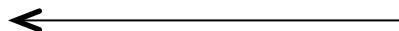
```
//      key          value  
// m["Suzy"] = "206-685-2181";  
m.put("Suzy", "206-685-2181");
```



Map

What was Suzy's phone number?

```
// m["Suzy"]  
m.get("Suzy")
```



Map

"206-685-2181"

Map Example

Key = Title, Value = Article

Key:

“Mantis Shrimp”

Value:

Mantis shrimp

From Wikipedia, the free encyclopedia

Mantis shrimp or stomatopods are marine crustaceans, the members of the order Stomatopoda. Most species can grow to around 10 centimetres (3.9 in) in length, though a few species reach up to 38 cm (15 in).^[2] The largest ever caught has a length of 46 cm (18 in) in the ocean near Fort Pierce, Florida of USA.^[3] The carapace of mantis shrimp covers only the rear part of the head and the first four segments of the thorax. There are more than 400 species of Mantis shrimp. Varieties range from shades of brown to vivid colours, and are among the most important predators in many shallow, tropical and sub-tropical marine habitats. Despite being common, they are poorly understood as many species spend most of their life tucked away in burrows and holes.^[4]



Scientific classification

Kingdom:	Animalia
Phylum:	Arthropoda
Subphylum:	Crustacea
Class:	Malacostraca
Subclass:	Hoplocarida
Order:	Stomatopoda
	Latreille, 1817

Superfamilies and families [1]

Bathysquilloidea

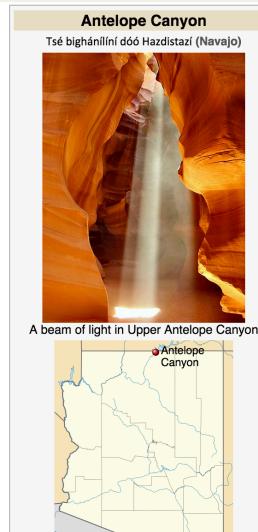
Key:

“Antelope Canyon”

Value:

Antelope Canyon is a slot canyon in the American Southwest. It is located on Navajo land east of Page, Arizona. Antelope Canyon includes two separate, photogenic slot canyon sections, referred to individually as *Upper Antelope Canyon* or *The Crack*; and *Antelope Canyon* or *The Corkscrew*.^[2]

The Navajo name for Upper Antelope Canyon is Tsé bighánílíní, which means "the place where water runs through rocks." Lower Antelope Canyon is Hazdistazí (advertised as "Hasdestwazi" by the Navajo Parks and Recreation Department), or "spiral rock arches." Both are located within the LeChee Chapter of the Navajo Nation.^[4]



Contents [hide]

- 1 Geology
- 2 Tourism and photography
- 2.1 Upper Antelope Canyon

Map Example

Key = Title, Value = Article

Key:

“Mantis Shrimp”

Value:

“<p>Mantis shrimp or stomatopods are marine crustaceans, the members of the order Stomatopoda. though a few species reach up to 38 ...”

Key:

“Antelope Canyon”

Value:

Antelope Canyon is a slot canyon in the American Southwest. It is located on Navajo land east of Page, Arizona. Antelope Canyon includes two separate, photogenic slot canyon sections, referred to individually as *Upper Antelope Canyon* or *The Crack*; and *Antelope Canyon* or *The Corkscrew*.^[2]

The Navajo name for Upper Antelope Canyon is Tsé bighánílíní, which means "the place where water runs through rocks." Lower Antelope Canyon is Hazdistazí (advertised as "Hasdestwazi" by the Navajo Parks and Recreation Department), or "spiral rock arches." Both are located within the LeChee Chapter of the Navajo Nation.^[4]

Contents [hide]

- 1 Geology
- 2 Tourism and photography
- 2.1 Upper Antelope Canyon



A beam of light in Upper Antelope Canyon



Creating a Map

Requires 2 type parameters:
one for keys, one for values.

```
// maps from string keys to  
// integer values  
Map<string, int> votes;
```

