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<>();`	java.util.Set is abstract; cannot be instantiated
`Set<String> names = new HashSet<>();`	[]
`names.add("WAS")`	true
`names.add("BAL")`	true
`names.add("PHI")`	true
`names`	[PHI, WAS, BAL]
`names.contains("DEN")`	false
`names.add("DEN")`	true
`names.contains("DEN")`	true
`names.contains("den")`	false
`names.add("DEN")`	false
`names.add(123)`	int cannot be converted to java.lang.String
`names.size()`	4
`names`	[PHI, WAS, DEN, BAL]
`names.remove("WAS")`	true
`names.remove("IND")`	false
`names`	[PHI, DEN, BAL]
`names.isEmpty()`	false
`names.clear()`	
`names.size()`	0
`names.isEmpty()`	true

Questions (20 min) Start time:

1. For the collection above:

a) What is the interface name? Set

b) What is the class name? HashSet

c) What is the variable name? names

d) What is the element type? String

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

   a) `add` true if this set did not already contain the specified element

   b) `remove` true if this set contained the specified element

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

   a) What was the size of `names` at this point? 4

   b) How many times was the `add` method called? 6

   c) Explain why these two numbers are different.

   DEN was added a second time, but it was already in the set.
   123 could not be added, because its type didn't match.

4. Continuing the previous question:

   a) In what order were the strings added to the set? WAS, BAL, PHI, DEN

   b) In what order were they displayed in the output? PHI, WAS, DEN, BAL

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

   Sets have no defined order

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

The `java.util.Set` interface models the mathematical notion of a ***set***, i.e., a collection that has no duplicates. For example, you could have the set of all computer science majors.

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?

```
set1.addAll(set2);    // Union
set1.retainAll(set2); // Intersection
set1.removeAll(set2); // Difference
```