Opening Discussion

- Minute essay comments:
  - Scala syntax compared to other languages.
  - Why fill passed \((a, i+1, x)\)?
  - If you don't know how many values you will have, arrays are not ideal.
  - When to use Lists vs. Arrays.
  - Setting using multiple values.
  - All quizzes and tests are on paper. Coding is for IcPs and sssignments.
  - Uses for match.
More

- Getting lots of data into vi.
- Submitting assignments.
Recap Arrays and Lists

- **Creation**
  - Array(5,7,4)
  - List(8,5,3)
  - new Array[Double](1000)
  - 1::2::3::Nil

- **Comparison**
  - Arrays: mutable, fixed size.
  - Lists: immutable, :: to make new, longer list

- **Indexing:** start at 0
  - arr(5), arr(5)=”hi”
Using Lists

- You can do direct access on lists, but it is inefficient.
- The better method is to use the head and tail methods.
- The elements in a list can't be changed. However, you can efficiently add new elements at the front to make a new list.
- Lists work very well with recursion.
You can make patterns with Lists and Arrays.

For Arrays:
- Array(1,2,a,b,c)

For Lists:
- List(1,2,a,b,c)
- h::t - matches any non-empty list
- Nil - matches an empty list
There are lots of methods on collections. The API can help us see all of them.

Part of collections:
- drop, init, last, slice, splitAt, take, takeRight

Boolean tests:
- contains, endsWith, isEmpty, nonEmpty, startsWith

Searching:
- indexOf, lastIndexOf

Other:
- mkString, reverse, zip, zipWithIndex
Other Methods

- If the elements in a list support addition or multiplication, you can use the sum and product methods.
- If they are ordered you can do min and max.
- Having sum and length makes averages really easy.
- With min you can even drop a grade easily.
Higher Order Methods

- The most powerful methods are ones you can pass functions into.
  - exists, forall – Boolean checks like for math.
  - filter, partition – separate collection based on Boolean.
  - map – apply function to all the elements.
  - reduceLeft – apply function moving through collection
  - foldLeft – apply function moving through, but allows initial value so it can return a different type. This is curried.
Let's Put These Into Action

- I want to spend the rest of the class time playing with these methods and seeing what we can do with them.
- A String is a collection so you can do these things with a String as well.
- String also has a method called split.
What questions do you have?
Quiz #3 on Monday.
Basic Arrays and Lists

- The two most basic collection types in Scala are arrays and lists.
- We can make either by following the type name with a parenthesized list of elements.
- Can create an “empty” array using new.
- Can build Lists with :: operator. Nil is empty.

Comparison
- Arrays are mutable, but fixed in size.
- Lists are immutable, but it is easy to add an element and get a new list.
You should notice that when we make an array or a list, the type is followed by square brackets.

These types are parametric. So they take type arguments.

In Scala, type parameters are placed in square brackets.
We can get to the elements in an array by putting an index in parentheses. The index is 0-referenced.

- `arr(5)`

This syntax can be used in expressions to read values.

It can also be used in assignments to store values in the array. This is what it means to be mutable.

Let's look at some examples of this.
Using Lists

- You can do direct access on lists, but it is inefficient.
- The better method is to use the head and tail methods.
- The elements in a list can't be changed. However, you can efficiently add new elements at the front of the list.
- Lists work very well with recursion.
List and Array Patterns

- You can make patterns with Lists and Arrays.
- For Arrays:
  - Array(1,2,a,b,c)
- For Lists:
  - List(1,2,a,b,c)
  - h::t  - matches any non-empty list
  - Nil  - matches an empty list
Questions?

- The first assignment is due Friday by midnight, but you might want to aim for earlier as you might find it hard to submit outside this building.
- I will show you how to submit on Friday in class.