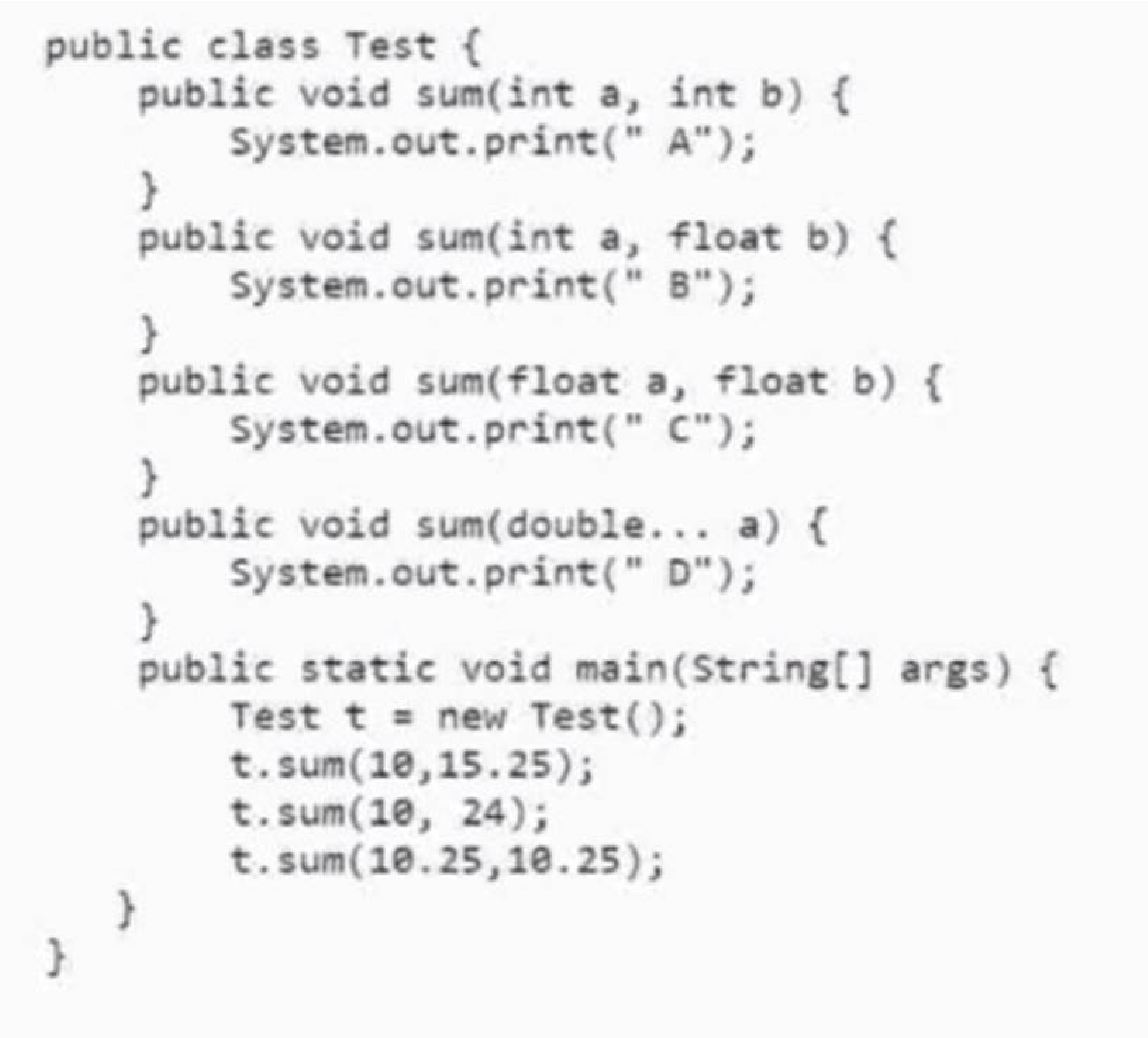
Oracle

Exam Questions 1z0-829

Java SE 17 Developer

NEW QUESTION 1
Given:



```
public class Test {
    public void sum(int a, int b) {
        System.out.print(" A");
    }
    public void sum(int a, float b) {
        System.out.print(" B");
    }
    public void sum(float a, float b) {
        System.out.print(" C");
    }
    public void sum(double... a) {
        System.out.print(" D");
    }
    public static void main(String[] args) {
        Test t = new Test();
        t.sum(10,15.25);
        t.sum(10, 24);
        t.sum(10.25,10.25);
   }
}
```

