CS1110 13 February 2012 Casting About

Top finalists from a real-life "Dilbert quotes contest"

As of tomorrow, employees will be able to access the building only using individual security cards. Pictures will be taken next Wednesday and employees will receive their cards in two weeks." (Fred Dales, Microsoft)

I need an exact list of specific unknown problems we might encounter. (Lykes Lines Shipping)

Email is not to be used to pass on information or data. It should be used only for company business. (Accounting manager, Electric Boat Company)

This project is so important, we can't let things that are more important interfere with it. (Advertising/Marketing manager, United Parcel Service)

Doing it right is no excuse for not meeting the schedule. (Plant manager, Delco Corporation)

CS1110 13 February 2012 Casting About

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

Study Secs 4.2 and 4.3 in text

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

	0	1	2
Vector<Animal> v	a0	null	a1

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

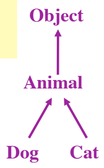
A. the one in the hidden partition for Object of a0

B. the one in partition Animal of a0

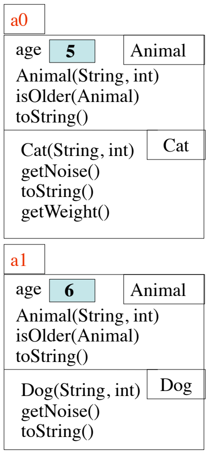
C. the one in partition Cat of a0

D. the one in partition Dog of a1

E. None of these


Object
Animal
Dog
Cat

the class hierarchy:
(→ means "extends" or "is a kind of")


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

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

	0	1	2
Vector<Animal> v	a0	null	a1

QUESTION: Should a call

v.get(k).getWeight()

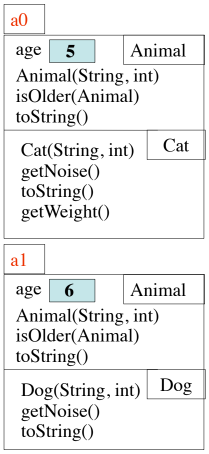
be allowed (should the program compile)?

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)
toString()
Cat(String, int) Cat
getNoise()
toString()
getWeight()

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

	0	1	2
Vector<Animal> v	a0	null	a1

Apparently, v[k] is an Animal!

QUESTION: Should a call

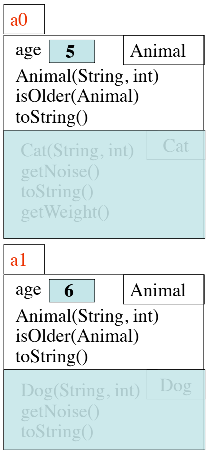
v.get(k).getWeight()

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)
toString()
Cat(String, int) Cat
getNoise()
toString()
getWeight()

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

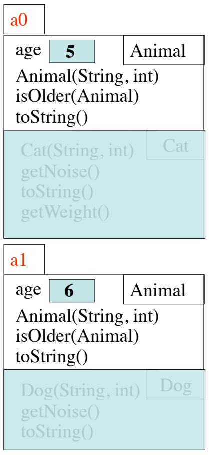
	0	1	2
Vector<Animal> v	a0	null	a1

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)
toString()
Cat(String, int) Cat
getNoise()
toString()
getWeight()

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

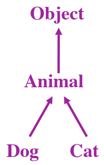
Casting up the class hierarchy

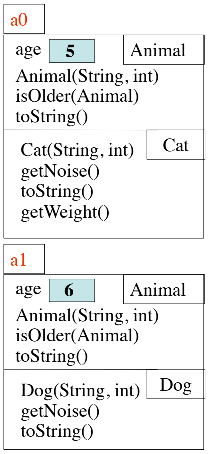
You know about casts like

(int) (5.0 / 7.5)

(double) 6

double d= 5;   // automatic cast





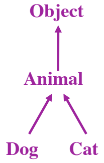
We now discuss casts up and down the class hierarchy.

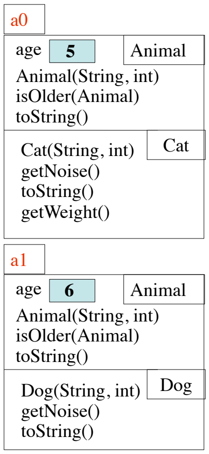
Animal h= new Cat("N", 5);

Cat c= (Cat) h;

Implicit casting up the class hierarchy

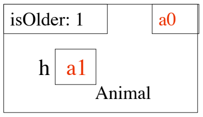
```
public class Animal {
    /** = "this is older than h" */
    public boolean isOlder(Animal h)
      { return this.age > h.age; }
}
```





