Oracle

Exam Questions 1z0-829

Java SE 17 Developer

About Exambible

Your Partner of IT Exam

Found in 1998

Exambible is a company specialized on providing high quality IT exam practice study materials, especially Cisco CCNA, CCDA, CCNP, CCIE, Checkpoint CCSE, CompTIA A+, Network+ certification practice exams and so on. We guarantee that the candidates will not only pass any IT exam at the first attempt but also get profound understanding about the certificates they have got. There are so many alike companies in this industry, however, Exambible has its unique advantages that other companies could not achieve.

Our Advances

* 99.9% Uptime

All examinations will be up to date.

* 24/7 Quality Support

We will provide service round the clock.

* 100% Pass Rate

Our guarantee that you will pass the exam.

* Unique Gurantee

If you do not pass the exam at the first time, we will not only arrange FULL REFUND for you, but also provide you another exam of your claim, ABSOLUTELY FREE!

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 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 3
Given:

```
class Product {
    String name; double price;
    Product(String s, double d) {
        this.name = s;
        this.price = d;
    }
}
class ElectricProduct extends Product {
    ElectricProduct(String name, double price) {
        super(name, price);
    }
}
```

and the code fragment:

```
List<Product> p = List.of(
new ElectricProduct("CellPhone",100),
new ElectricProduct("ToyCar",90),
new ElectricProduct("Motor",200),
new ElectricProduct("Fan",300)
);

DoubleSummaryStatistics sts = p.stream().filter(a -> a instanceof ElectricProduct)
                                        .collect(Collectors.summarizingDouble(a ->
a.price));
String s1 = p.stream().filter(a -> a instanceof Product)
                      .collect(Collectors.mapping(p2 -> p2.name, Collectors.joining(",")));
System.out.println(sts.getMax());
System.out.println(s1);
```

A. 300.00CellPhone,ToyCar,Motor,Fan
B. 100.00CellPhone,ToyCar,Motor,Fan
C. 100.00 CellPhone,ToyCar
D. 300.00CellPhone.ToyCar

Answer: A

Explanation:
The code fragment is using the Stream API to perform a reduction operation on a list of ElectricProduct objects. The reduction operation consists of three parts: an identity value, an accumulator function, and a combiner function. The identity value is the initial value of the result, which is 0.0 in this case. The accumulator function is a BiFunction that takes two arguments: the current result and the current element of the stream, and returns a new result. In this case, the accumulator function is (a,b) -> a + b.getPrice (), which means that it adds the price of each element to the current result. The combiner function is a BinaryOperator that takes two partial results and combines them into one. In this case, the combiner function is (a,b) -> a + b, which means that it adds the two partial results together.
The code fragment then applies a filter operation on the stream, which returns a new stream that contains only the elements that match the given predicate. The predicate is p -
> p.getPrice () > 10, which means that it selects only the elements that have a price greater than 10. The code fragment then applies a map operation on the filtered stream, which returns a new stream that contains the results of applying the given function to each element. The function is p -> p.getName (), which means that it returns the name of each element.
The code fragment then calls the collect method on the mapped stream, which performs a mutable reduction operation on the elements of the stream using a Collector. The Collector is Collectors.joining (??,??), which means that it concatenates the elements of the stream into a single String, separated by commas.
The code fragment then prints out the result of the reduction operation and the result of the collect operation, separated by a new line. The result of the reduction operation is 300.00, which is the sum of the prices of all ElectricProduct objects that have a price greater than 10. The result of the collect operation is CellPhone,ToyCar,Motor,Fan, which is the concatenation of the names of all ElectricProduct objects that have a price greater than 10. Therefore, the output of the code fragment is: 300.00 CellPhone,ToyCar,Motor,Fan
References: Stream (Java SE 17 & JDK 17) - Oracle, Collectors (Java SE 17 & JDK 17) - Oracle

NEW QUESTION 4
Which two code fragments compile?
A)

```
class L6 {
    public static void main(String[] args) {
        var x = new ArrayList<>();
        x.add(10);
        x.add("30");
        System.out.println(x);
    }
}
```

