

See [Shallow Copy at Wikipedia](#)
CIS 181

Composition

```
graph LR; obj1 -- contains --> containedObj1; obj1 -- contains --> containedObj2;
```

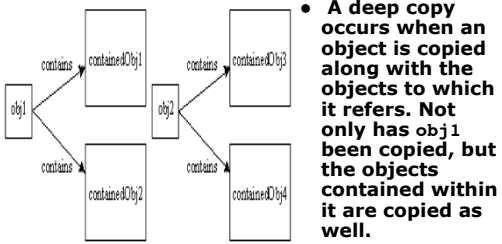
- Here we see the notion that an object is composed of other objects. Remember, variables that refer to objects are reference variables, they hold the address of the object.

clone()

```
graph LR; obj1 -- contains --> containedObj1; obj1 -- contains --> containedObj2; clonedObj1 -- contains --> containedObj1; clonedObj1 -- contains --> containedObj2;
```

- If a shallow copy is performed on obj 1, then it is copied but its contained objects are not.

Deep Copy



- The default `clone()` method, inherited from `Object`, makes a shallow copy of the object. To achieve a deep copy, extra logic must be added that explicitly calls all contained objects' `clone()` methods, which in turn call their contained objects' `clone()` methods, and so on. Getting this correct can be difficult and time consuming, and is rarely fun.