Map Members

<code>m.clear();</code>	removes all key/value pairs from the map
<code>m.containsKey(key)</code>	returns true if the map contains a mapping for the given key
<code>m[key]</code> or <code>m.get(key)</code>	returns the value mapped to the given key; if key not found, adds it with a default value (e.g. 0, "")
<code>m.isEmpty()</code>	returns true if the map contains no k/v pairs (size 0)
<code>m.keys()</code>	returns a Vector copy of all keys in the map
<code>m[key] = value;</code> or <code>m.put(key, value);</code>	adds a mapping from the given key to the given value; if the key already exists, replaces its value with the given one
<code>m.remove(key);</code>	removes any existing mapping for the given key
<code>m.size()</code>	returns the number of key/value pairs in the map
<code>m.toString()</code>	returns a string such as "{a:90, d:60, c:70}"
<code>m.values()</code>	returns a Vector copy of all values in the map

Map Example

```
Map<string, string> wiki;  
  
// adds name / text pair to dataset  
wiki.put("Stanford Band", articleHTML);  
  
// returns corresponding articleHTML  
wiki.get("Mantis Shrimp");  
  
// removes the article  
remove("Trump");
```

Types of Maps

Map

Iterate over
elements in
sorted order

Really Fast

Implemented using a
binary search tree

HashMap

Element order
is jumbled

Really,
Ridiculously
Fast

Implemented using a
hash table

Tallies

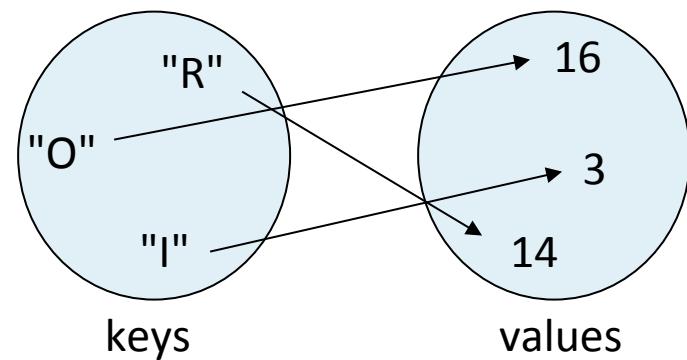
count digits: 22092310907

index	0	1	2	3	4	5	6	7	8	9
value	3	1	3	0	0	0	0	1	0	2

Tallies

// (R)omney, (O)bama, (I)ndependent
count votes:
"R000000RRRRRRO00000ORORRIORRIORIO"

key	"R"	"O"	"I"
value	14	16	3



Problem #2: Tally Words



Looping Over a Map

```
Map<string, double> gpa = load();

for (string name : gpa) {
    cout << name << "'s GPA is ";
    cout << gpa[name] << endl;
}
```

