



Oracle

Exam Questions 1z0-829

Java SE 17 Developer

NEW QUESTION 1

Given:

```
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 Chess 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 2

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 InterruptedException.

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 3

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 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

Given:

```
public class Test {  
    static interface Animal {  
    }  
  
    static class Dog implements Animal {  
    }  
  
    private static void play(Animal a) {  
        System.out.print("flips");  
    }  
  
    private static void play(Dog d) {  
        System.out.print("runs");  
    }  
  
    public static void main(String[] args) {  
        Animal a1 = new Dog();  
        Dog a2 = new Dog();  
        play(a1);  
        play(a2);  
    }  
}
```

What is the result?

- A. flipsflips
- B. Compilation fails
- C. flipsruns
- D. runsflips
- E. runsruns

Answer: B

Explanation:

The code fragment will fail to compile because the play method in the Dog class is declared as private, which means that it cannot be accessed from outside the class. The main method is trying to call the play method on a Dog object, which is not allowed. Therefore, the code fragment will produce a compilation error.

NEW QUESTION 7

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 8

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 9

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. Interface story extends STnt {} Interface Art extends SInt {}
- B. Public interface story extends sInt {} Public interface Art extends SInt {}
- C. Sealed interface Story 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 `SIInt` that permits `Story` and `Art`. The correct answer is option C, which is a sealed interface `Story` that extends `SIInt` and a non-sealed class `Art` that implements `SIInt`. 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 `publc`, and `sInt` should be `SIInt` 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 10

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

B)

```
Console console = Console.getInstance();
```

C)

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

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#console\(\)](https://docs.oracle.com/javase/17/docs/api/java.base/java/lang/System.html#console())
- ? https://education.oracle.com/products/trackp_OCPJSE17
- ? <https://mylearn.oracle.com/ou/learning-path/java-se-17-developer/99487>

NEW QUESTION 10

Given the course table:

COURSE_ID	COURSE_NAME	COURSE_FEE	COURSE_LEVEL
1021	Java Programmer	400.00	1
1022	Java Architect	600.00	2
1023	Java Master	600.00	2

Given the code fragment:

```
try (Connection con = DriverManager.getConnection(connectionString)) {
    Statement statement = con.createStatement(TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
    String qry = "UPDATE course SET course_fee = ? where COURSE_LEVEL = ?";
    PreparedStatement prStmt = con.prepareStatement(qry, TYPE_SCROLL_INSENSITIVE);
    prStmt.setDouble(1,600.00);
    prStmt.setInt(2,2);
    System.out.println(prStmt.executeUpdate());
}
catch(SQLException sqlException) {
    System.out.println(sqlException);
}
```

- A. 2
- B. false
- C. true
- D. 1

Answer: C

Explanation:

The code fragment will execute the update statement and set the course fee of the course with ID 1021 to 5000. The `executeUpdate` method returns an `int` value that indicates the number of rows affected by the SQL statement. In this case, only one row will be updated, so the result variable will be 1. The `if` statement will check if the result is greater than 0, which is true, and print `??Updated successfully??`. Therefore, the output of the code fragment is true. References:

- https://education.oracle.com/products/trackp_OCPJSE17, <https://mylearn.oracle.com/ou/learning-path/java-se-17-developer/99487>,
- [https://docs.oracle.com/en/java/javase/17/docs/api/java.sql/java/sql/Statement.html#executeUpdate\(java.lang.String\)](https://docs.oracle.com/en/java/javase/17/docs/api/java.sql/java/sql/Statement.html#executeUpdate(java.lang.String))

NEW QUESTION 12

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 14

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 16

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 20

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 23

.....

Thank You for Trying Our Product

We offer two products:

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

2nd - Questions 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](#)