12.1 Lists and Sets in Java

In this section, we will learn about how to use Java's built-in `List` and `Set` data structures as well as build our own `ArraySet`.

Getting Started

In this course, we've already built two kinds of lists: `AList` and `SLList`. We also built an interface `List61B` to enforce specific list methods `AList` and `SLList` had to implement. You can find the code at the following links:

- `List61B`
- `AList`
- `SLList`

This is how we might use `List61B` type:

```
List61B<Integer> L = new AList<>();
L.addLast(5);
L.addLast(10);
L.addLast(15);
L.print();
```

Lists in Real Java Code

We built a list from scratch, but Java provides a built-in `List` interface and several implementations, e.g. `ArrayList`. Remember, since `List` is an interface we can't instantiate it! We must instantiate one of its implementations.

To access this, we can use the full name ('canonical name') of classes/interfaces:

```
java.util.List<Integer> L = new java.util.ArrayList<>();
```

However, this is a bit verbose. Instead, we can import java libraries:

```
import java.util.List;
import java.util.ArrayList;

public class Example {
    public static void main(String[] args) {
        List<Integer> L = new ArrayList<>();
        L.add(5);
        L.add(10);
        System.out.println(L);
    }
}
```

Sets

Sets are a collection of unique elements - you can only have one copy of each element. Unlike Lists, there is also no sense of order: you can't index into a set, nor can you control where each element is inserted into the set.

Java Sets

Java has the `Set` interface along with implementations, e.g. `HashSet`. Remember to import them if you don't want to use the full name!

```
import java.util.Set;
import java.util.HashSet;
```

Example use:

```
Set<String> s = new HashSet<>();
s.add("Tokyo");
s.add("Lagos");
System.out.println(s.contains("Tokyo")); // true
```

Python Equivalent

In python, we simply call `set()`. To check for `contains` we don't use a method but the keyword `in`. Here's an example:

```
s = set()
s.add("Tokyo")
s.add("Lagos")
print("Tokyo" in s) // True
```

DIY: ArraySet

Our goal is to make our own set, `ArraySet`, with the following methods:

- `add(value)`: add the value to the set if not already present
- `contains(value)`: check to see if ArraySet contains the key
- `size()`: return number of values

If you would like to try it yourself, find 'Do It Yourself' `ArraySet starter code` here. In the lecture clip below, Professor Hug goes develops the solution:

Previous
12. Inheritance IV: Iterators, Object Methods

Next
12.2 Exceptions

Last updated 5 months ago