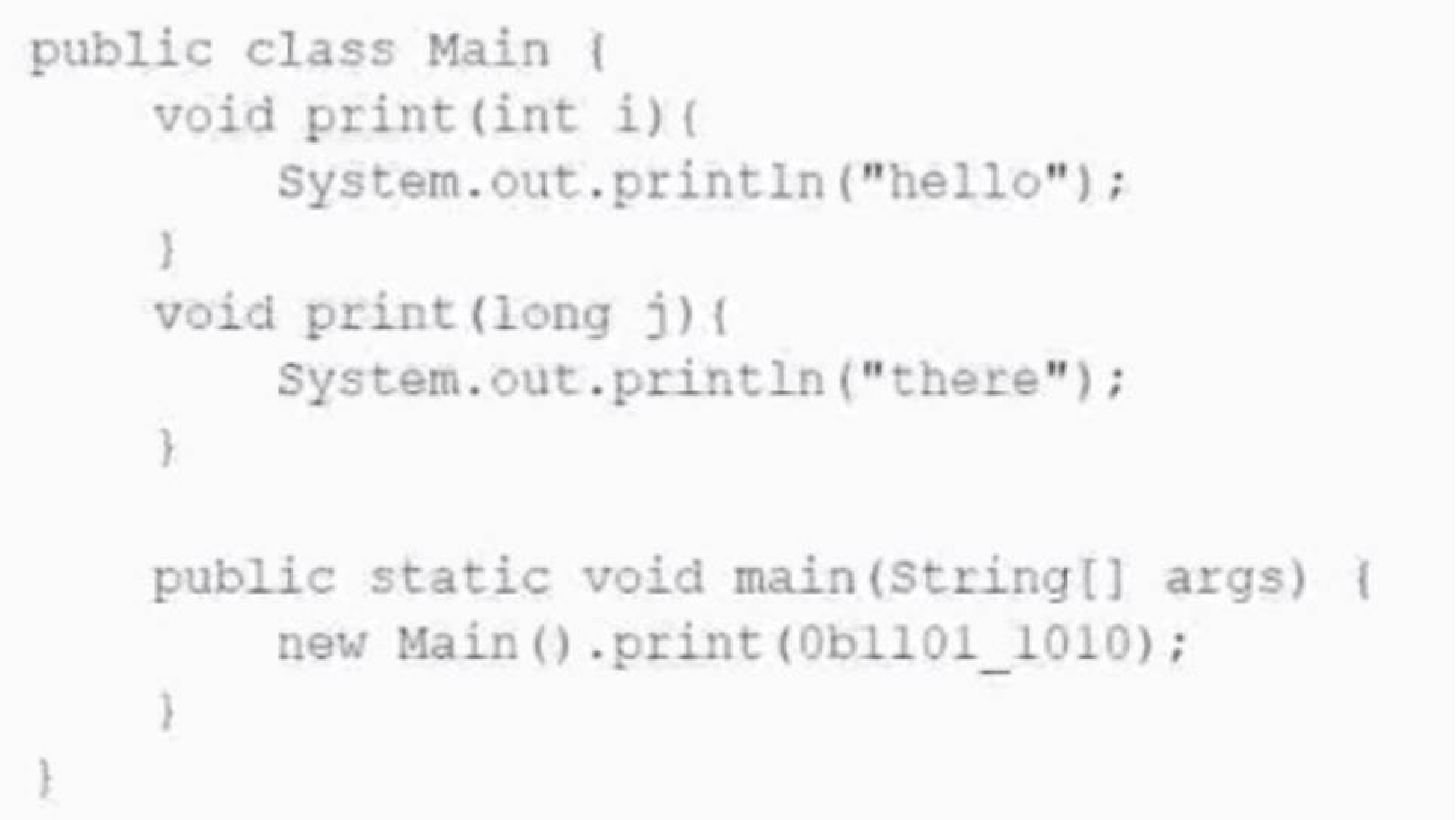
What is the result?

A. B A C
B. D A D
C. B A D
D. D D D

Answer: C

Explanation:
The answer is C because the code demonstrates the concept of method overloading and type conversion in Java. Method overloading allows different methods to have the same name but different parameters. Type conversion allows values of one data type to be assigned to another data type, either automatically or explicitly. In the code, the class Test has four methods named sum, each with different parameter types: int, float, and double. The main method creates an instance of Test and calls the sum method with different arguments. The compiler will choose the most specific method that matches the arguments, based on the following rules:
? If there is an exact match between the argument types and the parameter types, that method is chosen.
? If there is no exact match, but there is a method with compatible parameter types, that method is chosen. Compatible types are those that can be converted from one to another automatically, such as int to long or float to double.
? If there is more than one method with compatible parameter types, the most specific method is chosen. The most specific method is the one whose parameter types are closest to the argument types in terms of size or precision.
In the code, the following method calls are made:
? test.sum(10, 10.5) -> This matches the sum(int a, float b) method exactly, so it is chosen. The result is 20.5, which is converted to int and printed as 20 (B).
? test.sum(10) -> This does not match any method exactly, but it matches the sum(double a) method with compatible types, as int can be converted to double automatically. The result is 10.0, which is printed as 10 (A).
? test.sum(10.5, 10) -> This does not match any method exactly, but it matches two methods with compatible types: sum(float a, float b) and sum(double a, double b). The latter is more specific, as double is closer to the argument types than float. The result is 20.5, which is printed as 20 (D).
Therefore, the output is B A D. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Method Overloading in Java
? Type conversion in Java with Examples
? Java Method Overloading with automatic type conversions

NEW QUESTION 2
Given:



