# CSC 143

Overriding methods from the Object class: equals, toString

# **Object class**

- The ultimate superclass
- Any object is an Object MyClass c = new MyClass(); System.out.print(c instanceof Object); // always prints true
- All methods from Object are inherited by any other class

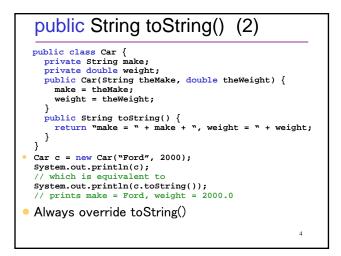
2

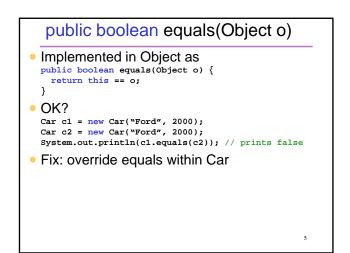
- Should they be overridden?
- Take a look at toString and equals.

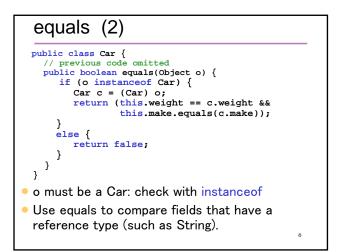
# public String toString() (1) public class Car { private String make; private double weight; public Car(String theMake, double theWeight) { make = theMake; weight = theWeight; } } Car c = new Car("Ford", 2000); System.out.println(c.toString()); // prints Car@82ba41 // name of the class + some hash code

 Override toString in Car to make it more meaningful

3





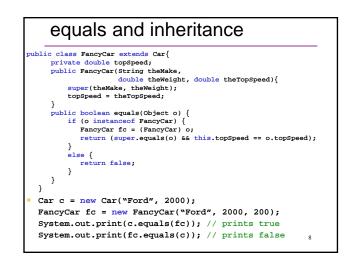


### Does it work?

Car c1 = new Car("Ford", 2000); Car c2 = new Car("Ford", 2000); System.out.println(c1.equals(c2)); // prints true.

7

- But wait!
- What if Car is inherited?



# What is going on?

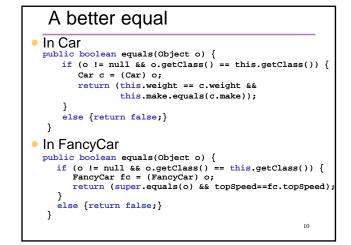
•A FancyCar is a Car fc instanceof Car is true

- •A Car is not a FancyCar
- c instanceof FancyCar is false
- •One requirement of equals is that if x.equals(y) is true, then y.equals(x) is also true.

•instanceof checks an is\_a relationship

•A necessary condition for two variables to be equal is that they have the same dynamic type.

•Get the dynamic type with getClass(). Don't use instanceof.



## One last word

- Check the class website for the complete code of the previous examples
- If equals is overridden, override hashcode as well. See later when talking about hash maps...

9