Sets and Maps

Arrays and lists are straightforward for storing a collection of objects. In this activity, you'll gain experience with two other kinds of collections. Sets and maps are quite useful for implementing a wide variety of algorithms.

Manager: Recorder:

Presenter: Reflector:

Content Learning Objectives

After completing this activity, students should be able to:

- Summarize methods in the `Set` interface.
- Summarize methods in the `Map` interface
- Explain differences between sets and maps.

Process Skill Goals

During the activity, students should make progress toward:

- Discuss results while running code interactively. (Oral Communication)

Model 1 Set of Strings

Type each line of code below in *JShell*, ***one at a time***, and record the results. You only need to record the output to the right of the "==>" symbol. For example, if *JShell* outputs `$8 ==> true`, then just write `true`. If an error occurs, record the error message.

Java code	Shell output
`Set<String> names = new Set<>();` `Set<String> names = new HashSet<>();`	

`names.add("WAS")`
`names.add("BAL")`
`names.add("PHI")`
`names`	

`names.contains("DEN")`
`names.add("DEN")`
`names.contains("DEN")`
`names.contains("den")`	

`names.add("DEN")`
`names.add(123)`
`names.size()`
`names`	

`names.remove("WAS")`
`names.remove("IND")`
`names`	

`names.isEmpty()`
`names.clear()`
`names.size()`
`names.isEmpty()`	

Questions (20 min) Start time:

1. For the collection above:

a) What is the interface name?

b) What is the class name?

c) What is the variable name?

d) What is the element type?

2. Based on the shell output, describe what the following methods return:

   a) `add`

   b) `remove`

3. Consider the contents of `names` just before `"WAS"` was removed.

   a) What was the size of `names` at this point?

   b) How many times was the `add` method called?

   c) Explain why these two numbers are different.

4. Continuing the previous question:

   a) In what order were the strings added to the set?

   b) In what order were they displayed in the output?

   c) Why do you think the two orders are different?

5. In your own words, summarize what a `Set` is in Java. Give an example from everyday life.

6. In discrete mathematics, sets have three basic operations:

   - Union ($S \cup T$) : all elements in S or T (or both)
   - Intersection ($S \cap T$) : elements in both S and T
   - Difference ($S - T$) : elements in S but not in T

Based on the documentation for `java.util.Set`, which methods implement these operations?

Model 2 Map of Team Names

The following abbreviations are for National Football League (NFL) teams:

ATL	Atlanta Falcons
DEN	Denver Broncos
IND	Indianapolis Colts
MIA	Miami Dolphins
SEA	Seattle Seahawks

Complete the table below using *JShell* (the same way you did for Model 1).

Java code	Shell output

`Map<String, String> teams;`
`teams = new Map<>();`
`teams = new HashMap<>();`
`teams.isEmpty()`	

`teams.put("MIA", "Miami Dolphins")`
`teams.put("MIA", "Miami")`
`teams.size()`
`teams`	

`teams.put("ATL", "Atlanta")`
`teams.put("SEA", "Seattle")`
`teams`	

`teams.containsKey("ATL")`
`teams.containsKey("DEN")`
`teams.containsValue("Miami")`
`teams.containsValue("Dolphins")`	

`teams.get("SEA")`
`teams.get("IND")`
`teams.get(0)`	

`teams.remove("MIA")`
`teams.remove("MIA")`
`teams`	

`teams.keySet()` `teams.values()`	

Questions (25 min) Start time:

7. For the collection above:

a) What is the interface?

b) What is the class?

c) What type of keys?

d) What type of values?

8. Based on the shell output, describe what the following methods return:

a) `put`

b) `get`

9. What type of object does the `keySet` method return? Describe its contents.

10. What type of object does the `values` method return? Describe its contents.

11. In your own words, summarize what a `Map` is in Java. Give an example from everyday life.

12. Why did `teams.get(0)` return null, even though there were values in the map?

13. Write Java code that defines a map named `dow` that represents the seven days of the week as follows: Sun=1, Mon=2, Tue=3, etc. Run your code in *JShell* to make sure it works.

14. Print the `dow` variable in *JShell*. What do you notice about the order of its contents?