CS234
MODULE 4 – SETS AND MAPS

• What is a set?
• What is a map?
• How can we implement them?

Updated: 2018-05-07
Sets

A Set ADT is exactly like a mathematical set. It contains things.

- No Duplicates.
- No Order.
- Lots of Mathematical Operations
Set ADT

<table>
<thead>
<tr>
<th>Operation</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set()</td>
<td>contains(item)</td>
</tr>
<tr>
<td>add(item)</td>
<td>remove(item)</td>
</tr>
<tr>
<td>union(other)</td>
<td>intersection(other)</td>
</tr>
<tr>
<td>subtract(other)</td>
<td>difference(other)</td>
</tr>
<tr>
<td>clear()</td>
<td>items()</td>
</tr>
</tbody>
</table>
Operations

- Union: $A \cup B = \text{Red, Blue, or Purple}$
- Intersection: $A \cap B = \text{Purple}$
- Subtract $A - B = \text{Just Blue}$
- Difference $A \triangle B = \text{Red or Blue (no Purple)}$
Choices

Q: Should A union B return a new set, or mutate A?

A1: \_(ツ)_/\ Depends what you’re doing!

A2: 2+2 doesn’t mutate anything, so let’s not have A U B mutate anything.
Uses
Data Structure

OK, what data structure should we use?

Our options are: We have seen: Fixed length arrays, and dynamic arrays.

Dynamic arrays make the most sense. Python has one built-in as the List primitive

(\textit{Python has a Set primitive, but that would be cheating!})
A Set of Courses

Abstract View

Physical View*

(*viewing the list abstractly)
Union Of Two Lists

A + B, Done!

Oh, oops, what about duplicates?

Let’s see what we can do about that.
Intersection of Two Lists

The exact same idea, only instead of “or” we want “and”.

Code is pretty similar.
Subtracting Two Lists?

Just make the loop remove instead of insert!
Symmetric Difference?

Well, we could calculate it more directly, but we could also simply do \((A \cup B) – (A \cap B)\)
Multisets

There’s a related ADT where duplicates are allowed.

What changes to we need to make?

• Even easier, union is just list concatenation
• Or, could have elements be a value plus a count
  • Problems???
Improvements?

We could use a Dictionary primitive instead of a List.

We didn’t because: We don’t know how it works, and not every language has one. But it wouldn’t be complicated.

Just have the Set’s items be the Dictionary’s keys. Associate them all with “True” or something.
Maps

A Map is an ADT that *maps* a set of items (keys) to other items (values).

It’s just another name for a Dictionary. Also sometimes called Tables.

Racket called them Maps, C++ calls them Maps, we’ll call them Maps, too. Mostly to avoid confusion with Python’s primitive *dict* type.
Uses for Maps

So many things.

- Map is best Collection.

- Student Records
  - Keys are student number, values are all the info you want to know
Implementation

What would work out of the structures and ADTs we’ve seen so far?

• Array? Sure
• List? Sure
• Set? Maybe
• Dictionary? That would be cheating
Using Lists

We can have a list of keys and a list of values. Just make sure to keep the indices lined up
Using One List

We could make a Record Class with a “key” and “value” field.
Python Code

Future Dan: Make it so