Exam Questions 1z0-829

Java SE 17 Developer

https://www.2passeasy.com/dumps/1z0-829/

NEW QUESTION 1
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 2
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 3
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 4
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 5
Given the code fragment:

```
List<Integer> listOfNumbers = List.of(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
```

Which code fragment returns different values?

A. int sum = listOfNumber
B. parallelStream () reduce (5, Integer:: sum) ;
C. int sum = listOfNumber
D. Stream () reduce (5, (a, b) -> a + b) ;
E. int sum = listOfNumber
F. Stream () reduce (Integer:: sum) ; +5;
G. int sum = listOfNumber
H. parallelStream () reduce ({m, n) -> m +n) orElse (5) +5;
I. int sum = listOfNumber
J. Stream () reduce (0, Integer:: sum) + 5

Answer: C

Explanation:
The answer is C because the code fragment uses a different syntax and logic for the reduce operation than the other options. The reduce method in option C takes a single parameter, which is a BinaryOperator that combines two elements of the stream into one. The method returns an Optional, which may or may not contain a value depending on whether the stream is empty or not. The code fragment then adds 5 to the result of the reduce method, regardless of whether it is present or not. This may cause an exception if the Optional is empty, or produce a different value than the other options if the Optional is not empty.
The other options use a different syntax and logic for the reduce operation. They all take two parameters, which are an identity value and a BinaryOperator that combines an element of the stream with an accumulator. The method returns the final accumulator value, which is equal to the identity value if the stream is empty, or the result of applying the BinaryOperator to all elements of the stream otherwise. The code fragments then add 5 to the result of the reduce method, which will always produce a valid value.
For example, suppose listOfNumbers contains [1, 2, 3]. Then, option A will perform the following steps:
? Initialize accumulator to identity value 5
? Apply BinaryOperator Integer::sum to accumulator and first element: 5 + 1 = 6
? Update accumulator to 6
? Apply BinaryOperator Integer::sum to accumulator and second element: 6 + 2 = 8
? Update accumulator to 8
? Apply BinaryOperator Integer::sum to accumulator and third element: 8 + 3 = 11
? Update accumulator to 11
? Return final accumulator value 11
? Add 5 to final accumulator value: 11 + 5 = 16
Option B will perform the same steps as option A, except using a lambda expression instead of a method reference for the BinaryOperator. Option D will perform the same steps as option A, except using parallelStream instead of stream, which may change the order of applying the BinaryOperator but not the final result.
Option E will perform the same steps as option A, except using identity value 0 instead of 5.
Option C, however, will perform the following steps:
? Apply BinaryOperator Integer::sum to first and second element: 1 + 2 = 3
? Apply BinaryOperator Integer::sum to previous result and third element: 3 + 3 = 6
? Return Optional containing final result value 6
? Add 5 to Optional value: Optional.of(6) + 5 = Optional.of(11)

As you can see, option C produces a different value than the other options, and also uses a different syntax and logic for the reduce operation. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Guide to Stream.reduce()

NEW QUESTION 6
Given the code fragment:

```
int a = 2;
int b = ~a;
int c = a^b;
boolean d = a < b & a > c++;
System.out.println(d + " " + c);
boolean e = a > b && a > c++;
System.out.println(e + " " + c);
```

What is the result?

A. false 1false 2
B. true 1false 2
C. false 1ture 2
D. falase 0true 1

Answer: B

Explanation:
The code fragment is comparing the values of a, b, and c using the < and > operators. The first comparison, d, is checking if a is less than b and greater than c. Since a is equal to 2, b is equal to -2, and c is equal to -4, this comparison will evaluate to true. The second comparison, e, is checking if a is greater than b and a is greater than c. Since a is equal to 2, b is equal to -2, and c is equal to -4, this comparison will evaluate to false.
Therefore, the result will be true 1 false 2. References: Operators (The Java™ Tutorials > Learning the Java Language - Oracle

NEW QUESTION 7
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 8
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      ole(Syst    .in);
                   em.cons          em
```

B)

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

C)

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

D)

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

E)

```
Console console = new       (new Inp
                  Console          utStreamReader(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 9
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 10
Given:

```
class StockException extends Exception {
    public StockException(String s) { super(s); }
}
class OutofStockException extends StockException {
    public OutofStockException(String s) { super(s); }
}
```

and the code fragment:

```
public class Test {
    public static void main(String[] args) throws OutofStockException {
        m();
    }
    public static void m() throws OutofStockException {
        try {
            throw new StockException("Raised.");
        } catch (Exception e) {
            throw new OutofStockException(e.getMessage());
        }
    }
}
```

Which statement is true?

