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 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 2
Which statement is true?

A. The tryLock () method returns a boolean indicator immediately regardless if it has or has not managed to acquire the lock.
B. The tryLock () method returns a boolean indicator immediately if it has managed to acquire the lock, otherwise it waits for the lock acquisition.
C. The lock () method returns a boolean indicator immediately if it has managed to acquire the lock, otherwise it waits for the lock acquisition.

D. The Lock () method returns a boolean indicator immediately regardless if it has or has not managed to acquire the lock

Answer: A

Explanation:
The tryLock () method of the Lock interface is a non-blocking attempt to acquire a lock. It returns true if the lock is available and acquired by the current thread, and false otherwise. It does not wait for the lock to be released by another thread. This is different from the lock () method, which blocks the current thread until the lock is acquired, and does not return any value. References: https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/util/concurrent/locks/Lock.html#tryLock(), 3, 4, 5

NEW QUESTION 3
Given the code fragment:

```
List lst = new ArrayList();
lst.add("e1");
lst.add("e3");
lst.add("e2");

int x1 = Collections.binarySearch(lst, "e3");
System.out.println(x1);
Collections.sort(lst);
int x2 = Collections.binarySearch(lst, "e3");
System.out.println(x2);

Collections.reverse(lst);
int x3 = Collections.binarySearch(lst, "e3");
System.out.println(x3);
```

What is the result?

A. 2
B. -2
C. 22E.111F.12-4

Answer: B

Explanation:
The code fragment uses the Collections.binarySearch method to search for the string ??e3?? in the list. The first search returns the index of the element, which is 2. The second search returns the index of the element, which is 0. The third search returns the index of the element, which is -4. The final result is 2. References: Collections (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
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 6
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 7
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 8
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 9
Given the code fragment:

```
String a = "Hello! Java";
System.out.print(a.indexOf("Java"));
a.replace("Hello!", "Welcome!");
System.out.print(a.indexOf("Java"));
StringBuilder b = new StringBuilder(a);
System.out.print(b.indexOf("Java"));
```

What is the result?

A. 81111
B. 8109
C. 777
D. 71010
E. 888
F. 7107

Answer: B

Explanation:
The code fragment is creating a string variable ??a?? with the value ??Hello! Java??. Then, it is printing the index of ??Java?? in ??a??. Next, it is replacing ??Hello!?? with ??Welcome!?? in ??a??. Then, it is printing the index of ??Java?? in ??a??. Finally, it is creating a new StringBuilder object ??b?? with the value of ??a?? and printing the index of ??Java?? in ??b??. The output will be 8109 because the index of ??Java?? in ??a?? is 8, the index of ??Java?? in ??a?? after replacing ??Hello!?? with ??Welcome!?? is 10, and the index of ??Java?? in ??b?? is 9. References: Oracle Java SE 17 Developer source and documents: [String (Java SE 17 & JDK 17)], [StringBuilder (Java SE 17 & JDK 17)]

NEW QUESTION 10
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 12
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 16
Given the code fragment:

```
Pet p = new Pet("Dog");
Pet p1 = p;
p1.name = "Cat";
p = p1;
System.out.println(p.name);
p = null;
System.out.println(p1.name);
```

What is the result?

A. A.Cat Dog
B. A NullPointerException is thrown CatCat
C. Dog Dog
D. Cat null

Answer: D

Explanation:
The answer is E because the code fragment creates a new Pet object with the name ??Dog?? and assigns it to the variable p. Then, it assigns p to p1. Next, it changes the name of p1 to ??Cat??. Then, it assigns p1 to p. Finally, it sets p to null and prints the name of p and p1. The output will be ??Cat?? and ??null?? because p is set to null and p1 still points to the Pet object with the name ??Cat??.

NEW QUESTION 20
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 il = new Item();
15.       il.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 25
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.      in);
                     Console
```

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 28
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 33
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 34
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 37
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 38
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 43
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());
```

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 45
Given:

```
public class Weather {
    public enum Forecast {
        SUNNY, CLOUDY, RAINY;
        @Override
        public String toString() { return "SNOWY";}
    }

    public static void main(String[] args) {
        System.out.print(Forecast.SUNNY.ordinal() + " ");
        System.out.print(Forecast.valueOf("cloudy".toUpperCase()));
    }
}
```

What is the result?

A. 1 RAINY
B. Compilation fails
C. 1 Snowy
D. 0 CLOUDY
E. 0 Snowy

Answer: E

Explanation:
The code is defining an enum class called Forecast with three values: SUNNY, CLOUDY, and RAINY. The toString() method is overridden to always return ??SNOWY??. In the main method, the ordinal value of SUNNY is printed, which is 0, followed by the value of CLOUDY converted to uppercase, which is

??CLOUDY??. However, since the
toString() method of Forecast returns ??SNOWY?? regardless of the actual value, the output
will be ??0 SNOWY??. References: Enum (Java SE 17 & JDK 17), Enum.EnumDesc (Java SE 17 & JDK 17)

NEW QUESTION 48
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 53
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 57
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 59
.....

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