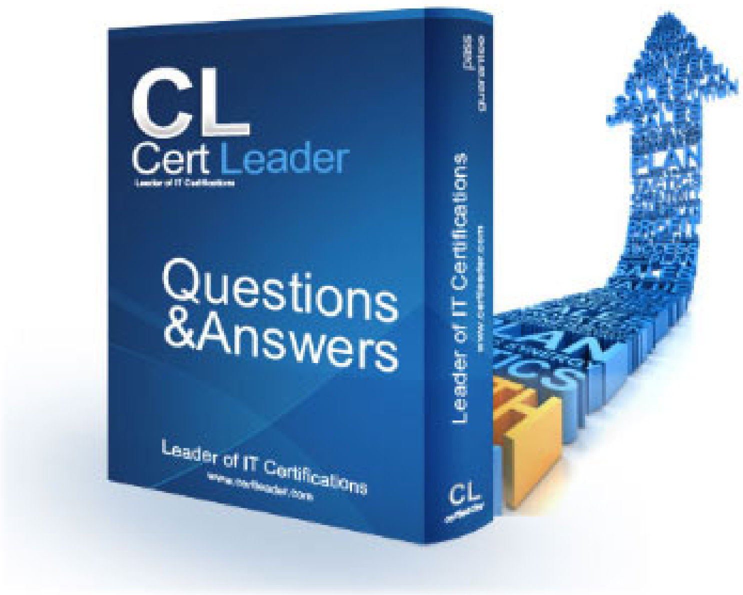
1z0-829 Dumps

Java SE 17 Developer

https://www.certleader.com/1z0-829-dumps.html



NEW QUESTION 1
Which statement is true?

A. IllegalStateException is thrown if a thread in waiting state is moved back to runnable.
B. thread in waiting state consumes CPU cycles.
C. A thread in waiting state must handle InterrupedException.
D. After the timed wait expires, the waited thread moves to the terminated state.

Answer: C

Explanation:
A thread in waiting state is waiting for another thread to perform a particular action, such as calling notify() or notifyAll() on a shared object, or terminating a joined thread. A thread in waiting state can be interrupted by another thread, which will cause the waiting thread to throw an InterruptedException and return to the runnable state. Therefore, a thread in waiting state must handle InterruptedException, either by catching it or declaring it in the throws clause. References: Thread.State (Java SE 17 & JDK 17), [Thread (Java SE 17 & JDK 17)]

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 the code fragment:

```
String myStr = "Hello Java 17";
String myTextBlk1 = """
                    Hello Java 17""";
String myTextBlk2 = """
                    Hello Java 17
                    """;
System.out.print(myStr.equals(myTextBlk1)+":");
System.out.print(myStr.equals(myTextBlk2)+":");
System.out.print(myTextBlk1.equals(myTextBlk2)+":");
System.out.println(myTextBlk1.intern() == myTextBlk2.intern());
```

A. True:false:true:true
B. True:true:false:false
C. True:false:true:false
D. True:false:false:false

Answer: C

Explanation:
The code fragment compares four pairs of strings using the equals() and intern() methods. The equals() method compares the content of two strings, while the intern() method returns a canonical representation of a string, which means that it returns a reference to an existing string with the same content in the string pool. The string pool is a memory area where strings are stored and reused to save space and improve performance. The results of the comparisons are as follows:
? s1.equals(s2): This returns true because both s1 and s2 have the same content, ??Hello Java 17??.
? s1 == s2: This returns false because s1 and s2 are different objects with different references, even though they have the same content. The == operator compares the references of two objects, not their content.
? s1.intern() == s2.intern(): This returns true because both s1.intern() and s2.intern() return a reference to the same string object in the string pool, which has the content ??Hello Java 17??. The intern() method ensures that there is only one copy of each distinct string value in the string pool.
? ??Hello Java 17?? == s2: This returns false because ??Hello Java 17?? is a string literal, which is automatically interned and stored in the string pool, while s2 is a string object created with the new operator, which is not interned by default and stored in the heap. Therefore, they have different references and are not equal using the == operator.
References: String (Java SE 17 & JDK 17) - Oracle

