Preguntas tipo examen Programador Java 11 Certificado 1Z0-819

Pregunta 1

Consider the following code appearing in a file named TestClass.java:

```
class Test{ }  // 1
public class TestClass {
  var v1; // 2
  public int main(String[] args)  { // 3
     var v2; //4
     double x=10, double y;  // 5
     System.out.println[]; // 6
     for(var k=0; k<x; k++){ } //7
     return 0;
  }
}
```

Which of the lines are valid?(choose 3)

}

A. //1.
B. //2.
C. //3.
D. //4.
E. //5.
F. //6.
G. //7

Explicación: La línea 1 es correcta porque se puede definir otra clase en un fichero Java, además de la pública, siempre que no lleve el modificador public. La línea 2 es incorrecta porque no se puede aplicar inferencia de tipos con variables atributo. La línea 3 contiene una definición correcta de método en Java. La línea 4 no es correcta porque si se aplica inferencia de tipos en variables locales es obligatorio asignar un valor inicial a dicha variable. La línea 5 es incorrecta porque sobraría la indicación del tipo double en la segunda variable. La línea 6 es incorrecta porque la llamada al método se debe realizar con paréntesis, no con corchetes. La línea 7 es correcta, se puede aplicar inferencia de tipos en la variable de control de un for

Pregunta 2

What will the following code print ?

```
class OperTest{
  public static void main(String[] args){
    int a = 5;
    int b = 6;
    a += (a = 3);
    b = b + (b =2);
    System.out.println(a+ ",  "+b);
  }
}
```

A. 3, 4
B. 0, 0
C. 5, 6
D. 8, 8

Explicación. A la hora de evaluar las expresiones, primero se procesa la operación entre paréntesis, sin embargo, se emplea el valor actual de la variable en el resto de la expresión. Es decir, por ejemplo, en a+=(a=3), se procesa primero (a=3), pero a la hora de sumarlo a la variable a, se emplea el valor actual de ésta, es decir, 5: a=5+3;

Pregunta 3

Which of the following declarations are valid? (choose 4)

A. float f1 = 1.0;
B. float f = 43e1;
C. float f = -1;
D. float f = 0x0123;
E. float f=10;
F. var fx=1.0;float fy=fx;
G. var f=6f;

Explicación: Los literales de tipo decimal son considerados double y no se pueden asignar directamente a una variable de tipo float. Ese es el motivo por el que las instrucciones A, B y F son incorrectas. En C, D y E se asigna un int a un float, produciendose una conversión implícita de tipos. En la instrucción G, la letra f a continuación del 6 hace que sea tratado como float.

Pregunta 4

Which of the following statements will evaluate to true?

A. "String".replace('g','G') == "String".replace('g','G')
B. "String".replace('g','g') == new String("String").replace('g','g')
C. "String".replace('g','G')=="StrinG"
D. "String".replace('g','g')=="String"
E. None of these.

Explicación: En A, ambas instrucciones generan una nueva cadena resultante de la sustitución. Como son objetos diferentes, a pesar de contener el mismo texto, la comparación con == resulta falsa. En B, la instrucción de la izquierda no genera un nuevo objeto, ya que la cadena resultante es la misma y se reutiliza la existente, pero la instrucción de la derecha si genera un nuevo objeto, por lo que la comparación es falsa. En C, la instrucción de izquierda genera un nuevo objeto , como el literal corresponde a otro objeto, a pesar de ser la misma cadena la comparación también resulta falsa. En D, la llamada a replace no genera un nuevo objeto, se reutiliza "String" y como la expresión de la derecha es un literal que coincide con éste, estamos ante el mismo objeto, por lo que la comparación resulta verdadera

Pregunta 5

Consider the following code:

```
class Outsider
{
  public class Insider{ }
}
public class TestClass
{
  public static void main(String[] args)
  {
     var os = new Outsider();
     // 1 insert line here
   }
}
```

Which of the following options can be inserted at //1?

