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:

```java
import java.io.Serializable;
public class Software implements Serializable {
    private String title;
    public Software(String title) {
        this.title = title;
        System.out.print("Software ");
    }
    public String toString() { return title; }
}

public class Game extends Software {
    private int players;
    public Game(String title, int players) {
        super(title);
        this.players = players;
        System.out.print("Game ");
    }
    public String toString() { return super.toString()+" "+players; }

}

import java.io.*;
public class AppStore {
    public static void main(String[] args) {
        Software s = new Game("Chess", 2);
        try(ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream("game.ser"))) {
            out.writeObject(s);
        } catch (Exception e) {
            System.out.println("write error");
        }
        try(ObjectInputStream in = new ObjectInputStream(new FileInputStream("game.ser"))) {
            s = (Software)in.readObject();
        } catch (Exception e) {
            System.out.println("read error");
        }
        System.out.println(s);
    }
}
```

What is the result?

A. Software Game Chess 0
B. Software Game Software Game Chese 2
C. Software game write error
D. Software Game Software Game chess 0
E. Software Game Chess 2
F. Software Game read error

Answer: B

Explanation:
 The answer is B because the code uses the writeObject and readObject methods of
the ObjectOutputStream and ObjectInputStream classes to serialize and deserialize the Game object. These methods use the default serialization mechanism,
which writes and reads the state of the object??s fields, including the inherited ones. Therefore, the title field of the Software class is also serialized and
deserialized along with the players field of the Game class. The toString method of the Game class calls the toString method of the Software class using
super.toString(), which returns the value of the title field. Hence, when the deserialized object is printed, it shows ??Software Game Software Game Chess 2??.
References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Serialization and Deserialization in Java with Example

NEW QUESTION 3
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 4
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 5
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 6
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 7
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 8
Given:

```java
final class Folder {    // line n1
    // line n2
    public void open(){
        System.out.print("Open ");
    }
}

public class Test {
    public static void main(String[] args) throws Exception {
        try (Folder f = new Folder()) {
            f.open();
        }
    }
}
```

Which two modifications enable the code to print Open Close?
A)

At line n2, insert:

```java
final void close() (
    System.out.print("Close ");
```

B)

Replace line n1 with:

```java
class Folder extends Closeable (
```

C)

Replace line n1 with:

```java
class Folder extends Exception (
```

D)

Replace `line n1` with:

```
class Folder implements AutoCloseable {
```

E)

At `line n2`, insert:

```
public void close() throws IOException {
    System.out.print("Close ");
}
```

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

Answer: BE

Explanation:
 The code given is a try-with-resources statement that declares a resource of type AutoCloseable. The resource is an anonymous class that implements the AutoCloseable interface and overrides the close() method. The code also has a print() method that prints the value of the variable s. The code is supposed to print ??Open Close??, but it does not compile because of two errors.
The first error is at line n1, where the anonymous class is missing a semicolon at the end of its declaration. This causes a syntax error and prevents the code from compiling. To fix this error, option B adds a semicolon after the closing curly brace of the anonymous class.
The second error is at line n2, where the print() method is called without an object reference. This causes a compilation error because the print() method is not static and cannot be invoked without an object. To fix this error, option E adds an object reference to the print() method by using the variable t.
Therefore, options B and E are correct and enable the code to print ??Open Close??.

NEW QUESTION 9
Given:

```
1. class Item {
2.     String name;
3.     public static void display() {
4.         name = "Vase";
5.         System.out.println(name);
6.     }
7.     public void display(String design) {
8.         this.name += name;
9.         System.out.println(name);
10.     }
11. }
12. public class App {
13.     public static void main(String[] args) {
14.         Item i1 = new Item();
15.         i1.display("Flower");
16.     }
17. }
```

Which action enables the code to compile?

A. Replace 15 with item.display ("Flower");
B. Replace 2 with static string name;
C. Replace 7 with public void display (string design) {
D. Replace 3 with private static void display () {

Answer: C

Explanation:
 The answer is C because the code fragment contains a syntax error in line 7, where the method display is declared without any parameter type. This causes a compilation error, as Java requires the parameter type to be specified for each method parameter. To fix this error, the parameter type should be added before the parameter name, such as string design. This will enable the code to compile and run without any errors. References:
? Oracle Certified Professional: Java SE 17 Developer
? Java SE 17 Developer
? OCP Oracle Certified Professional Java SE 17 Developer Study Guide
? Java Methods

NEW QUESTION 10
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 10
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 11
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 13
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 14
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 15
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((l, p)-> l+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 16
Given the code fragment:

```
List<String> specialDays = List.of("NewYear","Valentines","Spring","Labour");
System.out.print(specialDays.stream().allMatch(s ->s.equals("Labour")));
System.out.print(" " + specialDays.stream().anyMatch(s ->s.equals("Labour")));
System.out.print(" " + specialDays.stream().noneMatch(s -> s.equals("Halloween")));
System.out.print(" " +specialDays.stream().findFirst());
```

What is the result?

A. False true true optional (Newyear)
B. 0110
C. True true false NewYear
D. 010 optional (Newyear)

Answer: A

Explanation:
 The code fragment is using the stream methods allMatch, anyMatch, noneMatch, and findFirst on a list of strings called specialDays. These methods are used to perform matching operations on the elements of a stream, such as checking if all, any, or none of the elements satisfy a given predicate, or finding the first element that matches a predicate1. The predicate in this case is that the string equals ??Labour?? or ??Halloween??. The output will be:
? False: because not all of the elements in specialDays are equal to ??Labour?? or ??Halloween??.
? true: because at least one of the elements in specialDays is equal to ??Labour?? or ??Halloween??.
? true: because none of the elements in specialDays are equal to both ??Labour?? and ??Halloween??.
? Optional[NewYear]: because the first element in specialDays that matches the predicate is ??NewYear??, and the findFirst method returns an Optional object that may or may not contain a non-null value2.
References: Stream (Java SE 17 & JDK 17), Optional (Java SE 17 & JDK 17)

NEW QUESTION 20
......

Thank You for Trying Our Product

We offer two products:

1st - We have Practice Tests Software with Actual Exam Questions

2nd - Questons and Answers in PDF Format

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

100% Actual & Verified — Instant Download, Please Click
Order The 1z0-829 Practice Test Here