NEW QUESTION 4
Given:

```
public class App {
    public int x = 100;

    public static void main(String[] args) {
        int x = 1000;
        App t = new App();
        t.myMethod(x);
        System.out.println(x);
    }
    public void myMethod(int x) {
        x++;
        System.out.println(x);
        System.out.println(this.x);
    }
}
```

What is the result?
A.
1001
1001
1000
B.
101
101
1000
C.
100
100
1000
D.
1001
100

1000

A.

Answer: D

Explanation:
The code fragment is using the bitwise operators & (AND), | (OR), and ^ (XOR) to perform operations on the binary representations of the integer values. The & operator returns a 1 in each bit position where both operands have a 1, the | operator returns a 1 in each bit position where either operand has a 1, and the ^ operator returns a 1 in each bit position where only one operand has a 1. The binary representations of the integer values are as follows:
? 1000 = 1111101000
? 100 = 1100100
? 101 = 1100101
The code fragment performs the following operations:
? x = x ^ y; // x becomes 1111010101, which is 1001 in decimal
? y = x ^ y; // y becomes 1100100, which is 100 in decimal
? x = x ^ y; // x becomes 1100101, which is 101 in decimal
The code fragment then prints out the values of x, y, and z, which are 1001, 100, and 1000 respectively. Therefore, option D is correct.

NEW QUESTION 5
Assuming that the data, txt file exists and has the following content:
Text1 Text2 Text3
Given the code fragment:

```
try {
     Path p = new File("data.txt").toPath();
     Stream lines = Files.lines(p);
     String data = lines.collect(Collectors.joining("-"));
     System.out.println(data);
     String data2 = Files.readAllLines(p).get(3);
     System.out.println(data2);
} catch (IOException ex) {
     System.out.println(ex);
}
```

What is the result?

A. text1- text2- text3- text3
B. text1-text2-text3 text1text2 text3
C. text1-text2-text3A java.lang.indexoutofBoundsException is thrown.
D. text1-text2-text3 text3

Answer: D

Explanation:
The answer is D because the code fragment reads the file ??data.txt?? and collects all the lines in the file into a single string, separated by hyphens. Then, it prints the resulting string. Next, it attempts to read the fourth line in the file (index 3) and print it. However, since the file only has three lines, an IndexOutOfBoundsException is thrown. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Read contents of a file using Files class in Java

NEW QUESTION 6
Which statement is true about modules?

A. Automatic and unnamed modules are on the module path.
B. Only unnamed modules are on the module path.
C. Automatic and named modules are on the module path.
D. Only named modules are on the module path.
E. Only automatic modules are on the module path.

Answer: C

Explanation:
A module path is a sequence of directories that contain modules or JAR files. A named module is a module that has a name and a module descriptor (module-info.class) that declares its dependencies and exports. An automatic module is a module that does not have a module descriptor, but is derived from the name and contents of a JAR file. Both named and automatic modules can be placed on the module path, and they can be resolved by the Java runtime. An unnamed module is a special module that contains all the classes that are not in any other module, such as those on the class path. An unnamed module is not on the module path, but it can read all other modules.

NEW QUESTION 7
Given:

```
class A {public void mA() {System.out.println("mA");}}
class B extends A {public void mB() {System.out.println("mB");}}
class C extends B {public void mC() {System.out.println("mC");}}

public class App {
    public static void main(String[] args) {
        A bobj = new B();
        A cobj = new C();
        if (cobj instanceof B v) {
            v.mB();
            if (v instanceof C v1) { v1.mC(); }
        } else {
            cobj.mA();
        }
    }
}
```

What is the result?