A. Insider in = os.new Insider();
B. Outsider.Insider in = os.new Insider();
C. Insider in = Outsider.new Insider();
D. os.Insider in = os.new Insider();

Explicación: Para hacer referencia a una clase interna de una clase desde otra exterior, es necesario indicar ClaseExterna.ClaseInterna o haberla importado

Pregunta 6

Assuming that the following code compiles without any error, identify correct statements (choose 2).

```
interface Processor {

  A process(String str);

}

class ItemProcessor implements Processor{

  @Override

  public B process(String str){

     return new B(str);

  }

}
```

A. B must be a sub type of A.
B. A must be a sub type of B.
C. B must be final.
D. A cannot be abstract.
E. B cannot be abstract

Explicación. Al cambiar el tipo de devolución del método durante la sobrescritura, según las reglas de la misma, ello solo es posible si el tipo de devolución del nuevo método es un subtipo del original. Además, dado que en la sobrescritura del método process de ItemProcessor se crea una instancia de B, esta clase no podrá ser abstracta

Pregunta 7

Given:

```
class A{
  public List<Number> getList(){
     //valid code
  };
}
class B extends A{
  @Override
  *INSERT CODE HERE*
     //valid code
  };
}
```

What can be inserted in the above code?

A. public List<? extends Integer> getList(){
B. public List<? super Integer> getList(){
C. public ArrayList<? extends Number> getList(){
D. public ArrayList<? super Number> getList(){
E. public ArrayList<Number> getList(){

Explicación: El tipo de devolución del nuevo método sobrescrito debe ser igual o un subtipo del original. Además, al ser de tipo genérico, éste tipo debe ser igual al original.

Pregunta 8

Consider the following code:

```
public class TestClass{
  public void method(Object o){
    System.out.println("Object Version");
  }
  public void method(ClaseB s){
    System.out.println("ClaseB Version");
  }
  public void method(ClaseA s){
    System.out.println("ClaseA Version");
  }
  public static void main(String args[]){
    TestClass tc = new TestClass();
    tc.method(null);
  }
}
```

What would be the output when the above program is compiled and run? (Assume that ClaseB is a subclass of ClaseA)

A. It will print Object Version
B. It will print ClaseA Version
C. It will print ClaseB Version
D. It will not compile.
E. It will throw an exception at runtime

Explicación: Aunque los tres métodos encajan con la llamada, se ejecuta el más específico. Si los parámetros de dos de ellos no tuvieran relación de herencia, se habría producido un error de compilación en la instrucción de llamada a method(null) debido a una ambiguedad.

Pregunta 9

Given the following definitions and reference declarations:

```
interface I1 { }

interface I2 { }

class C1 implements I1 { }

class C2 implements I2 { }

class C3 extends C1 implements I2 { }

C1 o1;

C2 o2;

C3 o3;
```

Which of these statements are legal? (choose 3)

A. class C4 extends C3 implements I1, I2 { }
B. o3 = o1;
C. o3 = o2;
D. I1 i1 = o3; I2 i2 = (I2) i1;
E. I1 b = o3;

Explicación. B no es correcta porque C3 es subtipo de C1, y no al revés. C no es correcta porque no hay relación de herencia entre C3 y C2.

Pregunta 10

This following code appears in a file named VehicleType.java. Why would it not compile?

```
//In file VehicleType

package objective1;

public enum VehicleType

{

   SUV, SEDAN, VAN, SPORTSCAR;

   public VehicleType()

   {   }

}
```

A. VehicleType's definition cannot be public.
B. VehicleType's constructor cannot be public.
C. package statement is invalid for VehicleType.
D. VehicleType must be defined as a class instead of enum since it is the only definition in the file.

Explicación: Los constructores de una enumeración deben ser privados, nunca públicos

Pregunta 11

What will the following code print when compiled and run?

```
class X{
   public X(){
      System.out.print("In X");
   }
}
class Y extends X{
   public Y(){
      super();
      System.out.print("In Y");
   }
}
class Z extends Y{
   public Z(){
      System.out.print("In Z");
   }
}
public class Test {
   public static void main(String[] args) {
      Y y = new Z();
   }
}
```

A. It will not compile.
B. In X In Y In Z
C. In Z In Y In X
D. In Y In X In Z
E. In Z In X In Y

Explicación: Al crearse un objeto de una clase, primero se llama al constructor de la superclase

Pregunta 12

Given:

```
public class MyTest{

        public static void main(String[] args){

                System.out.println(args[1]+":"+args[3]+":"+args[2]);

        }

}
```

Which is the output when execute the following command?:

java MyTest print name my

A. print my name
B. print name my
C. name null my
D. It will throw an exception
E. Compilation fails

Explicación: Se produce una excepción ArrayIndexOutOfBoundsException, dado que el array con los comandos de entrada es de tres elementos ycomo el índice del primero es 0 la posición args[3] no existe.