A. The program throws StockException.
B. The program fails to compile.
C. The program throws outofStockException.
D. The program throws ClassCastException

Answer: B

Explanation:
The answer is B because the code fragment contains a syntax error that prevents it from compiling. The code fragment tries to catch a StockException in line 10, but the catch block does not have a parameter of type StockException. The catch block should have a parameter of type StockException, such as:

catch (StockException e) { // handle the exception }
This is required by the Java syntax for the catch clause, which must have a parameter that is a subclass of Throwable. Without a parameter, the catch block is invalid and causes a compilation error.
Option A is incorrect because the program does not throw a StockException, as it does not compile.
Option C is incorrect because the program does not throw an OutofStockException, as it does not compile.
Option D is incorrect because the program does not throw a ClassCastException, as it does not compile. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? The try-with-resources Statement (The Java™ Tutorials > Essential Classes > Exceptions)
? The catch Blocks (The Java™ Tutorials > Essential Classes > Exceptions)

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 12

Given the code fragment:

```
record Product(int pNumber, String pName) {
    int regNo = 100;
    public int getRegNumber() {
        return regNo;
    }
}

public class App {
    public static void main(String[] args) {
        Product p1 = new Product (1111, "Ink Bottle");
    }
}
```

Which action enables the code to compile?

A. Replace record with void.
B. Remove the regNO initialization statement.
C. Make the regNo variable static.
D. Replace thye regNo variable static
E. Make the regNo variable public

Answer: E

Explanation:
The code will compile if the regNo variable is made public. This is because the regNo variable is being accessed in the main method of the App class, which is outside the scope of the Product class. Making the regNo variable public will allow it to be accessed from outside the class. 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/tutorial/java/javaOO/accesscontrol.html

NEW QUESTION 14
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
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 19
Given the code fragments:

```
class Test {
    volatile int x = 1;
    AtomicInteger xObj = new AtomicInteger(1);
}
```

and

```
public static void main(String[] args) {
    Test t = new Test();
    Runnable r1 = () -> {
        Thread trd = Thread.currentThread();
        while (t.x < 3 ) {
            System.out.print(trd.getName()+" : "+t.x+" : ");
            t.x++;
        }
    };
    Runnable r2 = () -> {
        Thread trd = Thread.currentThread();
        while (t.xObj.get() < 3) {
            System.out.print(trd.getName()+" : "+t.xObj.get()+" : ");
            t.xObj.getAndIncrement();
        }
    };
    Thread t1 = new Thread(r1,"t1");
    Thread t2 = new Thread(r2,"t2");
    t1.start();
    t2.start();
}
```

Which is true?

A. The program prints t1 : 1: t2 : 1: t1 : t2 : 2 : in random order.
B. The program prints t1 : 1 : t2: 1 : t1 : 2 : t2: 2:

C. The program prints t1 : 1: t2 : 1: t1 : 1 : t2 : 1 : indefinitely
D. The program prints an exception

Answer: B

Explanation:
The code creates two threads, t1 and t2, and starts them. The threads will print their names and the value of the Atomic Integer object, x, which is initially set to 1. The threads will then increment the value of x and print their names and the new value of x. Since the threads are started at the same time, the output will be in random order.
However, the final output will always be t1 : 1 : t2: 1 : t1 : 2 : t2: 2: References: AtomicInteger (Java SE 17 & JDK 17) - Oracle

NEW QUESTION 21
.....

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual 1z0-829 Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the 1z0-829 Product From:

https://www.2passeasy.com/dumps/1z0-829/

Money Back Guarantee

1z0-829 Practice Exam Features:

* 1z0-829 Questions and Answers Updated Frequently

* 1z0-829 Practice Questions Verified by Expert Senior Certified Staff

* 1z0-829 Most Realistic Questions that Guarantee you a Pass on Your FirstTry

* 1z0-829 Practice Test Questions in Multiple Choice Formats and Updatesfor 1 Year