ArrayLists

Array Lists

The Java Collections Framework gives implementations for some standard data structures. We will do our own versions of several of these as we study them.

In Lab 2 we will re-implement the ArrayList class.

Example 1

I want to write a program that reads a bunch of numbers (integers) from the user, then at the end prints them out. To do this in C we would use an array, and hope it is big enough to hold the data. If we run out of room we could make a bigger array.

Python would solve this with a list. In Java we have **ArrayLists** to hold the data. ArrayList is a generic class that takes a type parameter:

```
public class ArrayList<E> {
        ....
}
```

There are many useful methods of class ArrayList, but we will focus here on a few. If L is an ArrayList, then

- L.size() is the number of entries in L
- L.add(x) appends x to the end of the list
- L.add(i, x) adds x to the list at positon i, shifting the tail of the list back one to make room.
- L.get(i) is the element at position i.

This makes our program very simple. We start by creating the ArrayList:

```
ArrayList<Integer> L = new ArrayList<Integer>();
```

Each time we get a new data value x we add it to the list:

```
L.add(x);
```

At the end we print the list: using either

```
for (int x: L )
        System.out.println(x)
```

or, if we want to use indices

```
for (int i = 0; i < L.size(); i++)
        System.out.println( L.get(i) );
```

For the complete code see SimpleList.java