Pregunta 13

Given:

```
interface Oper{
        public int calc(int a, int b);
}
public class Test{
        public static void main(String[] args) {
                int result=0;
                //line 1
                result=obj.calc(3,8);
                System.out.println(result);
        }
}
```

Which two codes can be inserted in line 1, independently, to compile?

A. Oper obj=new Oper();
B. Oper obj=()->a+b;
C. Oper obj=(a,b)->a*b;
D. Oper obj=return a*b;
E. Oper obj=(int a, int b)->{return a-b;}

Explicación: Al tratarse de una interfaz funcional, se puede implementar mediante expresiones lambda. Como el método de la interfaz tiene dos parámetros, las dos únicas expresiones lambda que se ajustan a ese formato son las indicadas en C y E y en ambas, las implementaciones devuelven correctamente un resultado int.

Pregunta 14

```
package b;

public class Person() {

        protected Person() {   //line 1

        }

}

package a;

import b.Person;

public class Test {  //line 2

        public static void main(String[] args) {

                Person person=new Person();   //line 3

        }

}
```

The program doesn't compile. Which two independent actions resolve the problem? (choose two)

A. In line 1, change the access modifier to private
B. In line 1, change the access modifier to public
C. In line 2, add extends Person to the Test class an in line 3 change de creation object to Person person=new Test();
D. In line 2, change access modifier to protected
E. In line 1, remove the access modifier

Explicación: El programa no compila porque el constructor de Person es inaccesible desde fuera del paquete y, por tanto, la línea 3 provoca un error de compilación. La respuesta A es incorrecta porque si el modificador de acceso es privado el constructor será más inaccesible. B es correcta, ya que si el constructor es publico podrá ser instanciada la clases desde cualquier otra clase de otro paquete. C es correcta porque si Test hereda Person, estando en otro paquete podrá tener acceso al constructor de Person a través de la herencia, es decir, al ser llamado desde el constructor de Test. D es incorrecta porque una clase no puede llevar el modificador protected. E es incorrecta porque al quitar protected, se le daría ámbito de paquete que es aún inferior a protected

Pregunta 15

Given:

```
public class Main{
        public static void main(String[] args){
                Consumer consumer=msg->System.out::print; //line 1
                consumer.accept("hello functional!");
        }
}
```

This code results in a compilation error. Which code should be inserted on line 1 for a successful compilation?

A. Consumer consumer=msg->{return System.out.print(msg);};
B. Consumer consumer=var msg->System.out.print(msg);
C. Consumer consumer=(String msg)->System.out::print(msg);
D. Consumer consumer= System.out::print;

Explicación. Una referencia a método sustituye a una expresión lambda completa, no solo a la parte de implementación. Por tanto:

System.out::print

es equilante a:

n->System.out.print(n)

Pregunta 16

```
public class Test {

	public static void main(String[] args) {

		for(int i=0;i<args.length;i++) {

			System.out.println(i+")."+args[i]);

			switch(args[i]) {

				case "one":

					continue;

				case "two":

					i--;

					continue;

				default:

					break;

			}

		}

	}

}
```

executed with this command:

java Test one two three

Which is the result?

A. 0).one
B. 0).one1).two
C. 0).one1).two2).Three
D. It creates an infinite loop printing 0).one1)two1)two..
E. A java.lang.NullPointerException is thrown

Explicación: Al ejecutar la segunda iteración del for con el segundo de los argumentos de línea de comandos, el switch entra en el caso "two", que decrementa la variable i de su estado actual 1 a 0. Como hay un continue después, el programa va directamente a la instrucción de incremento del for i++, volviendo de nuevo a procesar el segundo argumento de línea de comandos, y así indefinidamente.

Pregunta 17

Which two interfaces can be used in lambda expressions?

