Jason Green
ASURITE: jgreen44
jgreen44@asu.edu
Homework 2

3.2a. Not an illegal statement, but the two references to the objects are not equal to each other.
b. Not an illegal statement, but the two references to the objects are not equal to each other.
c. Legal
d. Illegal

3.6
An overloaded method is a method with the same name, but different arguments with different types passed to it. This can be in Parent/Child relationship, or you can overload methods within the same class.

```
public class Sandwich {

    // method with no arguments
    private void aMethod(){}
}
```

```
public class Sub extends Sandwich {

    // overloaded method with int argument
    private void aMethod(int a){}

    // overloaded method with double and string arguments
    private void aMethod(double a, String b){}
}
```

3.7

All public and protected methods are inherited into the subclass. If the subclass has a need for a different output or calculation, it can override the super class method. This is done by defining the method again in the subclass.

```
public class Sandwich {

    // method with no arguments
    public void aMethod(){
        System.out.println("This is the Super Class!");
    }
}
```

```
public class Sub extends Sandwich {
    @Override
    // overriden method in subclass
    public void aMethod(){
        System.out.println("This is the Sub Class!");
    }
}
```

3.8

As stated above, overriding a method is to define the same inherited method from the super class while having different code in the code block. If you accidentally pass different arguments, you have accidentally overloaded the method instead of overriding it. To avoid making this error, placing an **@Override** above the method is used. See previous example.

3.10

Yes, of course you can. It does not matter if the method that is being overloaded is in a Parent class or the same class. Example:

```
public class Main {

    public static void main(String[] args) {

        aMethod();
        aMethod(2);
        aMethod(3, "th method!");

    }

    public static void aMethod(){
        System.out.println("First method");

    }

    public static void aMethod(int a){
        System.out.println("This is the " + a + "nd method!");
    }

    public static void aMethod(int a, String b){
        System.out.println("This is the " + a + b);
    }
}
```

```
First method
This is the 2nd method!
This is the 3th method!

Process finished with exit code 0
```

3.15

No. Parent classes do no override methods from the child class. The child class inherits methods from the parent class. If the method is changed in terms of what is presented in the code block, then the child class has overridden the parent classes method, not the other way around. The parent class sets the blueprint, the child inherits.

6.2

A local class is a class that is defined within the boundaries of a code block. It can only access **constant** variables from the public, outer class. It also only visible within the code block and not outside of it and cannot be used outside of the code block.

A local inner class is a class that is called inside a method as an argument and you must instantiate this new class:

Ex: **public String aMethod(new SomeClass(arg1, arg2))**

Both are utilized when code for each class is small and creating a new class outside of the scope of the public class is not needed.