

Java 111 Chapter 8 (from version 1)

Total Questions: 17

Most Correct Answers: #17

Least Correct Answers: #13

1. An abstract class can only have abstract methods.

1/12 ☐ A True

5/12 ☒ B False

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.

6/12 ☒ A True

0/12 ☐ B False

3. Which of the following are true?

0/12 ☐ A An interface must be created using the keyword "abstract".

6/12 ☒ B An interface defines only abstract methods.

5/12 ☒ C A class can implement multiple interfaces.

4/12 ☒ D All interface methods are implicitly public.

4. All objects come out of an ArrayList<Object> as type Object, unless you use a cast.

4/12 ☒ A True

2/12 ☐ B False

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

0/12 ☐ A True

6/12 ☒ B False

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

0/12 ☐ A True

6/12 ☒ B False

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

6/12 ☒ A True

0/12 ☐ B False

8. Write an abstract method called eatCake that accepts one parameter for the number of slices to eat and returns a String.

Anon anon0cabed97d9f947d7

✗ i like pie

Anon anon0ed12e02031542c9

✗ public abstract String eatCake();

Anon anon14fde5297e3e4918

✗

```
public eatCake(String slices) {  
    return slices;  
}
```

Anon anon36f79a088df54742

✗ Public Abstract String eatCake(int slices);

Anon anon61fa87e7b678444c

✓ public abstract String eatCake(int numberOfSlices);

Anon anon715f3118ad8e46b3

✓ public abstract String eatCake(int numberOfSlices);

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

0/12 ☐ A JavaRockStar

0/12 ☐ B new

6/12 ☒ C rockstar

0/12 ☐ D none of the above

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

6/12 ☒ A JavaRockStar

0/12 ☐ B new

0/12 ☐ C rockstar

0/12 ☐ D none of the above

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

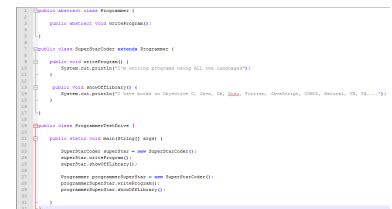
- 0/12 ☐ A JavaRockStar
- 0/12 ☐ B rockstar
- 6/12 ☒ C SuperStarCoder
- 0/12 ☐ D new
- 0/12 ☐ E none of the above

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

- 6/12 ☒ A JavaRockStar
- 0/12 ☐ B rockstar
- 0/12 ☐ C SuperStarCoder
- 0/12 ☐ D new
- 0/12 ☐ E none of the above

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

- 0/12 ☐ A Line 10 and 14 will each run twice.
- 1/12 ☐ B Line 10 will run twice, line 14 will run once.
- 2/12 ☒ C This will not compile due to line 29.
- 2/12 ☐ D This will not compile due to line 27.
- 1/12 ☐ E None of the above



```
1 public interface Program {
2     public method void testProgram();
3 }
4
5 public class SuperStarCoder extends Program {
6     public void testProgram() {
7         System.out.println("The testing program using ALL the languages");
8     }
9 }
10 public void testProgram() {
11     System.out.println("I have looked on Wikipedia to learn the Java, C#, PHP, Python, JavaScript, C++, Haskell, R#, F#, ...");
12 }
13
14 public class ProgramTestDrive {
15     public static void main(String[] args) {
16         SuperStarCoder superStar = new SuperStarCoder();
17         superStar.testProgram();
18         superStar.testProgram();
19         Program programSuperCoder = new SuperStarCoder();
20         programSuperCoder.testProgram();
21         programSuperCoder.testProgram();
22     }
23 }
24
25
26
27
28
29 Program programSuperCoder = new SuperStarCoder();
30
```

14. What is the proper way to create an interface called Payable?

- 0/12 ☐ A public abstract interface class Payable {}
- 0/12 ☐ B public abstract Payable {}
- 0/12 ☐ C public abstract interface class Payable extends Payable {}
- 6/12 ☒ D public interface Payable {}
- 0/12 ☐ E interface PayMe() extends Money implements Payable()

15. A class must extend a superclass before it can implement an interface.

- 1/12 ☐ A True
- 5/12 ☒ B False

16. If a class does not pass the IS-A test, it probably should not extend anything (other than Object).

6/12 ☒ A True

0/12 ☐ B False

17. An interface is a 100% abstract class, meaning it defines only abstract methods.

6/12 ☒ A True

0/12 ☐ B False