

1Z0-816^{Q&As}

Java SE 11 Programmer II

Pass Oracle 1Z0-816 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

<https://www.pass2lead.com/1z0-816.html>

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Oracle
Official Exam Center

- ⚙️ **Instant Download** After Purchase
- ⚙️ **100% Money Back** Guarantee
- ⚙️ **365 Days** Free Update
- ⚙️ **800,000+** Satisfied Customers



QUESTION 1

Given the declaration:

```
@interface Resource {  
    String name();  
    int priority() default 0;  
}
```

Examine this code fragment:

```
/* Loc1 */ class ProcessOrders { ... }
```

Which two annotations may be applied at Loc1 in the code fragment? (Choose two.)

- A. @Resource(priority=100)
- B. @Resource(priority=0)
- C. @Resource(name="Customer1", priority=100)
- D. @Resource(name="Customer1")
- E. @Resource

Correct Answer: AB

QUESTION 2

Given:

```
jdeps -jdkinternals C:\workspace4\SimpleSecurity\jar\classes.jar
```

Which describes the expected output?

- A. jdeps lists the module dependencies and the package names of all referenced JDK internal APIs. If any are found, the suggested replacements are output in the console.
- B. jdeps outputs an error message that the -jdkinternals option requires either the -summary or the verbose options to output to the console.
- C. The -jdkinternals option analyzes all classes in the .jar and prints all class-level dependencies.
- D. The -jdkinternals option analyzes all classes in the .jar for class-level dependencies on JDK internal APIs. If any are found, the results with suggested replacements are output in the console.

Correct Answer: A

-jdkinternals option analyzes all classes in the .jar for class-level dependencies on JDK internal APIs. If any are found, the results with suggested replacements are output in the console.

QUESTION 3

Which code fragment compiles?

- A.

```
Comparator comparator = new Comparator<?>() {  
    public int compare(Integer i, Integer j) {  
        return i.compareTo(j);  
    }  
};
```
- B.

```
var comparator = new Comparator<>() {  
    public int compare(Integer i, Integer j) {  
        return i.compareTo(j);  
    }  
};
```
- C.

```
Comparator<> comparator = new Comparator<Integer>() {  
    public int compare(Integer i, Integer j) {  
        return i.compareTo(j);  
    }  
};
```
- D.

```
Comparator<Integer> comparator = new Comparator<>() {  
    public int compare(Integer i, Integer j) {  
        return i.compareTo(j);  
    }  
};
```

A. Option A

B. Option B

C. Option C

D. Option D

Correct Answer: D

```
1 import java.io.*;
2 import java.util.*;
3 class abc {
4     public static void main(String[] args) {
5
6         Comparator<Integer> comparator = new Comparator<>() {
7             public int compare(Integer i, Integer j) {
8                 return i.compareTo(j);
9             }
10        };
11
12    }
13 }
14
```

QUESTION 4

Given:

```
public static void main(String[] args) {
    final List<String> fruits =
        List.of("Orange", "Apple", "Lemmon", "Raspberry");
    final List<String> types =
        List.of("Juice", "Pie", "Ice", "Tart");
    final var stream =
        IntStream.range(0, Math.min(fruits.size(), types.size()))
            .mapToObj((i) -> fruits.get(i) + " " + types.get(i) );
    stream. forEach(System.out::println);
}
```

What is the result?

- A. Orange Juice
- B. The compilation fails.
- C. Orange Juice Apple Pie Lemmon Ice Raspberry Tart
- D. The program prints nothing.

Correct Answer: C

```
12 > public class Person {
13 >     public static void main (String[] args) {
14 >         final List<String> fruits =
15 >             List.of("Orange", "Apple", "Lemmon", "raspberry");
16 >         final List<String> types =
17 >             List.of("Juice", "Pie", "Ice", "Tart");
18 >         final var stream =
19 >             IntStream.range(0, Math.min(fruits.size(), types.size()))
20 >                 .mapToObj ((i) -> fruits.get(i) + " " + types.get(i) );
21 >         stream. forEach(System.out::println);
22 >     }
23 >
24 > }
```

Result

compiled and executed in 1.227 sec(s)

```
Orange Juice
Apple Pie
Lemmon Ice
raspberry Tart
```

QUESTION 5

The screenshot shows an IDE with two tabs: Employee.java and Main.java. The Main.java tab is active, showing the following code:

```
1 import java.util.List;
2 Employee.java:1: java.util.function.BinaryOperator;
3
4 public class Main {
5     public static void main (String... args) {
6         List<Employee> list = List.of(new Employee("John", 80000.0), new Employee("Scott", 90000.0));
7         double starts = 0.0;
8         double ratio = 1.0;
9         BinaryOperator<Double> bo = (a, b) -> a + b;
10        double totalSalary = list.stream().map(e -> e.getSalary() * ratio).reduce(starts, bo);
11        //line 1
12        System.out.println("Total salary = " + totalSalary);
13    }
14 }
15 }
16 }
```

The console output shows:

```
Console 1
Total salary = 170000.0
Completed with exit code: 0
```

Which interface in the java.util.function package will return a void return type?

- A. Supplier
- B. Predicate
- C. Function
- D. Consumer

Correct Answer: D

Reference: <https://www.geeksforgeeks.org/java-8-consumer-interface-in-java-with-examples/>

[Latest 1Z0-816 Dumps](#)

[1Z0-816 PDF Dumps](#)

[1Z0-816 VCE Dumps](#)