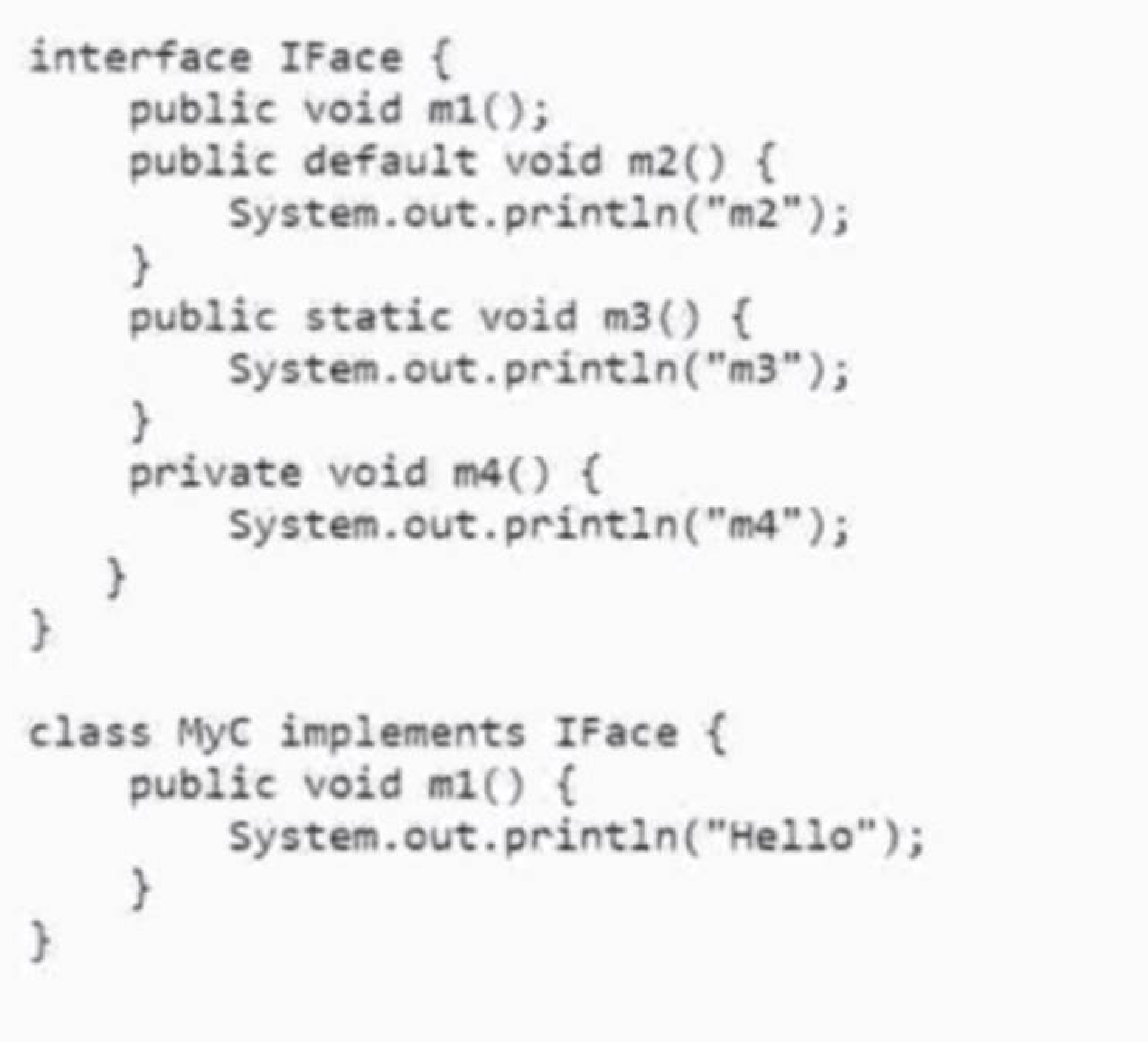
```
public class Main {
    void print(int i){
        System.out.println("hello");
    }
    void print(long j){
        System.out.println("there");
    }

    public static void main(String[] args) {
        new Main().print(0b1101_1010);
    }
}
```

A. Hello
B. Compilation fails
C. A NumberFormatException is thrown
D. there

Answer: B

Explanation:
The code fragment will fail to compile because the parseInt method of the Integer class is a static method, which means that it can be invoked without creating an object of the class. However, the code is trying to invoke the parseInt method on an object of type Integer, which is not allowed. The correct way to invoke the parseInt method is by using the class name, such as Integer.parseInt (s). Therefore, the code fragment will produce a compilation error. References: Integer (Java SE 17 & JDK 17) - Oracle

NEW QUESTION 3
Given:



```
interface IFace {
    public void m1();
    public default void m2() {
        System.out.println("m2");
    }
    public static void m3() {
        System.out.println("m3");
    }
    private void m4() {
        System.out.println("m4");
   }
}

class MyC implements IFace {
    public void m1() {
        System.out.println("Hello");
    }
}
```

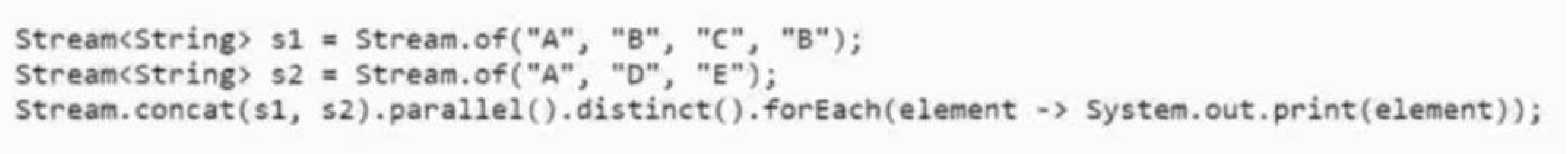
Which two method invocation execute?

A. IFace myclassobj = new Myc (); myclassObj.m3 ();
B. Ifnce.m3 ();
C. iFace mucloassObj = new Myc (); myClassObj.m4();
D. new MyC() .m2 ();
E. IFace .,4():
F. IFace.m2();

Answer: DE

Explanation:
The code given is an interface and a class that implements the interface. The interface has three methods, m1(), m2(), and m3(). The class has one method, m1(). The only two method invocations that will execute are D and E. D is a call to the m2() method in the class, and E is a call to the m3() method in the interface. References: https://education.oracle.com/products/trackp_OCPJSE17, 3, 4, 5

NEW QUESTION 4
Given the code fragment:



```
Stream<String> s1 = Stream.of("A", "B", "C", "B");
Stream<String> s2 = Stream.of("A", "D", "E");
Stream.concat(s1, s2).parallel().distinct().forEach(element -> System.out.print(element));
```

What is the result:

A. ADEABCB // the order of element is unpredictable
B. ABCE
C. ABCDE // the order of elements is unpredictable
D. ABBCDE // the order of elements is unpredictable

Answer: D

Explanation:
The answer is D because the code fragment uses the Stream API to create two streams, s1 and s2, and then concatenates them using the concat() method. The resulting stream is then processed in parallel using the parallel() method, and the distinct() method is used to remove duplicate elements. Finally, the forEach() method is used to print the elements of the resulting stream to the console. Since the order of elements in a parallel stream is unpredictable, the output could be any of the options given, but option D is the most likely. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer

? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Parallelizing Streams

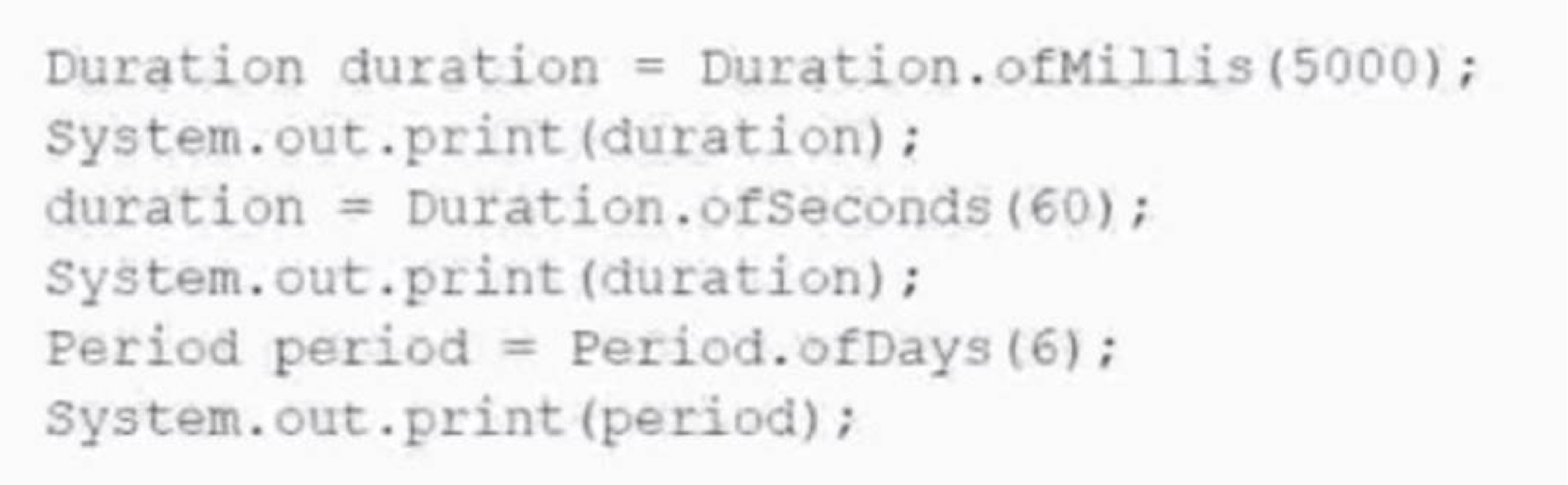
NEW QUESTION 5
Which statement is true about migration?

A. Every module is moved to the module path in a top-down migration.
B. Every module is moved to the module path in a bottom-up migration.
C. The required modules migrate before the modules that depend on them in a top-down^migration.
D. Unnamed modules are automatic modules in a top-down migration.

Answer: B

Explanation:
The answer is B because a bottom-up migration is a strategy for modularizing an existing application by moving its dependencies to the module path one by one, starting from the lowest-level libraries and ending with the application itself. This way, each module can declare its dependencies on other modules using the module-info.java file, and benefit from the features of the Java Platform Module System (JPMS), such as reliable configuration, strong encapsulation, and service loading.
Option A is incorrect because a top-down migration is a strategy for modularizing an existing application by moving it to the module path first, along with its dependencies as automatic modules. Automatic modules are non-modular JAR files that are treated as modules with some limitations, such as not having a module descriptor or a fixed name. A top-down migration allows the application to use the module path without requiring all of its dependencies to be modularized first.
Option C is incorrect because a top-down migration does not require any specific order of migrating modules, as long as the application is moved first and its dependencies are moved as automatic modules. A bottom-up migration, on the other hand, requires the required modules to migrate before the modules that depend on them.
Option D is incorrect because unnamed modules are not automatic modules in any migration strategy. Unnamed modules are modules that do not have a name or a module descriptor, such as classes loaded from the class path or dynamically generated classes. Unnamed modules have unrestricted access to all other modules, but they cannot be accessed by named modules, except through reflection with reduced security checks. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Migrating to Modules (How and When) - JavaDeploy
? Java 9 Modularity: Patterns and Practices for Developing Maintainable
Applications

NEW QUESTION 6
Given the code fragment:



```
Duration duration = Duration.ofMillis(5000);
System.out.print(duration);
duration = Duration.ofSeconds(60);
System.out.print(duration);
Period period = Period.ofDays(6);
System.out.print(period);
```

