**Second usage of “super”**

The second form of **super** acts somewhat like **this**, except that it always refers to the

superclass of the subclass in which it is used. This usage has the following general form:

super.*member*

Here, *member* can be either a method or an instance variable.

This second form of **super** is most applicable to situations in which member names

of a subclass hide members by the same name in the superclass. Consider this simple class

hierarchy:

// Using super to overcome name hiding.

**class** A {

**int** i;

}

// Create a subclass by extending class A.

**class** B **extends** A {

**int** i; // this i hides the i in A

B(**int** a, **int** b) {

**super**.i = a; // i in A

i = b; // i in B

}

**void** show() {

System.*out*.println("i in superclass: " + **super**.i);

System.*out*.println("i in subclass: " + i);

}

}

**class** UseSuper {

**public** **static** **void** main(String args[]) {

B subOb = **new** B(1, 2);

subOb.show();

}

}

**Result:**

i in superclass: 1

i in subclass: 2

Although the instance variable **i** in **B** hides the **i** in **A**, **super** allows access to the **i**

defined in the superclass. As you will see, **super** can also be used to call methods that are

hidden by a subclass.