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

Casts up the hierarchy done automatically



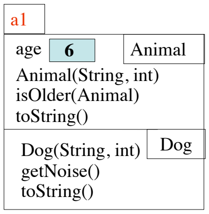
Upward automatic casts make sense. Here, any Dog is an Animal.

a1 is cast from Dog to Animal, automatically

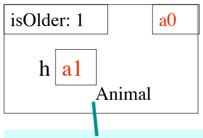
Implicit casting up the class hierarchy

```
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?



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

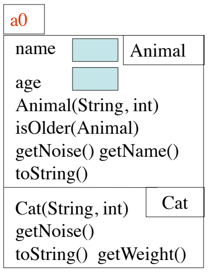
Semantic property. The class-type of the folder whose name is currently in h.

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

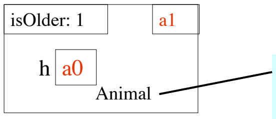
Apparently, h is an Animal, but **really**, it's a Dog.

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?

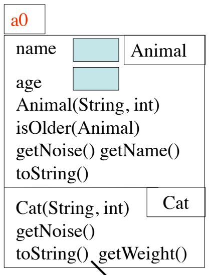
Determined by the apparent type: Only components in partition Animal (and above)!!!

h.getWeight() is illegal. Syntax error.

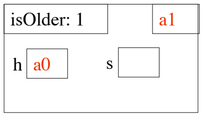
Apparent type of h: Animal
Real type of h: Cat

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)



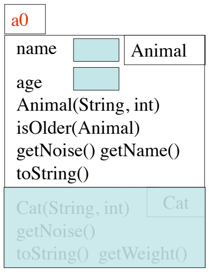
What method is called by h.toString() ?

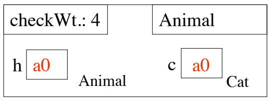
Determined by the real type: The overriding toString() in Cat.

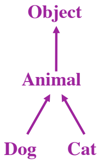
Apparent type of h: Animal
Real type of h: Cat

Explicit cast down the hierarchy

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







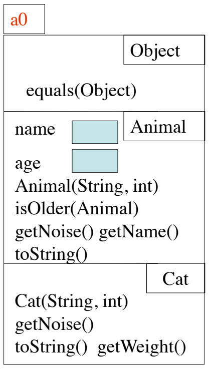
Here, **(Dog) h** would lead to a runtime error.

You can't cast an object to something that it is not!

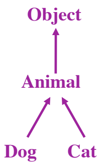
Apparent type of h: Animal
Real type of h: Cat

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;
}
```



Of course, you may want to define equals() in Cat and Dog as well, since a cat is probably not equal to a dog, even if they have the same name and age!



Applications

```
public class C {
    …
    public static void main(String[] args) {
        …
    }
    …
}
```

Application: java program with a class that contains a static procedure main with one argument: array of strings.

(The parameter can be used to give input values to the application. We don’t talk about that today; we always use **null**).

Run an application in DrJava:

1. Select the class with method main and hit button **run** (top navigation bar)

or

2. Type C.main(null); in the interactions pane.

jar file (Java Archive file)

(like tar file (Tape Archive file))

Contains (among other things)

(1) .class files

(2) a “manifest”, which says which class has method main

Manifest:
A list of passengers or an invoice of cargo for a vehicle (as a ship or plane).

Like today’s .zip file, except that it *must* contains certain things

Two ways to execute an application that is in a jar file without using DrJava

(1) Double click its icon in a directory.
(2) Type

```
java   -jar   images.jar
```

in a terminal window (or DOS, or command-line window)

Creating a jar file

(1) Look up “jar file” in the text. It will explain how to create jar file using a command-line prompt window (or DOS window).

(2) Use a “project” in DrJava.

Most Interactive Development Environments (IDEs) like DrJava and Ecclipse, have *projects*.

Project: A bunch of java files that make up a program together with tools for maintaining/manipulating it.

We show you how to create a project in DrJava and then how to create a jar file for it.

Projects in DrJava

We show only enough of projects to create a jar file

1. Load files for a program into DrJava.

2. Use menu item **Project -> New**, saving the file in the same directory. Give it any name you want, but don’t put an extension on it. No need to put anything in the Project Properties panel. Just click OK.

If you wrote a2 for the name, it creates a file a2.drjava

Note how the left pane, which contains the names of files being edits, has changed.

3. Click button **Compiler Project** (in the navigation bar)

4. Select menu item **Project -> Create jar file from project**. In the pane that opens,

1. select “Jar classes” (or “Jar All files” if you want),
2. click “Make executable”,
3. type in main class (no extension)
4. type in the name you want (e.g. a2.jar) --note where it will be stored! Navigate to somewhere else if you want it elsewhere.
5. click OK.