*The order is unpredictable in a HashMap

Sets, good

Maps, good

Sets + Maps, good

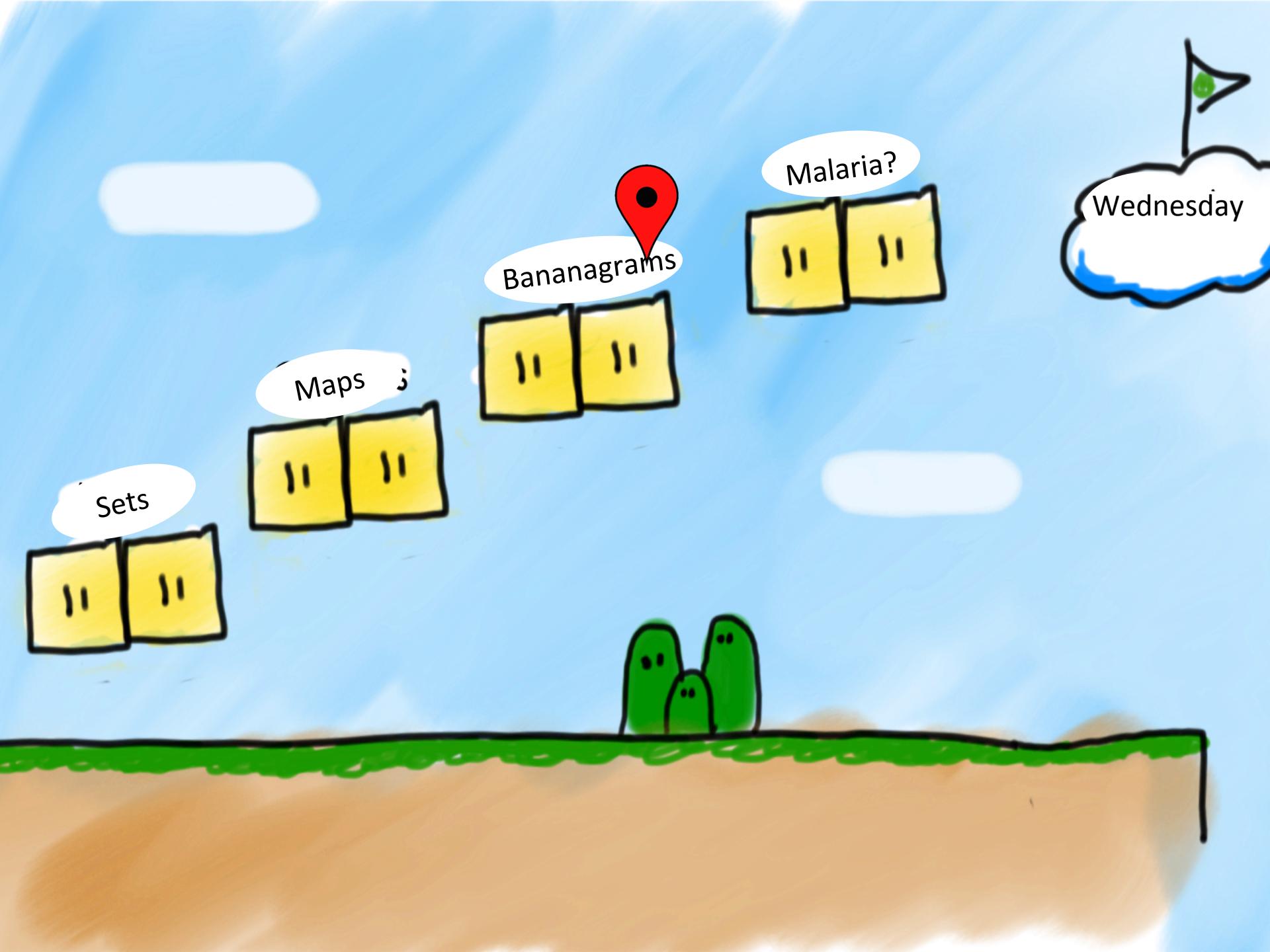
Sets

Maps

Bananagrams

Malaria?

