The partial contents of some classes/interfaces from the j2sdk and junit:

```
public class StringBuffer {
  public void append(String s);
  public String toString();
}
public interface Iterator {
  public boolean hasNext();
  public Object next();
  public void remove();
}
public interface List {
  public int size();
  public boolean isEmpty();
  public Iterator iterator();
  public Object get(int index);
  public Object set(int index, Object element);
  public Object remove(int index);
  public void add(int index, Object element);
  public void add(Object element); // add to the end
}
public class Assert {
  static public void assertTrue(boolean condition);
  static public void assertFalse(boolean condition);
  static public void fail();
  static public void assertEquals(Object expected, Object actual);
  static public void assertEquals(int expected, int actual);
  static public void assertNull(Object object);
  static public void assertNotNull(Object object);
  static public void assertSame(Object expected, Object actual);
  static public void assertNotSame(Object expected, Object actual);
}
```

Patterns Discussed In Class:

- Static Class: A set of global variables and functions.
- MVC lite: Separate IO from business logic.
- Static Factory: a static class whose purpose is to generate instances of an interface (or interfaces). The interface and factory are public, but the implementing classes are not.
- Builder: A mutable object used to build an immutable object. Add pieces to the builder gradually, then convert to its immutable representation (eg, StringBuffer).
- Command: Encapsulate a function as an object. The command can then be accessed from many places. It may also support undo/redo.
- Strategy: Encapsulate variation into a separate object. A method accesses the variation (strategy) by delegating to a field or parameter.
- State: Create the illusion that an object changes its actual type (using delegation to a field). A nice way to eliminate case statments.
- Composite: Make collections of objects have the same interface as single objects.