B)

```
class L2 {
    public void m(int x) {
        var x = 10;
    }
}
```

C)

```
class A {}
class B extends A{}
class L4 {
    public static void main(String[] args) {
        var x = new A();
        x = new B();
    }
}
```

D)

```
class L3 {
    public static void main(String[] args) {
        var a = 10;
        a = "30";
    }
}
```

E)

```
class L5 {
    public void m() {
        var strVar = null;
    }
}
```

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

Answer: BE

Explanation:
The two code fragments that compile are B and E. These are the only ones that use the correct syntax for declaring and initializing a var variable. The var keyword is a reserved type name that allows the compiler to infer the type of the variable based on the initializer expression. However, the var variable must have an initializer, and the initializer must not be null or a lambda expression. Therefore, option A is invalid because it does not have an initializer, option C is invalid because it has a null initializer, and option D is invalid because it has a lambda expression as an initializer. Option B is valid because it has a String initializer, and option E is valid because it has an int initializer. https://docs.oracle.com/en/java/javase/17/language/local-variable-type-inference.html

NEW QUESTION 5
Given:

```
public class Test {
   public static void main(String[] args) {
       List<String>  elements =
             Arrays.asList("car", "truck", "car",
                           "bicycle", "car", "truck", "motorcycle");
       Map<String, Long>  outcome =
             elements.stream().collect(Collectors.groupingBy(Function.identity(), Collectors.counting() ) );
       System.out.println(outcome);
   }
}
```

What is the result?

A. Bicycle =7, car=7, motorcycle=7, truck=7)
B. (3:bicycle, 0:car, 0motercycle, 5:truck)
C. (Bicycle, car, motorcycle, truck)
D. Bicycle-1, car=3, motorcycle=1, truck=2)
E. Compilation fails.

Answer: E

Explanation:
The answer is E because the code fragment contains several syntax errors that prevent it from compiling. Some of the errors are:
? The enum declaration is missing a semicolon after the list of constants.
? The enum constants are not capitalized, which violates the Java naming convention for enums.
? The switch expression is missing parentheses around the variable name.
? The case labels are missing colons after the enum constants.
? The default label is missing a break statement, which causes a fall-through to the next case.
? The println statement is missing a closing parenthesis and a semicolon. A possible corrected version of the code fragment is:
enum Vehicle { BICYCLE, CAR, MOTORCYCLE, TRUCK; } public class Test { public static void main(String[] args) { Vehicle v = Vehicle.BICYCLE; switch (v) {

case BICYCLE:
System.out.print(??1??); break; case CAR: System.out.print(??3??); break; case MOTORCYCLE: System.out.print(??1??); break; case TRUCK: System.out.print(??2??); break; default: System.out.print(??0??); break; } System.out.println(); } }
This would print 1 as the output. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Enum Types
? The switch Statement

NEW QUESTION 6
Given the product class:

```
import java.io.*;
public class Product implements Serializable {
  private static float averagePrice = 2.99f;
  private String description;
  private transient float price;
  public Product(String description, float price) {
    this.description = description;
    this.price = price;
  }
  public void readObject(ObjectInputStream in)
      throws IOException, ClassNotFoundException {
    in.defaultReadObject();
    price = averagePrice;
  }
  public String toString() {
    return description+" "+price+" "+averagePrice;
  }
}
```

And the shop class:

```
import java.io.*;
public class Shop {
  public static void main(String[] args) {
    Product p = new Product("Cookie", 3.99f);
    try {
      try (ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream("p.ser"))) {
        out.writeObject(p);
      }
      try (ObjectInputStream in = new ObjectInputStream(new FileInputStream("p.ser"))) {
        p = (Product)in.readObject();
      }
    } catch (Exception e) { e.printStackTrace(); }
    System.out.println(p);
  }
}
```

What is the result?

