CS 61B Spring 2023

Inheritance

Discussion 4: February 6, 2023

1 It's a Bird! It's a Plane! It's a CatBus!

On a research expedition studying air traffic, we discovered a new species: the Flying Interfacing CatBus, which acts like a vehicle and has the ability to honk (safety is important!).

(a) Given the `Vehicle` and `Honker` interfaces, fill out the `CatBus` class so that `CatBus`es can rev their engines and honk at other `CatBus`es.

```
interface Vehicle {
    void revEngine();
}

interface Honker {
    void honk();
}

public class CatBus ______________________ _________________, _________________ {
    @Override
    ____________ _____________ ____________________ {
        // CatBus revs its engine, implementation not shown
    }

    @Override
    ____________ _____________ ____________________{
        // CatBus honks, implementation not shown
    }

    /** Allows CatBus to honk at other CatBuses. */
    public void conversation(CatBus target) {
        honk();
        target.honk();
    }
}
```

(b) It's a lovely morning in the skies and we've encountered a horrible `Goose`, which also `implements Honker` (it has a knife in its beak!). Modify the `conversation` method signature so that `CatBus`es can `honk` at both `CatBus`es *and* `Goose`s while only having one argument, `target`.

(c) Assume that `CatBus` and `Goose` both use the default constructor. Which of the following lines compile?

```
Honker cb = new CatBus();
Honker h = new Honker();
CatBus g = new Goose();
```

2 Raining Cats and Dogs

(a) What would Java do after executing the main method in the `TestAnimal` class? Fill in the table provided with the method saved at compile time, the method called at runtime, and overall output for lines 8-30 if applicable. If there is an error, write whether it is a runtime error or compile time error, and then proceed through the rest of the code as if the erroneous line were not there.

```
public class Animal {
    public String name, noise;

    public Animal(String name) {
        this.name = name;
        this.noise = "Huh?";
    }

    public void greet(Animal a) { System.out.println("Hi " + a.name + ", I'm " + name); }
    public void play() { System.out.println("I love to play! " + noise); }
    public static void sleep() { System.out.println("Naptime!"); }
}

public class Cat extends Animal {
    public Cat(String name) {
        super(name);
        this.noise = "Meow!";
    }

    public void greet(Animal a) { System.out.println("Cat " + name + " says: " + noise); }
    public void play(String noise) {
        System.out.println("Woo it is so much fun being a cat!" + noise);
    }
}
public class Dog extends Animal {
    public Dog(String name) {
        super(name);
        noise = "Woof!";
    }

    public void greet(Animal a) { System.out.println("Dog " + name + " says: " + noise); }
    public void play(int happiness) {
        if (happiness > 10) {
            System.out.println("Woo it is so much fun being a dog!");
        }
    }
    public static void sleep() { System.out.println("I love napping!"); }
}
```

```
1  public class TestAnimal {
2      public static void main(String[] args) {
3          Animal a = new Dog("Pluto");
4          Animal b = new Animal("Bear");
5          Cat c = new Cat("Garfield");
6          Dog d = new Dog("Lucky");
7
8          Cat e = new Animal("Kitty");
9
10         a.greet(c);
11
12         a.sleep();
13
14         c.play(":)");
15
16         c.greet(d);
17
18         ((Animal) c).greet(d);
19
20         d.sleep();
21
22         a = c;
23
24         a.play(14);
25
26         ((Cat) b).play();
27
28         d = (Dog) a;
29
30         c = a;
31     }
32 }
```

Line	Compile time (static)	Runtime (dynamic)	Output
8			
10			
12			
14			
16			
18			
20			
22			
24			
26			
28			
30			

(b) Spoiler alert! There is an error on the last line, line 30. How could we fix this error?