

Quiz name: **Java 111 Chapter 8 (from version 1)**Date: **11/17/2015**Question with Most Correct Answers: **#14**Total Questions: **17**Question with Fewest Correct Answers: **#3**

---

1. An abstract class can only have abstract methods.

4/13

☐ A

True

7/13

☒ BFalse

---

2. When you don't want a class to be instantiated (in other words, you don't want anyone to make a new object of that class type) mark the class with the "abstract" keyword.

10/13

☒ A

True

1/13

☐ BFalse

---

3. Which of the following are true?

1/13

☐ A

An interface must be created using the keyword "abstract".

9/13

☒ B

An interface defines only abstract methods.

9/13

☒ C

A class can implement multiple interfaces.

6/13

☐ DAll interface methods are implicitly public.

---

4. All objects come out of an ArrayList&lt;Object&gt; as type Object, unless you use a cast.

11/13

☒ A

True

0/13

☐ BFalse

---

5. Multiple inheritance is allowed in Java, meaning you may extend multiple classes.

4/13

☐ A

True

7/13

☒ BFalse

---

6. If you override a superclass method in a subclass, you cannot invoke (call) the superclass method.

0/13

☐ A

True

11/13

☒ BFalse

---

7. An abstract method has no body and ends with curly braces.

5/13

☐ A

True

6/13

☒ B

False

---

8. You can extend only one class (i.e. you can have only one immediate superclass).

10/13



True

1/13



False

---

9. Given the following: `JavaRockStar rockstar = new JavaRockStar();` what is the object reference variable?

0/13



JavaRockStar

0/13



new

11/13



rockstar

0/13



none of the above

---

10. Given the following: `JavaRockStar rockstar = new JavaRockStar();` what is the object reference type?

11/13



JavaRockStar

0/13



new

0/13



rockstar

0/13



none of the above

---

11. Given the following: `SuperStarCoder rockstar = new JavaRockStar();` what is the object reference type?

1/13



JavaRockStar

1/13



rockstar

9/13



SuperStarCoder

0/13



new

0/13



none of the above

---

12. Given the following: `SuperStarCoder rockstar = new JavaRockStar();` what is the actual object type?

9/13



JavaRockStar

0/13



rockstar

2/13



SuperStarCoder

0/13



new

0/13



none of the above

---

13. Given the following, what output do you expect?

- 3/13 ☐ A Line 10 and 14 will each run twice.
- 0/13 ☐ B Line 10 will run twice, line 14 will run once.
- 5/13 ☒ C This will not compile due to line 29.
- 3/13 ☐ D This will not compile due to line 27.
- 0/13 ☐ E None of the above

```
1 public abstract class Programmer {
2     public abstract void writeProgram();
3 }
4
5 public class SuperheroClass extends Programmer {
6     public void writeProgram() {
7         System.out.println("I'm writing programs using my superpowers!");
8     }
9 }
10
11 public void writeProgram() {
12     System.out.println("I'm a doctor in the future, I know, I'm... Super, Doctor, Superhero, Doctor, Doctor, Oh, I'm...");
13 }
14
15 public class ProgrammerInterface {
16     public void writeProgram() {
17         System.out.println("I'm writing programs using my superpowers!");
18     }
19 }
20
21 public class ProgrammerInterface {
22     public void writeProgram() {
23         System.out.println("I'm writing programs using my superpowers!");
24     }
25 }
26
27 public class ProgrammerInterface {
28     public void writeProgram() {
29         System.out.println("I'm writing programs using my superpowers!");
30     }
31 }
32
33 public class ProgrammerInterface {
34     public void writeProgram() {
35         System.out.println("I'm writing programs using my superpowers!");
36     }
37 }
38
39 public class ProgrammerInterface {
40     public void writeProgram() {
41         System.out.println("I'm writing programs using my superpowers!");
42     }
43 }
44
45 public class ProgrammerInterface {
46     public void writeProgram() {
47         System.out.println("I'm writing programs using my superpowers!");
48     }
49 }
50
51 public class ProgrammerInterface {
52     public void writeProgram() {
53         System.out.println("I'm writing programs using my superpowers!");
54     }
55 }
56
57 public class ProgrammerInterface {
58     public void writeProgram() {
59         System.out.println("I'm writing programs using my superpowers!");
60     }
61 }
62
63 public class ProgrammerInterface {
64     public void writeProgram() {
65         System.out.println("I'm writing programs using my superpowers!");
66     }
67 }
68
69 public class ProgrammerInterface {
70     public void writeProgram() {
71         System.out.println("I'm writing programs using my superpowers!");
72     }
73 }
74
75 public class ProgrammerInterface {
76     public void writeProgram() {
77         System.out.println("I'm writing programs using my superpowers!");
78     }
79 }
80
81 public class ProgrammerInterface {
82     public void writeProgram() {
83         System.out.println("I'm writing programs using my superpowers!");
84     }
85 }
86
87 public class ProgrammerInterface {
88     public void writeProgram() {
89         System.out.println("I'm writing programs using my superpowers!");
90     }
91 }
92
93 public class ProgrammerInterface {
94     public void writeProgram() {
95         System.out.println("I'm writing programs using my superpowers!");
96     }
97 }
98
99 public class ProgrammerInterface {
100    public void writeProgram() {
101        System.out.println("I'm writing programs using my superpowers!");
102    }
103 }
```

14. What is the proper way to create an interface called Payable?
- 0/13 ☐ A public abstract interface class Payable {}
- 0/13 ☐ B public abstract Payable {}
- 0/13 ☐ C public abstract interface class Payable extends Payable {}
- 12/13 ☒ D public interface Payable {}
- 0/13 ☐ E interface PayMe() extends Money implements Payable()

15. A class must extend a superclass before it can implement an interface.
- 2/13 ☐ A True
- 10/13 ☒ B False

16. If a class does not pass the IS-A test, it probably should not extend anything (other than Object).
- 10/13 ☒ A True
- 1/13 ☐ B False

17. An interface is a 100% abstract class, meaning it defines only abstract methods.
- 8/13 ☒ A True
- 4/13 ☐ B False