A. Cookie 2.99 2.99
B. Cookie 3.99 2.99
C. Cookie 0.0 0.0
D. An exception is produced at runtime
E. Compilation fails
F. Cookie 0.0 2.99

Answer: E

Explanation:
The code fragment will fail to compile because the readObject method in the Product class is missing the @Override annotation. The readObject method is a special method that is used to customize the deserialization process of an object. It must be declared as private, have no return type, and take a single parameter of type ObjectInputStream. It must also be annotated with @Override to indicate that it overrides the default behavior of the ObjectInputStream class. Without the @Override annotation, the compiler will treat the readObject method as a normal method and not as a deserialization hook. Therefore, the code fragment will produce a compilation error. References: Object Serialization - Oracle, [ObjectInputStream (Java SE 17 & JDK 17) - Oracle]

NEW QUESTION 7
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 8
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 9
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 10
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 10
Given the code fragment:

```
abstract sealed interface SInt permits Story, Art {
default String getTitle() { return "Book Title" ; }
}
```

abstract sealed interface SInt permits Story, Art { default String getTitle() { return "Book Title" ; }
}
Which set of class definitions compiles?

A. Interace story extends STnt {} Interface Art extends SInt {}
B. Public interface story extends sInd {} Public interface Art extends SInt {}
C. Sealed interface Storty extends sInt {} Non-sealed class Art implements Sint {}
D. Non-sealed interface story extends SInt {} Class Art implements Sint {}
E. Non-sealed interface story extends SInt {} Non-sealed interaface Art extends Sint {}

Answer: C

Explanation:
The answer is C because the code fragment given is an abstract sealed interface SInt that permits Story and Art. The correct answer is option C, which is a sealed interface Story that extends SInt and a non-sealed class Art that implements SInt. This is because a sealed interface can only be extended by the classes or interfaces that it permits, and a non-sealed class can implement a sealed interface.
Option A is incorrect because interface is misspelled as interace, and Story and Art should be capitalized as they are the names of the permitted classes or interfaces.
Option B is incorrect because public is misspelled as public, and sInd should be SInt as it is the name of the sealed interface.
Option D is incorrect because a non-sealed interface cannot extend a sealed interface, as it would violate the restriction of permitted subtypes.
Option E is incorrect because both Story and Art cannot be non-sealed interfaces, as they
would also violate the restriction of permitted subtypes.
References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Sealed Classes and Interfaces in Java 15 | Baeldung
? Sealed Class in Java - Javatpoint

NEW QUESTION 13
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 18
Given the code fragment:

```
// line n1
String input = console.readLine("Input a number: ");
int number = Integer.parseInt(input);

if (number % 2 == 0) {
    System.out.println(number + " is even.");
} else {
    System.out.println(number + " is odd");
}
```

Which code line n1, obtains the java.io.Console object?
A)

```
Console console = Syst      .em.cons      ole(Syst      em      .in);
```

B)

```
Console console = Cons      ole.get      Intance(      );
```

C)

```
Console console = Syst      .em.cons      ole();
```

D)

```
Console console = new Console(System.in);
```

E)

```
Console console = new Console(new InputStreamReader(System.in));
```

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

Answer: A

Explanation:
The code fragment is trying to obtain the java.io.Console object, which is a class that provides methods to access the character-based console device, if any, associated with the current Java virtual machine. The correct way to obtain the Console object is to call the static method Console console() in the java.lang.System class. This method returns the unique Console object associated with the current Java virtual machine, if any. Therefore, option A is correct, as it calls System.console() and assigns it to a Console variable. References:
? https://docs.oracle.com/javase/17/docs/api/java.base/java/io/Console.html
? https://docs.oracle.com/javase/17/docs/api/java.base/java/lang/System.html#cons ole()
? https://education.oracle.com/products/trackp_OCPJSE17
? https://mylearn.oracle.com/ou/learning-path/java-se-17-developer/99487

NEW QUESTION 22
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 25
Given the code fragments:

```
class Car implements Serializable {
    private static long serialVersionUID = 454L;
    String name;
    public Car(String name) { this.name = name; }
}

class LuxuryCar extends Car {          // line n1
    int flag_HHC;
    public LuxuryCar(String name, int flag_HHC) {
        super(name);
        this.flag_HHC = flag_HHC;
    }
    public String toString() {
        return name + " : " + flag_HHC;
    }
}
```

and:

```
public static void main(String[] args) {    // line n2
    Car b = new LuxuryCar("Wagon", 200);
    try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("car.ser"));
         ObjectInputStream ois = new ObjectInputStream(new FileInputStream("car.ser"));) {
             oos.writeObject(b);
             System.out.println((Car)(ois.readObject());          // line n3
    }
}
```

Which action prints Wagon : 200?