A.
```
interface I1{
        void print();
        default void print(int a){}
    }
```

B.
```
interface I2{
        String toString(String cad);
        boolean test(int a, int b);
    }
```

C.
```
interface I3{
        default void comment();
    }
```

D.
```
interface I4{
        int send();
        boolean equals(Object ob);
    }
```

E.
```
interface I5{
        static void print();
        int get(int a);
    }
```

Explicación. La respuesta A define una interfaz con un único método abstracto, por lo que es funcional y se puede implementar mediante expresiones lambda. B contiene dos métodos abstractos, por lo que no es funcional. La interfaz de la respuesta C no contiene ningún método abstracto, por lo que no es funcional. La interfaz de la respuesta D contiene dos métodos abstractos, pero uno de ellos coincide con los de Object, por tanto, no se tiene en cuenta y la interfaz es considerada funcional. La interfaz de E no compila, ya que el método static debe estar implementado

Pregunta 18

Given:

```
List<Integer> nums=List.of(5,19,3,4,12,4,11,22,15,7);

        System.out.println(nums.stream()

                .mapToInt(n->n-4)

                .distinct()

                .filter(n->n>20)

                .findFirst()

                .getAsInt());
```

Which is the result?

A. It prints 0
B. It prints 20
C. It prints 22
D. It prints nothing without error
E. Exception is thrown

Explicación: Primeramente, sobre el stream de números, se aplica una función que resta 4 a cada elemento. Como el mayor de todos ellos es 22, al restar 4 quedaría 18, por lo que al aplicar el filtro ninguno de los elementos del stream cumpliria la condición de ser mayor que 20. Esto siginfica que findFirst() devolverá un Optional vacío, y al aplicarle el método getAsInt() provocará una excepción.

Pregunta 19

What will the following code print when run?

```
import java.nio.file.Path;

import java.nio.file.Paths;

public class PathTest {

   static Path p1 = Paths.get("c:\\level1\\psing\\Main.java");

   public static String getData(){

      String data = p1.getName(0).toString();

      return data;

   }

   public static void main(String[] args) {

      System.out.println(getData());

   }

}
```

A. IllegalArgumentException
B. ArrayIndexOutOfBoundsException
C. c:\
D. c:
E. level1

Explicación: Al aplicar getName(0) sobre el path obtenemos la primara parte del mismo, sin contar la raíz.

Pregunta 20

Given that the file test.txt is accessible and contains multiple lines, which of the following code fragments will correctly print all the lines from the file?

A. Stream<String> lines = Files.find(Paths.get("test.txt"));
   lines.forEach(System.out::println);
B. BufferedReader bfr = new BufferedReader(new FileReader("test.txt"));
   System.out.println(bfr.readLines());
C. Stream<String> lines = Files.list(Paths.get("test.txt"));
   lines.forEach(x->System.out.println(x));
D. Stream<String> lines = Files.lines(Paths.get("test.txt"));
   lines.forEach(System.out::println);
E. List<String> lines = Files.readAllLines(Paths.get("test.txt"));
   lines.forEach(s -> System.out.println(s));

Explicación: Los métodos de Files que devuelven el contenido de un fichero son lines(), que devuelve un Stream de cadenas de texto y readAllLines(), que devuelve una lista con todas las cadenas

Pregunta 21

Consider the following code:

```
public static boolean isValid(Path p){
    return p.startsWith("temp") && p.endsWith("geo.dat");
}
public static void print() {
    var p1 = Paths.get("\\temp\\datas");
    var p2 = p1.resolve("geo.dat");
    System.out.println(p2+" "+isValid(p2));
}
```

What will be printed when the method print() is executed?

A. \temp\datas\clients false
B. temp\datas\geo.dat false
C. \temp\datas\geo.dat false
D. temp\datas\geo.dat true
E. geo.dat false
F. \geo.dat false

Explicación: Dado que el path indicado en p2 es relativo, la llamada a resolve() devolverá la unión de ambos path. Como dicho path comienza por "\temp" en lugar de "temp", el método isValid() devolverá false.