A. Mb MC
B. Mb
C. Mb
D. MA
E. mA

Answer: E

Explanation:
The code snippet is an example of Java SE 17 code. The code is checking if the object is an instance of class C and if it is, it will print ??mC??. If it is not an instance of class C, it will print ??mA??. In this case, the object is not an instance of class C, so the output will be ??mA??. References: Pattern Matching for instanceof - Oracle Help Center

NEW QUESTION 8
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 9
Given the code fragment:

```
class Book {
      String author;
      String title;
      Book(String authorName, String title) {
           this.author = authorName;
           this.title = title;
      }
}

class SortBook {
      public static void main(String[] args) {
          List books = List.of(new Book("A1","T1"), new Book("A2", "T2"), new Book("A1","T2")); // Line n1
          books.sort((Book a, Book b) -> a.title.compareTo(b.title));        // Line n2
          System.out.println(books);
      }
}
```

Which action sorts the book list?

A. At Line n2, replace books,sort() with books.stream().sort(0.
B. At line n1, convert books type to mutable ArrayList type.
C. At Line n1, convert type to mutable array type.
D. At Line n2, replace compareTo () with compare ().

Answer: D

Explanation:
The code fragment is trying to sort a list of books using the Collections.sort() method. The correct answer is D, because the compareTo() method is not the correct way to compare two objects in a Comparator. The compare() method is the correct way to compare two objects in a Comparator and return an int value that indicates their order1. The compareTo() method is used to implement the Comparable interface, which defines the natural order of objects of a class2. The other options are incorrect because they either do not change the type of the list, which is already mutable, or they do not use the correct syntax for sorting a stream, which requires a terminal operation such as collect()3. References: Comparator (Java SE 17 & JDK 17), Comparable (Java SE 17 & JDK 17), Stream (Java SE 17 & JDK 17)

NEW QUESTION 10
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 15
Daylight Saving Time (DST) is the practice of advancing clocks at the start of spring by one hour and
adjusting them backward by one hour in autumn.
Considering that in 2021, DST in Chicago (Illinois) ended on November 7th at 2 AM, and given the fragment:

```
ZoneId zoneID = ZoneId.of("America/Chicago");
ZonedDateTime zdt = ZonedDateTime.of(
    LocalDate.of(2021, 11, 7),
    LocalTime.of(1, 30),
    zoneID
);
ZonedDateTime anHourLater = zdt.plusHours(1);
System.out.println(zdt.getHour() == anHourLater.getHour());
System.out.print(zdt.getOffset().equals(anHourLater.getOffset()));
```

What is the output?

A. true false
B. False false
C. true true
D. false true

Answer: A

Explanation:
The answer is A because the code fragment uses the ZoneId and ZonedDateTime classes to create two date-time objects with the same local date-time but different zone offsets. The ZoneId class represents a time-zone ID, such as America/Chicago, and the ZonedDateTime class represents a date-time with a time-zone in the ISO-8601 calendar system. The code fragment creates two ZonedDateTime objects with the same local date-time of 2021-11-07T01:30, but different zone IDs of America/Chicago and UTC. The code fragment then compares the two objects using the equals and isEqual methods.
The equals method compares the state of two objects for equality. In this case, it compares the local date-time, zone offset, and zone ID of the two ZonedDateTime objects. Since the zone offsets and zone IDs are different, the equals method returns false.
The isEqual method compares the instant of two temporal objects for equality. In this case, it compares the instant of the two ZonedDateTime objects, which is derived from the local date-time and zone offset. Since DST in Chicago ended on November 7th at 2 AM in 2021, the local date-time of 2021-11-07T01:30 in America/Chicago corresponds to the same instant as 2021-11-07T06:30 in UTC. Therefore, the isEqual method returns true.
Hence, the output is true false. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? ZoneId (Java Platform SE 8)
? ZonedDateTime (Java Platform SE 8)
? Time Zone & Clock Changes in Chicago, Illinois, USA
? Daylight Saving Time Changes 2023 in Chicago, USA

NEW QUESTION 18
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 21
......