KillTest

Higher Quality, Better Service!

Q&A

http://www.killtest.com

We offer free update service for one year.

Exam : **1Z0-816**

Title : Java SE 11 Programmer II

Version : DEMO

1.Given the code fragment:
Path currentFile = Paths.get(“/scratch/exam/temp.txt”);
Path outputFile = Paths get(“/scratch/exam/new.txt”);
Path directory = Paths.get(“/scratch/”);
Files.copy(currentFile, outputFile);
Files.copy(outputFile, directory);
Files.delete (outputFile);
The /scratch/exam/temp.txt file exists. The /scratch/exam/new.txt and /scratch/new.txt files do not exist.
What is the result?
A. /scratch/exam/new.txtand /scratch/new.txt are deleted.
B. The program throws a FileaAlreadyExistsException.
C. The program throws a NoSuchFileException.
D. A copy of /scratch/exam/new.txtexists in the /scratch directory and /scratch/exam/new.txt is deleted.
Answer: C

2.Which two are functional interfaces? (Choose two.)

```
27    public class Main {
28      public static void main(String[] args) {
29     Path currentFile = Paths.get("/scratch/exam/temp.txt");
30     Path outputFile = Paths.get("/scratch/exam/new.txt");
31     Path directory = Paths.get("/scratch/");
32
33     Files.copy(currentFile, outputFile);
34     Files.copy(outputFile, directory);
35     Files.delete (outputFile);
36    }
37    }
38
```

A)

```
@FunctionalInterface
interface MyRunnable {
   public void run();
}
```

B)

```
@FunctionalInterface
interface MyRunnable {
   public void run();
   public void call();
}
```

C)

```
interface MyRunnable {
   public default void run() {}
   public void run(String s);
}
```

D)

```
@FunctionalInterface
interface MyRunnable {
}
```

E)

```
interface MyRunnable {
   @FunctionalInterface
   public void run();
}
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E

Answer: CE

Explanation:

Reference: http://tutorials.jenkov.com/java-functional-programming/functional-interfaces.html

3.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

Answer: AB

4.Given:

```
interface MyInterface1 {
   public int method() throws Exception;
   private void pMethod() { /* an implementation of pMethod */ }
}
interface MyInterface2 {
   public static void sMethod() { /* an implementation of sMethod*/}
   public boolean equals();
}
interface MyInterface3 {
   public void method();
   public void method(String str);
}
interface MyInterface4 {
   public void dMethod() { /* an implementation of dMethod */ }
   public void method();
}
interface MyInterface5 {
   public static void sMethod();
   public void method(String str);
}
```

Which two interfaces can be used in lambda expressions? (Choose two.)

A. MyInterface1

B. MyInterface3

C. MyInterface5

D. MyInterface2

E. MyInterface4

Answer: CD

Explanation:

Reference: https://dzone.com/articles/functional-interface-and-lambda-expression

5.Given this enum declaration:

```
1. enum Alphabet {
2.    A, B, C
3.
4. }
```

Examine this code:

System.out.println(Alphabet.getFirstLetter());

What code should be written at line 3 to make this code print A?

A. final String getFirstLetter() { return A.toString(); }

B. static String getFirstLetter() { return Alphabet.values()[1].toString(); }

C. static String getFirstLetter() { return A.toString(); }

D. String getFirstLetter() { return A.toString(); }

Answer: C