CS1110 9 October 2008 Casting About

1. Casting between classes
2. Apparent and real classes.
3. Operator **instanceof**
4. The class hierarchy
5. function equals

Study Secs 4.2 and 4.3 in text

After today, you have learned ALL the basics of classes, and done extremely well. Be proud of yourselves.

Procrastination

Leave nothing for to-morrow that can be done to-day. Lincoln

How does a project get a year behind schedule? One day at a time. Fred Brooks

I don't wait for moods. You accomplish nothing if you do that. Your mind must know it has got to get down to work. Pearl S. Buck

When I start a new project, I procrastinate immediately so that I have more time to catch up. Gries

Buy a poster with the procrastinator's creed here:
www.procrastinationhelp.com/humor/procrastinators-creed

Vector<Animal> v

0	1	2
a0	null	a1

QUESTION: Which method is called by v.get(0).toString() ?

A. the one in the hidden partition for Object of a0

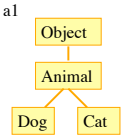
B. the one in partition Animal of a0

C. the one in partition Cat of a0

D. the one in partition Dog of a1

E. None of these

the class hierarchy:



a0
age 5 | Animal
Animal(String, int)
isOlder(Animal)
Cat(String, int) | Cat
getNoise()
toString()
getWeight()

a1
age 6 | Animal
Animal(String, int)
isOlder(Animal)
Dog(String, int) | Dog
getNoise()
toString()

2

Vector<Animal> v

0	1	2
a0	null	a1

QUESTION: Should a call
v.get(k).getWeight()
be allowed (should the program compile)?

A. Yes, because v[0] has that method.

B. No, because v[2] doesn't have that method.

C. No, because that method isn't available in Animal.

D. None of these

a0
age 5 | Animal
Animal(String, int)
isOlder(Animal)
Cat(String, int) | Cat
getNoise()
toString()
getWeight()

a1
age 6 | Animal
Animal(String, int)
isOlder(Animal)
Dog(String, int) | Dog
getNoise()
toString()

3

Vector<Animal> v

0	1	2
a0	null	a1

Apparently, v[k] is an Animal!

QUESTION: Should a call
v.get(k).getWeight()
be allowed (should the program compile)?

A. Yes, because v[0] has that method.

B. No because v[2] doesn't have that method.

C. No, because that method isn't available in Animal.

a0
age 5 | Animal
Animal(String, int)
isOlder(Animal)

a1
age 6 | Animal
Animal(String, int)
isOlder(Animal)

4

Vector<Animal> v

0	1	2
a0	null	a1

Apparently, v[k] is an Animal!

The call
v.get(k).getWeight()
is illegal, and the program won't compile, because: The apparent type of v[k], which is Animal, does not declare or inherit a method getWeight.

a0
age 5 | Animal
Animal(String, int)
isOlder(Animal)

a1
age 6 | Animal
Animal(String, int)
isOlder(Animal)

5

Casting up the class hierarchy

You know about casts like

(**int**) (5.0 / 7.5)

(**double**) 6

double d= 5; // automatic cast

We now discuss casts up and down the class hierarchy.

Animal h= new Cat("N", 5);

Cat c= (Cat) h;

a0
age 5 | Animal
Animal(String, int)
isOlder(Animal)
Cat(String, int) | Cat
getNoise()
toString()
getWeight()

a1
age 6 | Animal
Animal(String, int)
isOlder(Animal)
Dog(String, int) | Dog
getNoise()
toString()

6

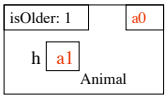
Implicit casting up the class hierarchy

```
public class Animal {

   /** = "this is older than h" */
   public boolean isOlder(Animal h)
     { return this.age > h.age; }
}
```

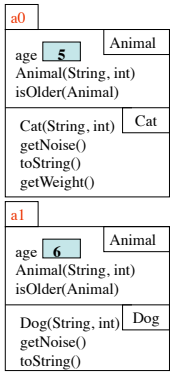
Casts up the hierarchy done automatically

```
c= new Cat("C", 5);
d= new Dog("D", 6);
c.isOlder(d)    ?????
```



Upward automatic casts make sense. Here, any Dog is an Animal

a1 is cast from Dog to Animal, automatically

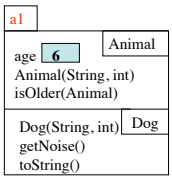


7

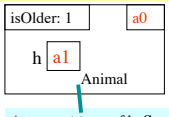
Implicit casting up the class hierarchy

```
public class Animal {
   /** = "this is older than h" */
   public boolean isOlder(Animal h)
     { return this.age > h.age; }
}
```

Two new terms to learn!




```
c= new Cat("C", 5);
d= new Dog("D", 6);
c.isOlder(d)   --what is its value?
```



Apparent type of h. **Syntactic property**. The type with which h is defined.

Real type of h: Doc (type of object a1).

Semantic property. The class-type of the folder whose name is currently in h.

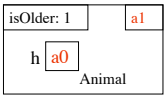
Apparently, h is an Animal, but **really**, it's a Dog.

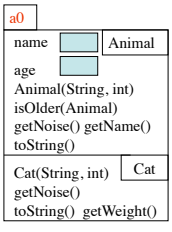
8

What components can h reference?

```
public class Animal {
   /** = "this is older than h" */
   public boolean isOlder(Animal h)
     { return this.age > h.age; }
}
```

```
c= new Cat("C", 5);
d= new Dog("D", 6);
d.isOlder(c)
```





What can isOlder reference in object h?

Determined by the apparent type: Only components in partition Animal (and above)!!!

h.getWeight() is illegal. Syntax error.

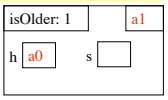
Apparent type of h: Animal
Real type of h: Cat

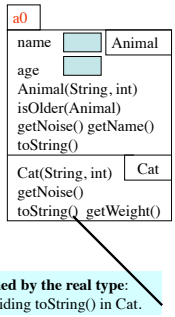
9

What method is called by h.toString() ?

```
public class Animal {
    public boolean isOlder(Animal h) {
       String s= h.toString();
       return this.age > h.age;
} }
```

```
c= new Cat("C", 5);
d= new Dog("D", 6);
d.isOlder(c)
```





Determined by the real type: The overriding toString() in Cat.

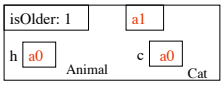
Apparent type of h: Animal
Real type of h: Cat

What method is called by h.toString() ?

10

Explicit cast down the hierarchy

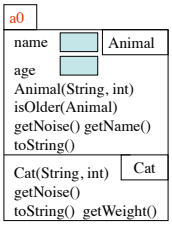
```
public class Animal {
    // If Animal is a cat, return its weight;
       otherwise, return 0.
    public int checkWeight(Animal h) {
      if ( ! (h instanceof Cat) )
          return 0;
      // h is a Cat
      int c= (Cat) h ;      // downward cast
      return c.getWeight();
}
```



Here, **(Dog) h** would lead to a runtime error.

Don't try to cast an object to something that it is not!

Apparent type of h: Animal
Real type of h: Cat

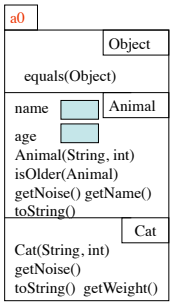


11

The correct way to write method equals

```
public class Animal {
    /** = "h is an Animal with the same
          values in its fields as this Animal */
    public boolean equals (Object h) {

      if (!(h instanceof Animal)) return false;
      Animal ob= (Animal) h;
      return name.equals(ob.name)   &&
             age == ob.age;
}
```



12