9.5 Constructors in Subclasses

As we explained, instantiating a subclass object begins a chain of constructor calls in which the subclass constructor, before performing its own tasks, explicitly uses `super` to call one of the constructors in its direct superclass or implicitly calls the superclass's default or no-argument constructor. Similarly, if the superclass is derived from another class—true of every class except `Object`—the superclass constructor invokes the constructor of the next class up the hierarchy, and so on. The last constructor called in the chain is *always* `Object`'s constructor. The original subclass constructor's body finishes executing *last*. Each superclass's constructor manipulates the superclass instance variables that the subclass object inherits. For example, consider again the `CommissionEmployee`–`BasePlusCommission-Employee` hierarchy from Figs. 9.10–9.11. When an app creates a `BasePlusCommission-Employee` object, its constructor is called. That constructor calls `CommissionEmployee`'s constructor, which in turn calls `Object`'s constructor. Class `Object`'s constructor has an *empty body*, so it immediately returns control to `CommissionEmployee`'s constructor, which then initializes the `CommissionEmployee` instance variables that are part of the `BasePlusCommissionEmployee` object. When

`CommissionEmployee`'s constructor completes execution, it returns control to `BasePlusCommissionEmployee`'s constructor, which initializes the `baseSalary`.

Software Engineering Observation 9.9

Java ensures that even if a constructor does not assign a value to an instance variable, the variable is still initialized to its default value (e.g., `0` *for primitive numeric types,* `false` *for* `boolean`*s,* `null` *for references).*