What can you put into an ArrayList?

- Anything that derives from Object
  - String
  - ArrayList
  - Counter
  - int, double, char, boolean, … NO!!! Not Directly

- What can we do if want a collection of int values
  - Put an upper bound on the count and use array

```java
int index = 0;
int[] a = new int[HUGE_NUMBER];
while (...) {
    if (condition()){
        a[index++] = value;
    }
}
```

What can you put into an ArrayList?

- Use a wrapper class (see java.lang.*)
  - int, double, char, boolean, ...
  - Integer, Double, Character, Boolean
- Can have your cake and eat it too
  ```java
  ArrayList<Integer> list = new ArrayList<Integer>();
  for (int k = 0; k < 10; k++){
      ints.add(k*k);
  }
  for (Integer jj : ints){
      System.out.println(jj);
  }
  ```
- All made practical by Version 5 of Java

Java 5.0 ArrayLists: Generics

- Can specify the Class stored in the ArrayList
- Constructors and mutators for demo class ALists
  ```java
  public class ALists { //side by side: old and new
    ArrayList a; ArrayList<String> aS;
    public ALists(){
        a = new ArrayList();
        aS = new ArrayList<String>();
    }
    public void aAdd(String s){
        a.add(s);
    }
    public void aGet(String s){
        a.add(s);
    }
    public void aSSGet(String s){
        aS.add(s);
    }
  }
  ```

Java 5.0 ArrayLists: Generics

- Accessor Methods
  ```java
  public String aGet(int k){
      return (String) a.get(k);
  }
  public String aSGet(int k){
      return aS.get(k);
  }
  public int aSize(){
      return a.size();
  }
  public int aSSize(){
      return aS.size();
  }
  ```
Java 5.0 ArrayLists: Generics

- Test our Code (use new form of for statement)

```java
public static void main(String[] args){
    String[] tStrings = {"one", "two", "three", "four", "five","six"};
    ALists aa = new ALists();
    for (String s: tStrings){
        aa.aAdd(s);
        aa.asAdd(s);
    }
    for (int k = aa.aSize() - 1; k >= 0; k--){
        System.out.println(aa.aGet(k) + " \t" + aa.asGet(k));
    }
}
```

- Exploring ArrayLists
  - Look at the Java 5.0 API
  - Note interfaces implemented
    - Serializable, Cloneable, Iterable
    - Collection, List, RandomAccess
  - Note other descriptive text
    - Regarding performance
    - Constructors
    - Methods
    - Don’t forget methods in parent classes

New Form of for Statement

- Form

```java
for (Class s: agregate){
    ...
    ...s... //use s -- next value each iteration
    ...
}
```

- What is required of agregate to allow this kind of access?
  - Class must implement iterable (irratable? :-)

Exploring ArrayLists

- Some Commonly Used Methods
  - boolean add(E o) // append
  - void add(int index, E element) // insert
  - void Clear()
  - boolean contains(Object elem)
  - E get(int index)
  - int indexOf(Object elem)
  - boolean remove(Object o)
  - E remove(int index)
  - E set(int index)
  - int size()
Exploring ArrayLists

- **Performance**
  - **Constant Time**
    - size, isEmpty, get, set, iterator, listIterator operations
  - **Linear Time**
    - All of the other operations run in linear time
- **What does all of this mean?**
- **Why do we care?**
- **Exercise:** Implement on an array the equivalent of
  1. void add(int index, E element)
  2. E remove(int index)
- **Remember:** Memory is an array (well sort of)

What is a char?

- **Differences between unicode and ASCII**
  - Why is unicode used? Why should we care? What should we know? How many of the details are important?
- **A char value can be treated like an int value**
  - Add integer to it, cast back to char
  - Subtract character from it, get int back
  - counters[‘z’ - s.charAt(k)]++;
- **Anatomy of the statement above??**

Lydia Kavraki

- **Awards**
  - Grace Murray Hopper
  - Brilliant 10
  - "I like to work on problems that will generally improve the quality of our life."
  - What’s the thing you love most about science?
  - “Working with students and interacting with people from diverse intellectual backgrounds. Discovery and the challenge of solving a tough problem, especially when it can really affect the quality of our lives. I find the whole process energizing.”

View Model Communication

- **View interacts with user**
  - Load file
  - Start new game
  - User guesses word (when?)
  - User responds with # letters in common (when?)
Model View Communication

- **Informative messages**
  - I know 5432 words
    - `showMessage(...)`

- **Messages requiring interaction (modal dialog)**
  - You guessed my word
  - You’ve guessed that word before (or …?)
    - `showModalInfo(...)`

- **Model changes and notifies view**
  - I’m thinking of a secret word
  - Your word has 4 letters in common, you’ve used one guess
    - `processModelResponse(...)`

Inheritance and models

- **We’ll see other models in other programs**
  - What interface in common?

- **AbstractModel implements code and calls unimplemented methods**
  - Who implements these methods?
  - Hollywood principle, template pattern