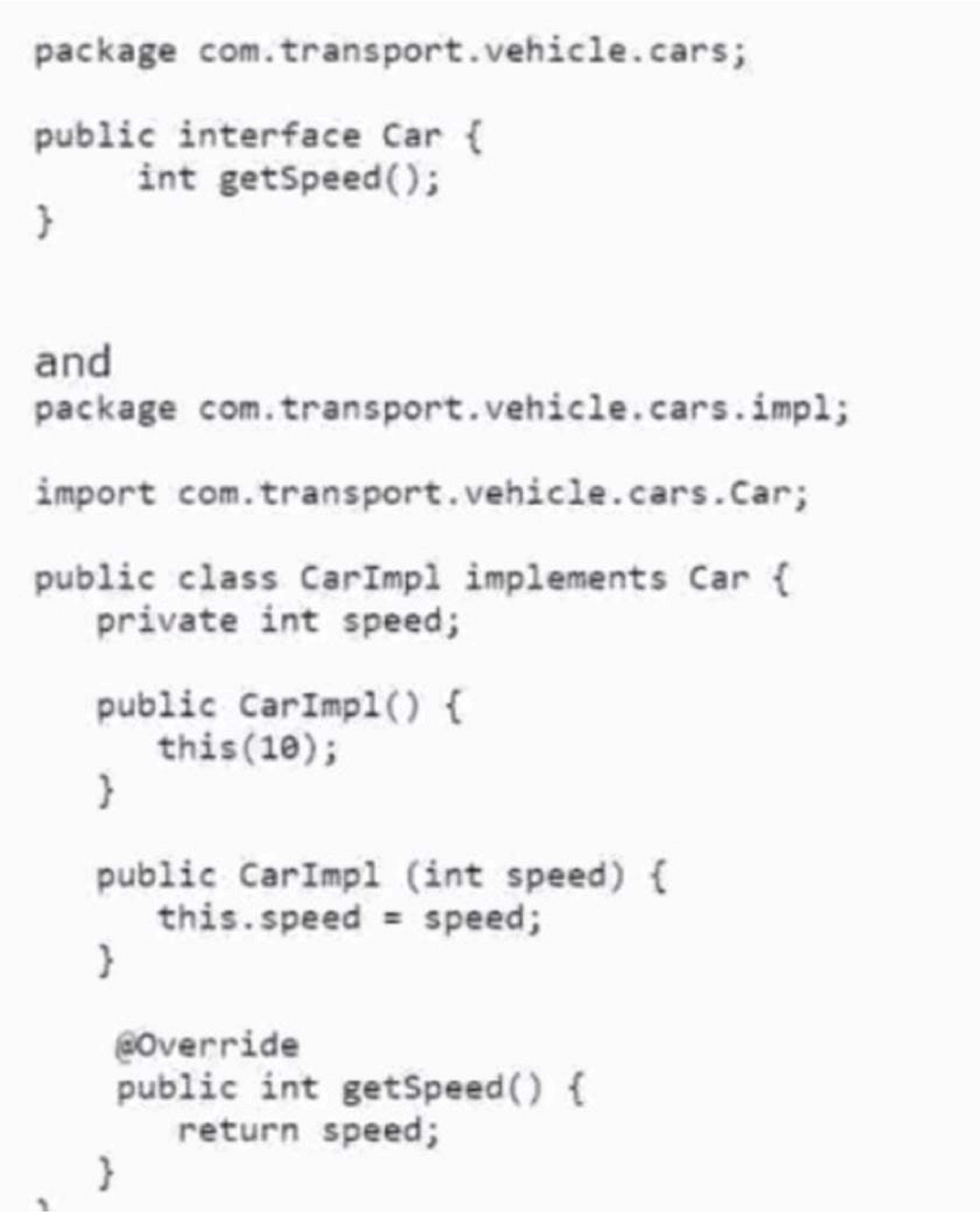
What is the result?

A. $SIM6D
B. PT5000PT60MP6D
C. PT5SPTIMP6D
D. 5000$60M6D

Answer: B

Explanation:
The code fragment is creating a Duration object with a value of 5000 milliseconds, then printing it. Then, it is creating another Duration object with a value of 60 seconds, then printing it. Finally, it is creating a Period object with a value of 6 days, then printing it. The output will be ??PT5000PT60MP6D??. References: https://docs.oracle.com/javase/8/docs/api/java/time/Duration.html, https://docs.oracle.com/javase/8/docs/api/java/time/Period.html

NEW QUESTION 7
Given:



```
package com.transport.vehicle.cars;

public interface Car {
     int getSpeed();
}


and
package com.transport.vehicle.cars.impl;

import com.transport.vehicle.cars.Car;

public class CarImpl implements Car {
   private int speed;

   public CarImpl() {
      this(10);
   }

   public CarImpl (int speed) {
      this.speed = speed;
   }

    @Override
    public int getSpeed() {
       return speed;
   }
}
```

Which two should the module-info file include for it to represent the service provider interface?

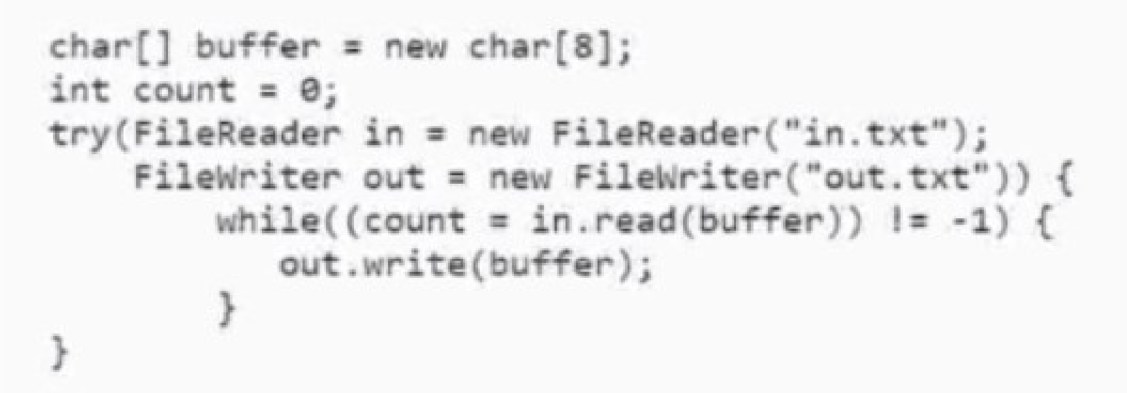
A. Requires cm.transport.vehicle,cars:
B. Provides.com.transport.vehicle.cars.Car with com.transport.vehicle.car
C. impt, CatImpl;
D. Requires cm.transport.vehicle,cars:
E. Provides.com.transport.vehicle.cars.Car impl,CarImp1 to com.transport.vehicle.car
F. Cars
G. exports com.transport.vehicle.cars.Car;
H. Exports com.transport.vehicle.cars;
I. Exports com.transport.vehicle;

