

CS 61B Spring 2021

Inheritance

Discussion 4: February 8, 2021

1 Raining Cats and Dogs

- (a) Below, four classes are defined. What would Java do after executing the main method in the TestAnimal class? Next to each blank, if something is printed write it down. 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. Lines in blue cover casting.

```

1 public class Animal {
2     public String name, noise;
3     public int age;
4
5     public Animal(String name, int age) {
6         this.name = name;
7         this.age = age;
8         this.noise = "Huh?";
9     }
10
11     public void greet() {System.out.println("Animal " + name + " says: " + this.noise);}
12
13     public void play() {System.out.println("Woo it is so much fun being an animal!")}
14 }
15
16 public class Cat extends Animal {
17     public Cat(String name, int age) {
18         super(name, age);
19         this.noise = "Meow!";
20     }
21
22     @Override
23     public void greet() {System.out.println("Cat " + name + " says: " + this.noise);}
24
25     public void play(String expr) {System.out.println("Woo it is so much fun being a cat!" + expr)}
26 }
27
28 public class Dog extends Animal {
29     public Dog(String name, int age) {
30         super(name, age);
31         noise = "Woof!";
32     }
33
34     @Override
35     public void greet() {System.out.println("Dog " + name + " says: " + this.noise);}

```

Handwritten notes and diagrams:

- Animal class: name, age, noise; methods: greet(), play().
- Cat class: noise, greet(), play(String), play().
- Dog class: noise, greet(), play(int), play().
- Arrows indicate inheritance from Animal to Cat and Dog.
- A dashed box highlights the play() method in the Animal class.

2 Inheritance

```

36
37 public void play(int happiness) {
38     if (happiness > 10) {
39         System.out.println("Woo it is so much fun being a dog!");
40     }
41 }
42 }
43
44 public class TestAnimal {
45     public static void main(String[] args) {
46         Animal a = new Animal("Pluto", 10);
47         Cat c = new Cat("Garfield", 6);
48         Dog d = new Dog("Fido", 4);
49         a.greet(); // Animal Pluto says: Huh?
50         c.greet(); // Cat Garfield says: Meow?
51         d.greet(); // Dog Fido says: Woof!
52         c.play(); // Woo it is so much fun being an animal!
53         c.play(":") // Woo it is so much fun being a cat! :D
54         a = c;
55         ((Cat) a).greet(); //
56         ((Cat) a).play(":D"); // Woo it is so much fun being a cat! :D
57         a.play(14); // Compile time error
58         ((Dog) a).play(12); // Run-time error
59         a.greet(); //
60         c = a; // Compile time error
61     }
62 }

```

(b) Spoiler alert! There is an error on the last line, line 60. How could we fix this error?

Using cast:
`c = (Cat) a`

2 An Exercise in Inheritance Misery

Cross out any lines that result in compiler errors, as well as subsequent lines that would fail because of the compiler error. Put an X through runtime errors (if any). Don't just limit your search to main, there could be errors in classes A,B,C. What does D.main output after removing these lines?

```

1 public class A {
2     public int x = 5;
3     public void m1() {      System.out.println("Am1-> " + x);      }
4     public void m2() {      System.out.println("Am2-> " + this.x);  }
5     public void update() {  x = 99;                                }
6 }
7 public class B extends A {
8     public void m2() {      System.out.println("Bm2-> " + x);      }
9     public void m2(int y) { System.out.println("Bm2y-> " + y);      }
10    public void m3() {      System.out.println("Bm3-> " + "called");  }
11 }
12 public class C extends B {
13     public int y = x + 1;
14     public void m2() {      System.out.println("Cm2-> " + super.x);  }
15     //public void m4() {      System.out.println("Cm4-> " + super.super.x); }
16     public void m5() {      System.out.println("Cm5-> " + y);        }
17 }
18 public class D {
19     public static void main (String[] args) {
20         // B a0 = new A(); CE x A m1() m2() update()
21         // a0.m1(); CE
22         // a0.m2(16); CE
23         A b0 = new B(); ✓
24         System.out.println(b0.x); ✓ 5 B m1()
25         b0.m1(); Am1->5 x B m1()
26         b0.m2(); Bm2->5
27         // b0.m2(61); CE
28         B b1 = new B(); ✓
29         b1.m2(61); ✓ y x C m1()
30         b1.m3(); ✓
31         A c0 = new C(); ✓ Bm2y->b1
32         c0.m2(); ✓ Bm3->called Cm2->5
33         // C c1 = (A) new C(); CE
34         A a1 = (A) c0; ✓
35         C c2 = (C) a1; ✓
36         c2.m3(); ✓ Bm3->called
37         // c2.m4();
38         c2.m5(); ✓ Cm5->6
39         ((C) c0).m3(); ✓ Bm3->called
40         // (C) c0.m2(); CE
41         b0.update(); ✓

```

overridden

↓ overridden

↓ overriding

m4() m5()

A | b0 → B | x | 5 | 99

B | b1 → B | x | 5

A | c0 → C | x | 5

A | a1 → C | y | 6

C | c2

4 Inheritance

```
42     b0.m1(); Am1 -> qq  
43 }  
44 }
```