Lesson 26: `ArrayList` (W08D1)

Balboa High School

Michael Ferraro

October 5, 2015

Do Now

Prepare PS #4a (paper form) for pick-up!

Consider the code below for `powIter()`, an iterative algorithm that raises a base, a, to the specified power, b. E.g., `powIter(2,3)` computes 2^3, returning 8.

```
public static int powIter(int a, int b) {
    int result = a;
    while ( ??? ) {
        result *= a;
        b--;
    }
    return result;
}
```

→ **Using a paper table**, determine the stopping condition to use for the `while()` loop. Afterward, verify using Java.

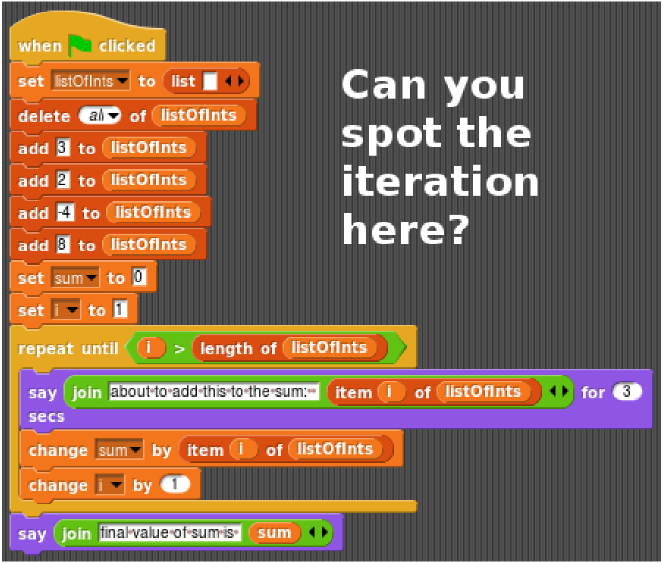
Aim

Students will learn how `ArrayLists` work, and how iterative and recursive procedures may take advantage of them.

What's a list?

- Run this Snap! program:
  - save this file: `list_adder.xml`
  - import into Snap!
- Then answer the following:
  - What does a list appear to be?
  - What is the purpose of `i`?

Iterating through a List



```
when ⚑ clicked
set listOfInts to list
delete all of listOfInts
add 3 to listOfInts
add 2 to listOfInts
add -4 to listOfInts
add 8 to listOfInts
set sum to 0
set i to 1
repeat until (i > length of listOfInts)
    say join about to add this to the sum:  item i of listOfInts for 3 secs
    change sum by item i of listOfInts
    change i by 1
say join final value of sum is  sum
```

Can you spot the iteration here?

What is an `ArrayList`?

- An `ArrayList` is a Java object that can hold a set (or *list*) of objects that may be:
    - added
    - removed
    - iterated through
    - . . . and more (later)

What is an `ArrayList`?

- An `ArrayList` is a Java object that can hold a set (or *list*) of objects that may be:
    - added
    - removed
    - iterated through
    - . . . and more (later)
- It's a Java class, complete with an API here

What is an `ArrayList`?

- An `ArrayList` is a Java object that can hold a set (or *list*) of objects that may be:
  - added
  - removed
  - iterated through
  - ...and more (later)
- It's a Java class, complete with an API here
- `ArrayList` is part of the `java.util` package, so when you use this class, you need to have this `import` statement:

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

Our first ArrayList

Let's create an ArrayList to hold String objects.

```
import java.util.ArrayList;

public class ArrayListOne { //create file in Lesson26 project!

    public static void main(String[] args) {
        ArrayList<String> listA = new ArrayList<String>();

    }
}
```

Our first ArrayList

Let's create an ArrayList to hold String objects.

```
import java.util.ArrayList;

public class ArrayListOne { //create file in Lesson26 project!

    public static void main(String[] args) {
        ArrayList<String> listA = new ArrayList<String>();

        String oneStr = "one";
        listA.add(oneStr);

    }
}
```

Our first `ArrayList`

Let's create an `ArrayList` to hold `String` objects.

```
import java.util.ArrayList;

public class ArrayListOne { //create file in Lesson26 project!

    public static void main(String[] args) {
        ArrayList<String> listA = new ArrayList<String>();

        String oneStr = "one";
        listA.add(oneStr);
        listA.add("two");
        listA.add("three");

    }
}
```

