NOTE: SCENARIO IS WHAT THE PUZZLE TAKER SEES.

SCENARIO:

Many applications need to write data into temporary files. In some cases these files contain sensitive information and, therefore, must be deleted before the application terminates. Consider such an application that writes sensitive data into a temporary file and deletes this file when it finishes processing it. The snippet of code below shows the implementation of the `processData` method. This method receives data as an array of bytes and processes it. Since the data (in both original or processed form) is sensitive, after the application exits, the temporary files must be deleted. Consider the snippet of code below and answer the following questions, assuming that the code has all required permissions to execute.

Code:

```java
// OMITTED: Import whatever is needed.
public static class DataProcessingUtils {
 public static byte[] processData (byte[] data)
   throws IOException {
  //OMITTED: prepare 'data' for processing

  Path file = Files.createTempFile("file", ".tmp");
  Files.write(file, data, StandardOpenOption.WRITE,
   StandardOpenOption.DELETE_ON_CLOSE);

  //OMITTED: continue processing 'data'
  //OMITTED: return the result of processing
 }
}
```

Questions:

1. What will the `processData` method do when executed?

2. If the JVM crashes after the execution of the `processData` method is completed, which one of the following is true?

a. The temporary file will be deleted when the program crashes.
b. An exception will be thrown and the program will print “Error in the processData method”.
c. The temporary file will not be deleted.
d. The temporary file may or may not be deleted.
e. The temporary file will be deleted only when the garbage collector runs.

[Other statistical questions will be imported here while creating the survey]

NOTE: ANSWER IS TO BE SHOWN TO THE PUZZLE TAKERS AT THE END OF THE

SESSION.

ANSWER:

d

According to the API documentation when the `DELETE_ON_CLOSE` option is present then the implementation makes a best effort attempt to delete the file when closed by the appropriate `close` method. If the close method is not invoked then a best effort attempt is made to delete the file when the Java virtual machine terminates (either normally, as defined by the Java Language Specification, or where possible, abnormally). Thus it is not guaranteed that the temporary file will be deleted.

NOTE: THE REST OF THIS DOCUMENT CONTAINS EXTRA INFORMATION FOR THE PROJECT RESEARCHERS. IT IS NOT TO BE SHOWN TO THE PUZZLE TAKERS.

TAGS:

java, file-class, io-operation, file-operation

CATEGORIES:

Blindspot - YES
Type - File
Number of distinct functions - 2
Number of total functions - 2
Blindspot function - Files.write
Function call omitted - NO
Blindspot type - Incorrect usage
Number of Parameters in the blindspot function - 4 parameters
Cyclomatic complexity - 2

NAME:

Path - An object that may be used to locate a file in a file system. It will typically represent a system dependent file path.

DESCRIPTION:

A `Path` represents a path that is hierarchical and composed of a sequence of directory and filename elements separated by a special separator or delimiter. A root component, that identifies a file system hierarchy, may also be present.

BLINDSPOT:

According to the API documentation, when the `DELETE_ON_CLOSE` option is present then the implementation makes a best effort attempt to delete the file when closed by the appropriate close method. If the close method is not invoked then a best effort attempt is made to delete the file when the Java virtual machine terminates (either normally, as defined by the Java Language Specification, or where possible, abnormally). As explicitly mentioned above there's no guarantee that the file will be deleted when the application exists or the JVM crashes.

CORRECT USE EXAMPLE:
#N/A

MORE INFORMATION:
#N/A

REFERENCES:

1. StandardOpenOption