Answer: BE

Explanation:
The answer is B and E because the module-info file should include a provides directive and an exports directive to represent the service provider interface. The provides directive declares that the module provides an implementation of a service interface, which is com.transport.vehicle.cars.Car in this case. The with clause specifies the fully qualified name of the service provider class, which is com.transport.vehicle.cars.impl.CarImpl in this case. The exports directive declares that the module exports a package, which is com.transport.vehicle.cars in this

case, to make it available to other modules. The package contains the service interface that other modules can use.
Option A is incorrect because requires is not the correct keyword to declare a service provider interface. Requires declares that the module depends on another module, which is not the case here.
Option C is incorrect because it has a typo in the module name. It should be com.transport.vehicle.cars, not cm.transport.vehicle.cars.
Option D is incorrect because it has a typo in the keyword provides. It should be provides, not Provides. It also has a typo in the service interface name. It should be com.transport.vehicle.cars.Car, not com.transport.vehicle.cars.Car impl. It also has an unnecessary to clause, which is used to limit the accessibility of an exported package to specific modules.
Option F is incorrect because it exports the wrong package. It should export com.transport.vehicle.cars, not com.transport.vehicle.cars.impl. The impl package contains the service provider class, which should not be exposed to other modules.
Option G is incorrect because it exports the wrong package. It should export com.transport.vehicle.cars, not com.transport.vehicle. The vehicle package does not contain the service interface or the service provider class. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Java Modules - Service Interface Module - GeeksforGeeks
? Java Service Provider Interface | Baeldung

NEW QUESTION 8
Given the content of the in. tart file: 23456789
and the code fragment:



```
char[] buffer = new char[8];
int count = 0;
try(FileReader in = new FileReader("in.txt");
    FileWriter out = new FileWriter("out.txt")) {
        while((count = in.read(buffer)) != -1) {
           out.write(buffer);
        }
}
```

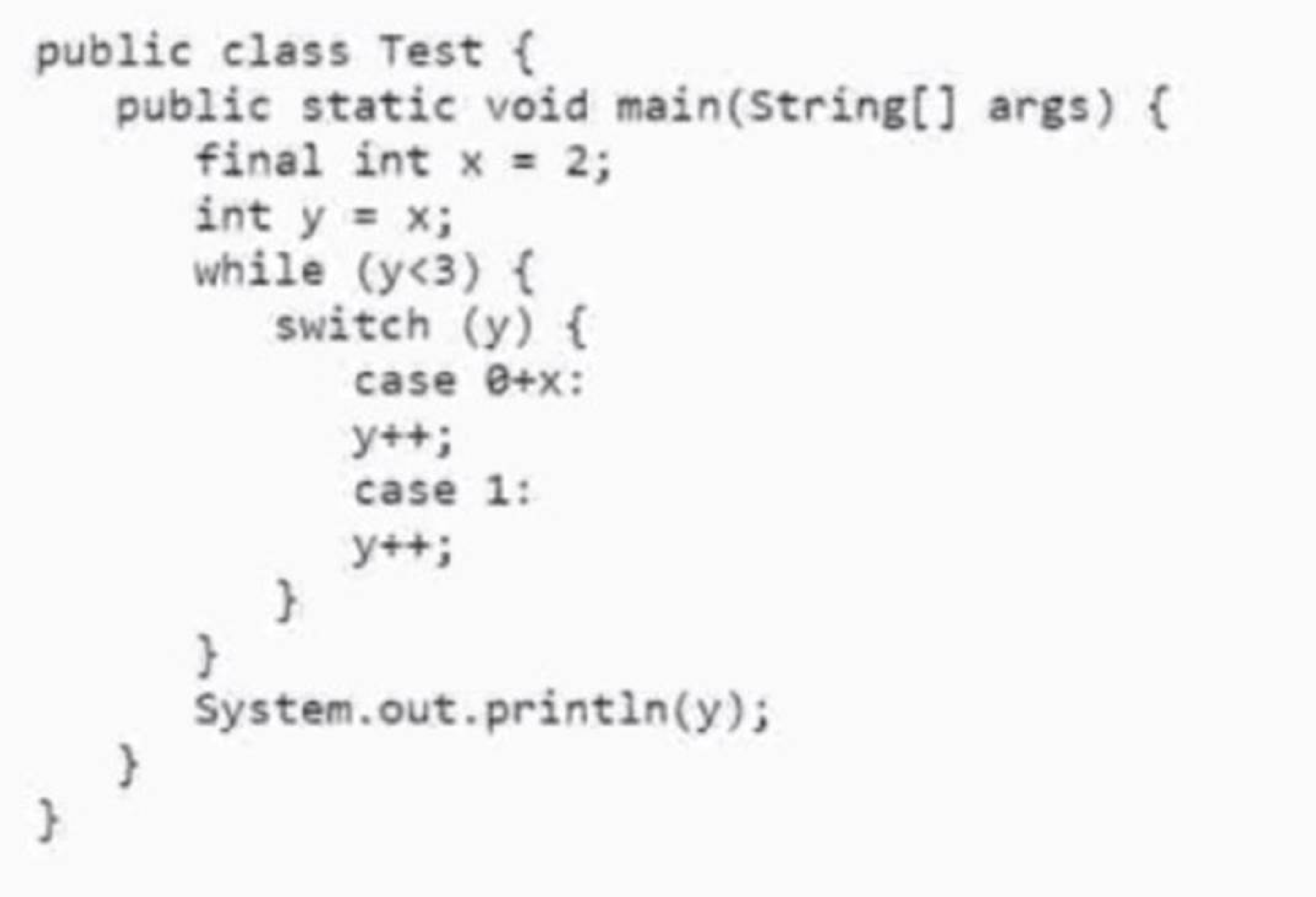
What is the content of the out .txt file?

A. 01234567801234
B. 012345678
C. 0123456789234567
D. 0123456789
E. 012345678901234
F. 01234567

Answer: D

Explanation:
The answer is D because the code fragment reads the content of the in.txt file and writes it to the out.txt file. The content of the in.txt file is ??23456789??. The code fragment uses a char array buffer of size 8 to read the content of the in.txt file. The while loop reads the content of the in.txt file and writes it to the out.txt file until the end of the file is reached. Therefore, the content of the out.txt file will be ??0123456789??.