Wednesday



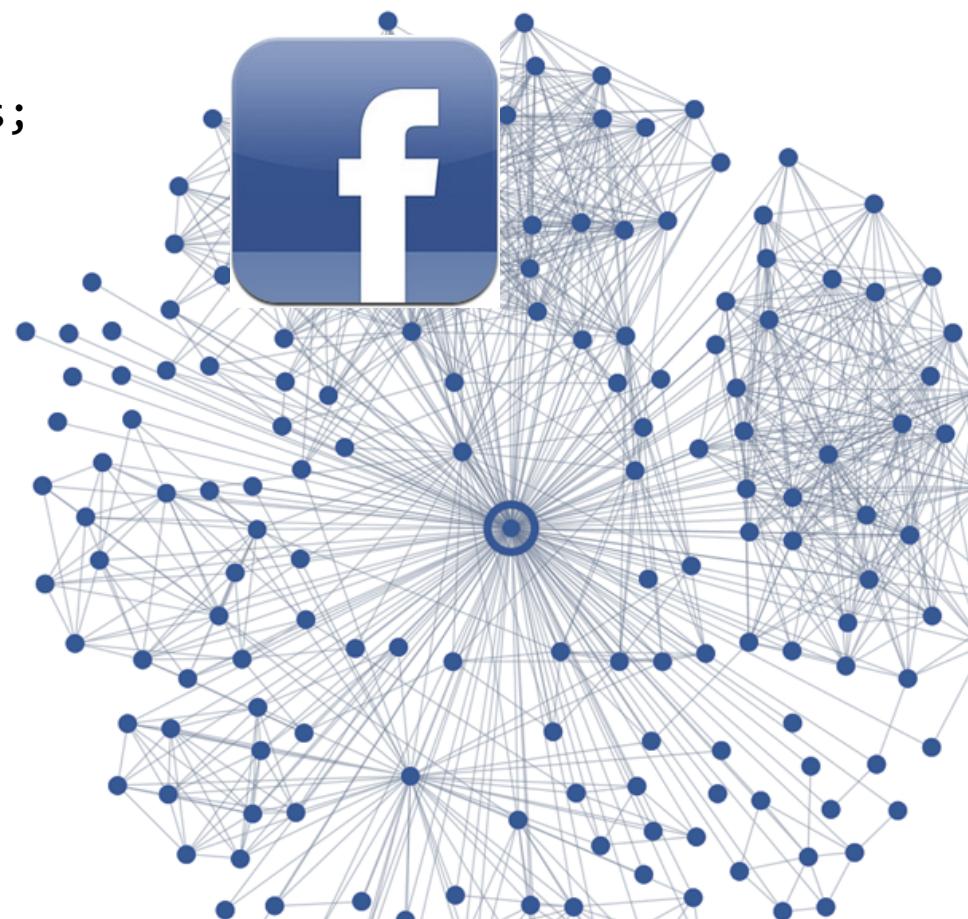
Compound Example

Example: how would you store people's Facebook friends?

- i.e., I need to quickly look up the names of all of Jimmy's buddies, or test whether a given person is a friend of Jimmy's or not.

```
Map<string, Set<string> > pals;
pals["Jim"] += "Kate";
pals["Jim"] += "Bill";
pals["Mar"] += "Stuart";
pals["Jim"] += "Cindy";
pals["Hal"] += "Mar";
pals["Mar"] += "Helene";
cout << pals << endl;

// {Hal:{Mar},
// Jim:{Bill, Cindy, Kate},
// Mar:{Helene, Stuart}}
```



Anagram Exercise

Write a program to find all anagrams of a word the user types.

Type a word [Enter to quit]: scared

Anagrams of scared:

cadres

cedars

sacred

scared



What is the appropriate collection to use to solve this problem?

– *Hint:* Use a compound collection...

Me This Morning

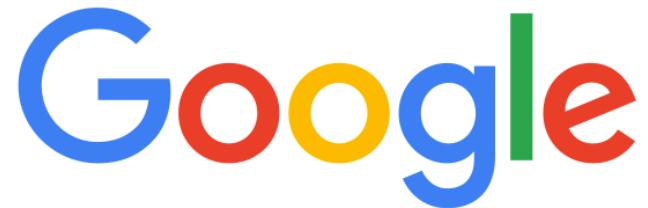
I'm really into this anagram solution. It feels like it can be used for so much more...

soon

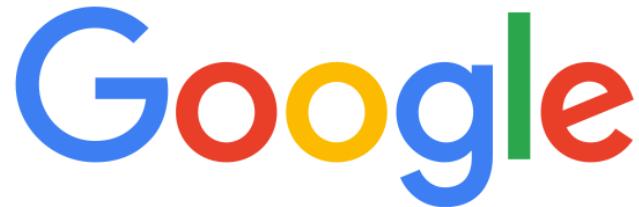


Bananagrams artificial intelligence

Me This Morning



Bananagrams vs computer



Bananagrams online



number of legal moves in bananagrams

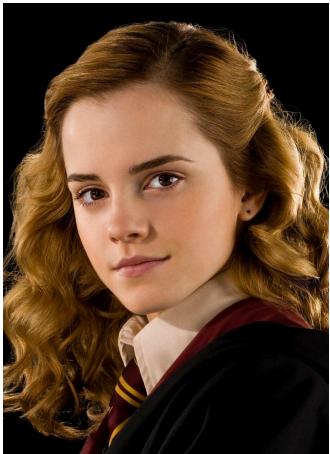
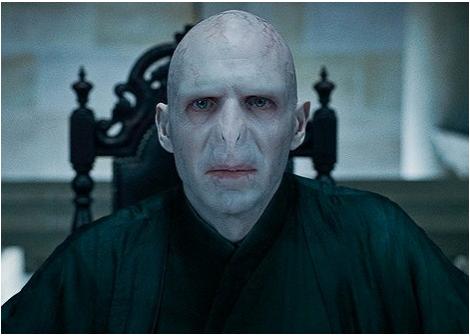
We may be the first...

Bananagrams



Interesting Puzzle

Counterfeiter



You (Distributor)



User

