Generic types for ArrayList

- Old Java (1.4 and older):

```
ArrayList strings = new ArrayList();
strings.add(“hello”);
String word = (String) strings.get(0);
```

- New (since 1.5):

```
ArrayList<String> strings = new ArrayList <String>();
strings.add(“hello”);
String word = strings.get(0);
```

Advantages

- Better readability
- Better type-safety: no casts (runtime checks), compiler can catch problems

Writing your own generic code

- Formal type parameter

public class Stack<E> { … }

  - convention: Short (single-char) uppercase
  - can be used wherever a Type is needed
  - will be replaced with actual Type

Writing your own generic class

```
public class Stack<E>
{

        public void push(E element)
        {
                contents.add(element);
        }

        public E pop()
        {
                int top = contents.size()-1;
                E result = contents.get(top);
                contents.remove(top);
                return result;
        }

        private ArrayList<E> contents = new ArrayList<E>();
}
```

Using your genetic class

```
Stack<Student> students = new
Stack<Student>();
```