NEW QUESTION 9
Given:



```
public class Test {
    public static void main(String[] args) {
        final int x = 2;
        int y = x;
        while (y<3) {
            switch (y) {
                case 0+x:
                y++;
                case 1:
                y++;
            }
        }
        System.out.println(y);
    }
}
```

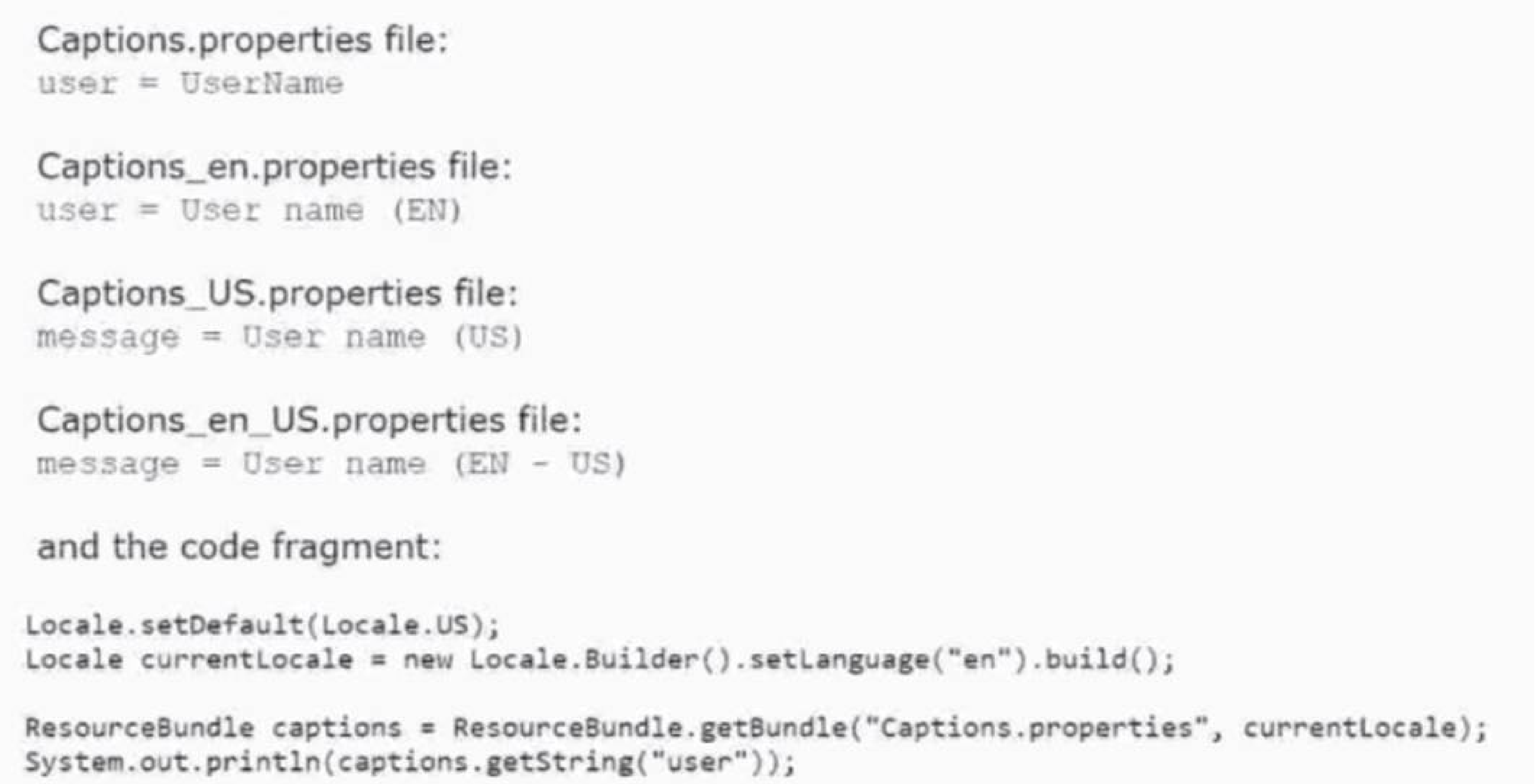
What is the result?

A. 4
B. 2
C. 6
D. Nothing is printed because of an indefinite loop.
E. Compilation fails.
F. 5
G. A runtime exception is thrown.
H. 3

Answer: E

Explanation:
The code will not compile because the variable ??x?? is declared as final and then it is being modified in the switch statement. This is not allowed in Java. A final variable is a variable whose value cannot be changed once it is initialized1. The switch statement tries to assign different values to ??x?? depending on the value of ??y??, which violates the final modifier. The compiler will report an error: The final local variable x cannot be assigned. It must be blank and not using a compound assignment. References: The final Keyword (The Java™ Tutorials > Learning the Java Language > Classes and Objects)

NEW QUESTION 10
Given:



Captions.properties file:

```
user = UserName
```

Captions_en.properties file:

```
user = User name (EN)
```

Captions_US.properties file:

```
message = User name (US)
```

Captions_en_US.properties file:

```
message = User name (EN - US)
```

and the code fragment:

```
Locale.setDefault(Locale.US);
Locale currentLocale = new Locale.Builder().setLanguage("en").build();

ResourceBundle captions = ResourceBundle.getBundle("Captions.properties", currentLocale);
System.out.println(captions.getString("user"));
```

