8. Vector Class JDK 1.1

Similar to ArrayList; it is used to store objects.
Depreciated by ArrayList in JDK 1.2

9. Composition Construction

```java
public class MyStack
{
    private java.util.ArrayList list = new java.util.ArrayList();

    public boolean isEmpty()
    {
        return list.isEmpty();
    }

    public int getSize()
    {
        return list.size();
    }

    public Object peek()
    {
        return list.get(getSize() - 1);
    }

    public Object pop()
    {
        Object o = list.get(getSize() - 1);
        list.remove(getSize() - 1);
        return o;
    }

    public Object push(Object o)
    {
        list.add(o);
        return o;
    }

    public int Search(Object o)
    {
        return list.lastIndexOf(o);
    }

    public String toString()
    {
        return "Stack: " + list.toString();
    }
}
```

<table>
<thead>
<tr>
<th>MyStack</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ isEmpty(): boolean</td>
</tr>
<tr>
<td>+ getSize(): int</td>
</tr>
<tr>
<td>+ peek(): Object</td>
</tr>
<tr>
<td>+ pop(): Object</td>
</tr>
<tr>
<td>+ push(o: Object): Object</td>
</tr>
<tr>
<td>+ search(o: Object): int</td>
</tr>
</tbody>
</table>

Implementing MyStack as a composition model rather than as an inheritance model enables the declaration of a completely new class without inheriting unnecessary and inappropriate methods from ArrayList.
10. **protected** Data & Methods

a. A protected data item or protected method in a public class can be accessed by any class in the same package or by its subclasses even if the subclasses are in different packages.

b. Visibility / Accessibility Modifiers

<table>
<thead>
<tr>
<th>Modifiers on members in a class</th>
<th>Accessed from the same class</th>
<th>Accessed from the same package</th>
<th>Accessed from a subclass</th>
<th>Accessed from a different package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>Protected</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
<td>☑</td>
</tr>
<tr>
<td>None (default)</td>
<td>☑</td>
<td>☑</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Private</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Private Modifier**
- Hide members so that they **cannot be accessed outside of the class** i.e., the members are not intended for use outside of the class
- **Used only for members of the class**

**No Modifier**
- Allow members of the class to be **accessed directly from any class within the same package** but not from other packages
- Can be **used on the class as well as the members of the class**

**Protected Modifier**
- Enable members to be **accessed by the subclasses in any package or classes in the same package**, i.e., members of the class are **intended for extenders of the class** but not for users of the class
- **Used only for members of the class**

**Public Modifier**
- Enable members of the class to be **accessed by any class**, i.e., members of the class are **intended for users of the class**
- Can be **used on the class as well as the members of the class**

A subclass may **override a method** from a superclass and **increase its visibility** in the subclass; but it may not restrict the methods visibility, e.g., if a method is defined to be public in the superclass, it cannot be changed to protected, none (default) nor private in the subclass!
11. Preventing Extending & Overriding

```java
public final class C {
    ...
}

public class Test {
    public final void m() {
        ...
    }
}
```

A final local variable is a constant within the method block, e.g.,
```java
    int m(int n) {
        final int x = 7;
        ...
    }
```