# CS 1110
# Final Exam: Review Session 2

Constructors in Subclasses
Apparent and Real Types
Casting

Biggest issue!!! You can't do questions on this topic correctly unless you draw variables, draw objects when they are created, and draw frames for method calls.

Learning to do this will help you to do the same thing when debugging.

## Motivating Problem

```
/** An instance is a bird */
public class Bird extends Creature {
   /** set of Birds in the zoo. */
   public static Bird[] aviary;

   /** Constructor: a Bird with name n */
   public Bird(String n) {
    super(n);
   }

   /** = "a Bird can (usually) fly" */
   public boolean canFly() {
      return true;
   }
}

/** An instance is a penguin */
public class Penguin extends Bird{

   /** Constructor: a new Penguin with name n*/
   public Penguin(String n) {
      super(n);
   }

   /** = "a Penguin can usually fly" */
   public boolean canFly() {
      return false;
   }
}
```

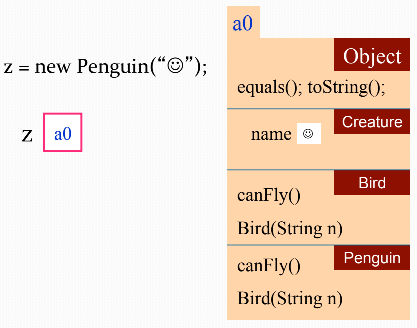
```
/** An instance is a zoo creature. */
public class Creature {
   /** This Creature's name */
   private String name;

   /** set of Creatures in the zoo*/
   public static Creature[] zoo;

   /** Constructor: a new Creature with name n */
   public Creature(String n) {
      setName(n);
   }

   /** = this Creature's name */
   public String getName() {
      return name;
   }

   /** Set this Creature's name to n */
   public void setName(String n) {
      name= n;
   }
}
```

## Key Points for this Subject

1. Subclass: inherits ALL components (fields and methods) from super class.
2. Even private fields are inherited; they appear in each object.
3. A subclass can OVERRIDE an inherited method.
4. DON'T override fields. It is called "shadowing the variables". We have never seen a good use of it. Don't do it.
5. Point 2 allows use of the bottom-up rule for finding the declaration for a reference to a file or method: start at bottom of object and search up until it is found.

## Example

- Penguin z = new Penguin("☺");

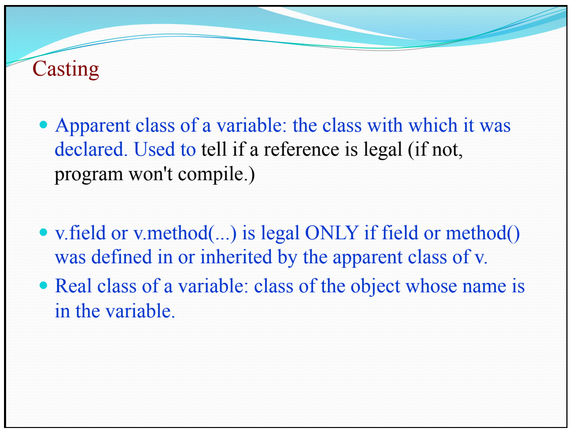


## Very Important!

- Principal: initialize fields in a superclass partition before fields in the subclass partition. There are several reasons for this. Just remember it --and follow it whenever you write a constructor.
- Therefore, EVERY constructor starts with a constructor call.
- If there is no explicit constructor call in a constructor, Java inserts super(); .

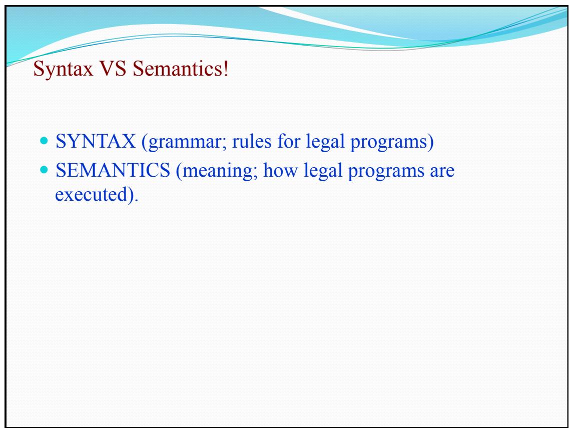
## Very Important! (contd.)

The first statement in a constructor can be one of:
- this(...);   //call another constructor in this class
- super(...); // call a constructor in the superclass



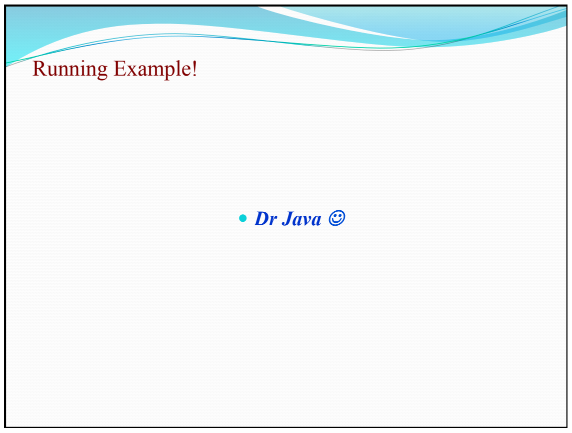
## Casting

- Apparent class of a variable: the class with which it was declared. Used to tell if a reference is legal (if not, program won't compile.)
- v.field or v.method(...) is legal ONLY if field or method() was defined in or inherited by the apparent class of v.
- Real class of a variable: class of the object whose name is in the variable.



## Syntax VS Semantics!

- SYNTAX (grammar; rules for legal programs)
- SEMANTICS (meaning; how legal programs are executed).



## Running Example!

- ***Dr Java*** ☺