Our first ArrayList

Let's create an ArrayList to hold String objects.

```
import java.util.ArrayList;

public class ArrayListOne { //create file in Lesson26 project!

    public static void main(String[] args) {
        ArrayList<String> listA = new ArrayList<String>();

        String oneStr = "one";
        listA.add(oneStr);  //adding a NAMED String
        listA.add("two");   //adding an ANONYMOUS String
        listA.add("three");

    }
}
```

Our first ArrayList

Let's create an ArrayList to hold String objects.

```
import java.util.ArrayList;

public class ArrayListOne { //create file in Lesson26 project!

    public static void main(String[] args) {
        ArrayList<String> listA = new ArrayList<String>();

        String oneStr = "one";
        listA.add(oneStr);
        listA.add("two");
        listA.add("three");

        System.out.println(listA); //what *method* must
    }                              //ArrayList provide?
}
```

Let's Iterate through `listA`

Comment out `System.out.println()` and replace with:

```
for ( String curStr : listA ) {
    System.out.println("current element:\t" + curStr);
}
```

Let's Iterate through `listA`

Comment out `System.out.println()` and replace with:

```
for ( String curStr : listA ) {
    System.out.println("current element:\t" + curStr);
}
```

- This form of a `for()` loop is a *for-each* loop:

  for each `String` *in* `listA`, *do . . .*

Let's Iterate through `listA`

Comment out `System.out.println()` and replace with:

```
for ( String curStr : listA ) {
    System.out.println("current element:\t" + curStr);
}
```

- This form of a `for()` loop is a *for-each* loop:

  for each `String` *in* `listA`, *do . . .*

- Iterating through an `ArrayList` this way does **not** modify the object; this is simply a way to *inspect* the contents of the `ArrayList`.

Let's Add Integers

We are now going to iterate through a list of `Integers`, returning the sum.

[1] This is due to a relatively new Java feature called *autoboxing* that we might discuss later.

Let’s Add Integers

We are now going to iterate through a list of `Integers`, returning the sum.

- `ArrayLists` hold **objects, not primitive variables** like `ints`; that’s OK since there’s a Java class called `Integer`, which will work the same for us[1]

[1] This is due to a relatively new Java feature called *autoboxing* that we might discuss later.

Let's Add Integers

We are now going to iterate through a list of `Integers`, returning the sum.

- `ArrayLists` hold **objects, not primitive variables** like `ints`; that's OK since there's a Java class called `Integer`, which will work the same for us[1]
- Download `ArrayListTwo.java` from here and import into `Lesson26`
- Read the code for the program, ask questions about what you don't understand

[1] This is due to a relatively new Java feature called *autoboxing* that we might discuss later.

Two-Minute Exercise

Quick: Go to the API for `ArrayList` here and find a method that will return the length of a list. Modify `ArrayListTwo` to print the length of the list.

And in Strolls *Recursion*. . .

- How might you think of the adding problem recursively?

And in Strolls *Recursion*. . .

- How might you think of the adding problem recursively?

The sum of the list $= \boxed{\boxed{\text{first element}} + \text{the sum of the rest of the elements}}$

And in Strolls *Recursion*. . .

- How might you think of the adding problem recursively?

The sum of the list = first element + the sum of the rest of the elements

- We need a way to get the first element off of a list: `remove()` method[2]

```
ArrayList<Integer> myList = new ArrayList<Integer>();
myList.add(6);
myList.add(7);
myList.add(8);
System.out.println(myList);
int firstElt = myList.remove(0); //0th elt is the first one
System.out.println("sum = " + firstElt
    + " + " sumOf" + myList);
```

[2] NOTE: `remove()` **modifies** the `ArrayList` object forever!

Classwork

- You've been given most of the basics of `ArrayList` that you need for PS #4b
- §5.1 & §5.4 of PS #4b should already be done (or in progress)
- Finish §§5.2–5.3, inclusive, of PS #4b
- Already done with those sections? Write a recursive procedure to take an `ArrayList` of one-character `String`s and return it in reversed form as a single `String`

For example, take an `ArrayList` like this one:

"a"	"b"	"c"	"d"	"e"	"f"

and return "fedcba"

HW

Finish all of §5 of PS #4b.