

WES-CS GROUP MEETING #15

Jeopardy Exam Review

This Jeopardy game has *Inheritance* and *Exceptions* questions. The *Inheritance* questions will use the following class definitions.

```
1. public class ParentThing {
2.     public int j;
3.     private int p;
4.     public void methodA() { System.out.println("Parent A1"); }
5.     public void methodA(String str) { System.out.println("Parent A2"); }
6.     public void methodB() { System.out.println("Parent B"); }
7.     public static void methodC() { System.out.println("Parent C"); }
8. }

9. public class ThingOne extends ParentThing {
10.     public int onej;
11.     public void methodA(int c) { System.out.println("ThingOne A"); }
12. }

13. public class ThingTwo extends ParentThing {
14.     public void methodB() { System.out.println("ThingTwoB1"); }
15.     public void methodB(String str) { System.out.println("ThingTwoB2"); }
16. }

17. public class ChildOne extends ThingOne {
18.     public void methodA() { System.out.println("ChildOneA"); }
19. }

20. public class ChildTwo extends ThingTwo {
21.     public void methodA() { System.out.println("ChildTwoA"); }
22. }
```

The *Inheritance* questions will assume that the following statements (not in any of the above classes) have been executed.

```
ParentThing parent = new ParentThing();
ThingOne one = new ThingOne();
ThingTwo two = new ThingTwo();
ChildOne kid1 = new ChildOne();
ChildTwo kid2 = new ChildTwo();
ParentThing p2 = new ChildTwo();
```

The *Exceptions* questions will use the following code.

```
public class TopEx extends Exception {}

public class Ex1 extends TopEx {}

public class Ex2 extends TopEx {}

1. public class ExceptionExample {
2.     public static void main(String[] args) {
3.         try {
4.             foo();
5.             bar();
6.         } catch (TopEx te) {
7.             System.out.println("ERROR 1");
8.         } finally {
9.             System.out.println("DONE");
10.        }
11.    }

12.    public static void foo() throws Ex2 {
13.        System.out.println("START FOO");
14.        // PROGRAM LINE A
15.        try {
16.            // PROGRAM LINE B
17.            bar();
18.        } catch (Ex1 ex) {
19.            System.out.println("ERROR 2");
20.        }
21.    }

22.    public static void bar() throws Ex1 {
23.        // PROGRAM LINE C
24.        try {
25.            // PROGRAM LINE D
26.        } catch (Ex2 ex) {
27.            System.out.println("ERROR 3");
28.        }
29.        System.out.println("DONE BAR");
30.    }

31.    public static void fraz(int k) throws Ex1, Ex2 {
32.        if (k==0) throw new Ex1();
33.        else throw new Ex2();
34.    }
35. }
```

INHERITANCE: EXPLAIN AND FIX THE ERROR

For 100: `parent.methodB("Testing");`

Answer: No `methodB` with a `String` param in the parent class.

Fix: `parent.methodB();`
`parent.methodA("Testing");`
`two.methodB("Testing");`
`kid2.methodB("Testing");`

For 200: `two.p = 10;`

Answer: The `p` field is a private field of `two`'s parent class.

Fix: `two.j = 10;`

For 300: `kid2.onej = 20;`

Answer: There is no `onej` field in the `ChildTwo`, `ThingTwo`, or `ParentThing` class.

Fix: `kid2.j = 20;`
`kid1.onej = 20;`
`one.onej = 20;`

For 400: `ThingOne.methodB();`

Answer: There is no static `methodB` in the `ThingOne` or the `ParentThing` class.

Fix: any of the variables `.methodB()`
`ThingOne.methodC()`

For 500: `kid2 = p2;`

Answer: The type of `p2` is `ParentThing`, which cannot be assigned to variable `kid2` because its type, `ChildTwo`, is a subtype of `parentThing`.

Fix: `kid2 = (ChildTwo)p2;`

INHERITANCE: WHAT IS PRINTED WHEN THIS CODE EXECUTES?

For 100: `two.methodB()`

Answer: `ThingTwoB1`

For 200: `kid1.methodB()`

Answer: `ParentB`

For 300: `kid1.methodA(45)`

Answer: `ThingOneA`

For 400: `kid2.methodA("Hello World");`

Answer: `ParentA2`

For 500: `p2.methodB()`

Answer: `ThingTwoB1`

EXCEPTIONS: ON WHICH “PROGRAM LINES” CAN THIS CODE GO?

For 100: `throw new NullPointerException();`

Answer: PROGRAM LINES A, B, C, and D

For 200: `throw new TopEx();`

Answer: no PROGRAM LINE

For 300: `throw new Ex1();`

Answer: PROGRAM LINES B, C, D

For 400: `throw new Ex2();`

Answer: PROGRAM LINES A, B, D

For 500: `fraz(10);`

Answer: PROGRAM LINES B and D

EXCEPTIONS: WHAT IS PRINTED WHEN THIS CODE REPLACES THIS PROGRAM LINE?

For 100: None of the "PROGRAM LINES" is replaced.

Answer: START FOO
DONE BAR
DONE BAR
DONE

For 200: PROGRAM LINE B is replaced with `throw new Ex1();`

Answer: START FOO
ERROR 2
DONE BAR
DONE

For 300: PROGRAM LINE C is replaced with `throw new Ex1();`

Answer: START FOO
ERROR 2
ERROR 1
DONE

For 400: PROGRAM LINE B is replaced with `throw new Ex2();`

Answer: START FOO
ERROR 1
DONE

For 500: PROGRAM LINE D is replaced with `throw new Ex1();`

Answer: START FOO
ERROR 2
ERROR 1
DONE