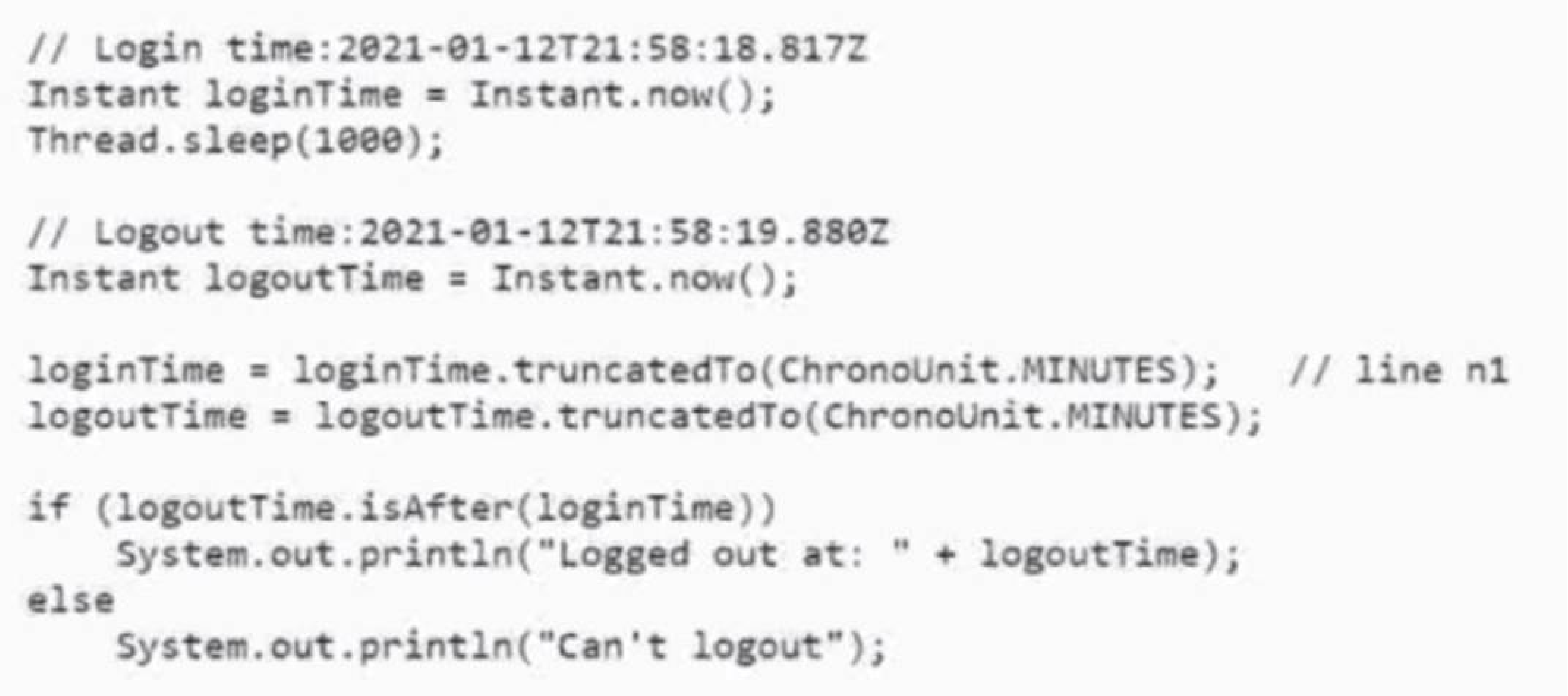
What is the result?

A. User name (US)
B. The program throws a MissingResourceException.
C. User name (EN – US)
D. UserName
E. User name (EN)

Answer: B

Explanation:
The answer is B because the code fragment contains a logical error that causes a MissingResourceException at runtime. The code fragment tries to load a resource bundle with the base name ??Captions.properties?? and the locale ??en_US??. However, there is no such resource bundle available in the classpath. The available resource bundles are:
? Captions.properties
? Captions_en.properties
? Captions_US.properties
? Captions_en_US.properties
The ResourceBundle class follows a fallback mechanism to find the best matching resource bundle for a given locale. It first tries to find the resource bundle with the exact locale, then it tries to find the resource bundle with the same language and script, then it tries to find the resource bundle with the same language, and finally it tries to find the default resource bundle with no locale. If none of these resource bundles are found, it throws a MissingResourceException.
In this case, the code fragment is looking for a resource bundle with the base name ??Captions.properties?? and the locale ??en_US??. The ResourceBundle class will try to find the following resource bundles in order:
? Captions.properties_en_US
? Captions.properties_en
? Captions.properties
However, none of these resource bundles exist in the classpath. Therefore, the ResourceBundle class will throw a MissingResourceException.
To fix this error, the code fragment should use the correct base name of the resource bundle family, which is ??Captions?? without the ??.properties?? extension. For example: ResourceBundle captions = ResourceBundle.getBundle(??Captions??, currentLocale); This will load the appropriate resource bundle for the current locale, which is ??Captions_en_US.properties?? in this case. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? ResourceBundle (Java Platform SE 8)
? About the ResourceBundle Class (The Java™ Tutorials > Internationalization)

NEW QUESTION 10
Given the code fragment:



```
// Login time:2021-01-12T21:58:18.817Z
Instant loginTime = Instant.now();
Thread.sleep(1000);

// Logout time:2021-01-12T21:58:19.880Z
Instant logoutTime = Instant.now();

loginTime = loginTime.truncatedTo(ChronoUnit.MINUTES);   // line n1
logoutTime = logoutTime.truncatedTo(ChronoUnit.MINUTES);

if (logoutTime.isAfter(loginTime))
    System.out.println("Logged out at: " + logoutTime);
else
    System.out.println("Can't logout");
```

What is the result?

A. Logged out at: 2021-0112T21:58:19.880z
B. Logged out at: 2021-01-12T21:58:00z
C. A compilation error occurs at Line n1.
D. Can??t logout

Answer: B

Explanation:
The code fragment is using the Java SE 17 API to get the current time and then truncating it to minutes. The result will be the current time truncated to minutes, which is why option B is correct. References:
? https://education.oracle.com/products/trackp_OCPJSE17
? https://mylearn.oracle.com/ou/learning-path/java-se-17-developer/99487
? https://docs.oracle.com/javase/17/docs/api/java.base/java/time/Instant.html#truncatedTo(java.time.temporal.TemporalUnit)

NEW QUESTION 14
......