A. At line n1, implement the java.io, Serializable interface.
B. At line n3, replace readObject (0 with readLine().
C. At Line n3, replace Car with LuxurayCar.
D. At Line n1, implement the java.io.AutoCloseable interface
E. At line n2, in the main method signature, add throws IOException, ClassCastException.
F. At line n2, in the main method signature, add throws IoException, ClassNotFoundException.

Answer: F

Explanation:
The code fragment is trying to read an object from a file using the ObjectInputStream class. This class throws an IOException and a ClassNotFoundException. To handle these exceptions, the main method signature should declare that it throws these exceptions. Otherwise, the code will not compile. If the main method throws these exceptions, the code will print Wagon : 200, which is the result of calling the toString method of the LuxuryCar object that was written to the file. References: ObjectInputStream (Java SE 17 & JDK 17) - Oracle, ObjectOutputStream (Java SE 17 & JDK 17) - Oracle

NEW QUESTION 30
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 32
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 34
Given the code fragment:

```
Integer rank = 4;
switch (rank) {
     case 1,4 -> System.out.println("Range1");
     case 5,8 -> System.out.println("Range2");
     case 9,10 -> System.out.println("Range3");
     default -> System.out.println("Not a valid rank.");
}
```

What is the result?

A. Range 1Range 2Range 3
B. Range1Note a valid rank.
C. Range 1Range 2Range 3Range 1Not a valida rank
D. Range 1

Answer: C

Explanation:
The code fragment is using the switch statement with the new Java 17 syntax. The switch statement checks the value of the variable rank and executes the corresponding case statement. In this case, the value of rank is 4, so the first case statement is executed, printing ??Range1??. The second and third case statements are also executed, printing ??Range2?? and ??Range3??. The default case statement is also executed, printing ??Not a valid rank??. References: Java Language Changes - Oracle Help Center

NEW QUESTION 37
Assume you have an automatic module from the module path display-ascii-0.2. jar. Which name is given to the automatic module based on the given JAR file?

A. Display.ascii
B. Display-ascii-0.2
C. Display-ascii
D. Display-ascii-0

Answer: C

Explanation:
An automatic module name is derived from the name of the JAR file when it does not contain a module-info.class file. If the JAR file has an ??Automatic-Module-Name?? attribute in its main manifest, then its value is the module name. Otherwise, the module name is derived from the JAR file??s name by removing any version numbers and converting it to lower case. Therefore, for a JAR named display-ascii-0.2.jar, the automatic module name would be display-ascii, following these rules.

NEW QUESTION 41
Given:

```
public class Test {
   public String attach1(List<String> data) {
       return data.parallelStream().reduce("w", (n,m) -> n+m, String::concat);
   }
   public String attach2(List<String> data) {
      return data.parallelStream().reduce((1, p)-> 1+p).get();
   }

   public static void main(String[] args) {
      Test t = new Test();
      var list = List.of("Table", "Chair");
      String x= t.attach1(list);
      String y= t.attach2(list);
     System.out.print(x+ " "+y);
    }
}
```

What is the result?

A. Tablechair Tablechair
B. Wtablechair tableChair
C. A RuntimeException is thrown
D. wTableChair TableChair
E. Compilation fails

Answer: E

Explanation:
The code fragment will fail to compile because the class name and the constructor name do not match. The class name is Furniture, but the constructor name is Wtable. This will cause a syntax error. The correct way to define a constructor is to use the same name as the class name. Therefore, the code fragment should change the constructor name to Furniture or change the class name to Wtable.

NEW QUESTION 44
.....

Relate Links

100% Pass Your 1z0-829 Exam with Exambible Prep Materials

https://www.exambible.com/1z0-829-exam/

Contact us

We are proud of our high-quality customer service, which serves you around the clock 24/7.

Viste - https://www.exambible.com/