Section 8 Handout

Sharing
- **Singleton**: Only one object exists at runtime
  - Use private constructor to prevent others from creating more objects
  - Implementation:
    1. Eager Initialization: creates the instance when the class is loaded to guarantee availability
    2. Lazy Initialization: only creates the instance once it’s needed to avoid unnecessary creation

- **Interning**: Only one object with a particular abstract value exists at runtime
  - Saves memory by compacting multiple copies

Factories: Enable program to return a subtype of the class they belong to
- **Factory Method**
  - A method that creates and returns objects
  - Method defines object interface, subclass instantiate
- **Factory Object**
  - Abstract superclass defines what can be customized
  - Concrete subclass completes and returns appropriate subclass

Builder
- The class uses the Builder from inner class instead of the constructor
- Useful when you have many constructor parameters and want to allow optional parameters

Structural Patterns
- **Adapter**: Turn different types of input and interface to the same functionality.
- **Decorator**: Adds (sometimes remove) existing functionalities
- **Proxy**: Providing the same functionality while adding additional features on accessing.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Functionality</th>
<th>Interface</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adapter</td>
<td>same</td>
<td>different</td>
<td>modify the interface</td>
</tr>
<tr>
<td>Decorator</td>
<td>different</td>
<td>same</td>
<td>extend behavior</td>
</tr>
<tr>
<td>Proxy</td>
<td><strong>same</strong></td>
<td>same</td>
<td>restrict access</td>
</tr>
</tbody>
</table>

*Functionality is same from user perspective but different inside*