The ArrayList Class

- Similar to an array, an ArrayList allows object storage
- Unlike an array, an ArrayList object:
  - Automatically expands when a new item is added
  - Automatically shrinks when items are removed
- Requires:

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

Creating an ArrayList

```
ArrayList<String> nameList = new ArrayList<String>();
```

Notice the word String written inside angled brackets <>

If we try to store any other type of object in this ArrayList, an error will occur.

Using an ArrayList

- To populate the ArrayList, use the add metho
  - nameList.add("James");
  - nameList.add("Catherine");
- To get the current size, call the size method
  - nameList.size(); // returns 2
- To access items in an ArrayList, use the get method

```
nameList.get(1);
```

Using an ArrayList

- The ArrayList class's toString method returns a string representing all ArrayList

```
System.out.println(nameList);
```

This statement yields :

```
[ James, Catherine ]
```

- The ArrayList class's remove method removes designated item fron

ArrayList

```
nameList.remove(1);
```

This statement removes the second item.

Using an ArrayList

- The ArrayList class's add method with one argument adds new items end of the ArrayList
- To insert items at a location of choice, use the add method with two argu

```
nameList.add(1, "Mary");
```

This statement inserts the String "Mary" at index 1

To replace an existing item, use the set method:

```
nameList.set(1, "Becky");
```

This statement replaces "Mary" with "Becky"

Using an ArrayList

- An ArrayList has a capacity, which is the number of items it ca increasing its size.
- The default capacity of an ArrayList is 10 items.
- To designate a different capacity, use a parameterized constructor:

```
ArrayList<String> list = new ArrayList<String>(100);
```