Pregunta 22

Given:

```
String qr = INSERT CODE HERE

try(PreparedStatement ps =  connection.prepareStatement(qr);){

        ...

}
```

What can be inserted in the above code? (choose 2)

A. "insert into USERINFO values(?, ?, ?, ?)"; (Assuming USERINFO table with four columns exists.)
B. "update USERINFO set NAME=?1 where ID=?2"; (Assuming USERINFO table with NAME and ID columns exists.)
C. "delete USERINFO from ID=2"; (Assuming USERINFO table with ID column exists.)
D. "select * from USERINFO where ID=2"; (Assuming USERINFO table with ID column exists.)

Explicación: La instrucción SQL de la respuesta B es incorrecta porque los parámetros se indican mediante el símbolo de interrogación simplemente, sin ningún número. La instrucción SQL de la respuesta C es incorrecta, dado que la condición se debe indicar en la cláusula where, no from.

Pregunta 23

Which of the following are valid JDBC URLs? (choose 2)

A. jdbc:mysql://localhost:3306/sample
B. jdbc://oracle.com/sample
C. mysql://jdbc.mysql/sample
D. jdbc:oracle:thin:@localhost:1521:testdb
E. http://192.168.1.10:3306/mysql/sampledb

Explicación: Las cadenas de conexión jdbc siguien el formato

jdbc:tipo:conexion

En el caso de la respuesta B, no es correcta porque se debería indicar el subprotocolo después de jdbc:, mientras que las cadenas indicadas en C y D son incorrectas porque no comienzan por jdbc

Pregunta 24

Assuming the Product class has a getPrice method, this code does not compile:

```
List products=List.of(new Product("play station",234.5),
                      new Product("mobile",190.2),
                      new Product("smart watch",98.7)); //line 1
Stream pst=products.stream(); //line 2
pst.filter(p->p.getPrice()>100.00)  //line 3
        .forEach(Sustem.out::println);
```

Which two statements, independently, would allow this code to compile? (Choose two.)

A. Replace line 3with pst.filter(a –> ((Product)a).getPrice() > 100.00)

B. Replace line 1 with List<Product> pst = products.stream();

C. Replace line 3 with pst.filter((Product a) –> a.getPrice() > 100.00)

D. Replace line 2 with Stream<Product> pst = products.stream();

Explicación. El programa no compila porque, al no haber especificado el tipo del stream, el compilador no reconoce el método getPrice() del objeto especificado en la expresión lambda de la línea 3. Con un casting como el indicado en l respuesta A o definiendo el Stream del tipo específico de objeto, el problema se resolvería

Pregunta 25

Given:

```
var nums = List.of(1, 2, 3, 4, 5, 6, 7).stream();

Predicate<Integer> p = //a predicate goes here

Optional<Integer> value = nums.filter(p).reduce((a, b)->a+b);

value.ifPresent(System.out::println);
```

Select two options:

A. setting p to a->a<0; will produce no output.
B. setting p to a->a<0; will generate a NullPointerException.
C. setting p to a->a<0; will generate a NoSuchElementException.
D. setting p to a->a%2==0; will produce 12.
E. setting p to a->a%2==0; will produce 16.

Explicación: En la respuesta A, la condición del predicado no es cumplida por ninguno de los elementos, por lo que el Optional generado estará vacío, isPresent resultará falso y no se mostrará nada. No se va a producir ningún error, por lo que las respuestas B y C son incorrectas. La condición del predicado de la respuesta D hace que el filtro devuelva los elementos 2, 4 y 6, que al ser sumados mediante reduce() dan como resultado 12. Dado que D es correcta, la respuesta E no puede serlo.

Pregunta 26

Given:

```
List<String> strList = Arrays.asList("a", "aa", "aaa");

Function<String, Integer> f = x->x.length();

Consumer<Integer> c = x->System.out.print("Len:"+x+" ");

Predicate<String> p = x->"".equals(x);

strList.forEach(  *INSERT CODE HERE*  );
```

What can be inserted in the code above?

A. f
B. c
C. p
D. c and p
E. None of the above.

Explicación. El método forEach de la lista de cadenas requiere un Consumer<String>, por lo que ninguna de las implementaciones definidas es válida.

Pregunta 27

You have been given an instance of an Executor and you use that instance to execute tasks. How many threads will be created for executing these tasks by the Executor?

A. Exactly 1.
B. One thread for each task that is submitted to the Executor.
C. As many as there are cores in the CPU.
D. Number of threads created by the Executor depends on how the Executor instance was created.
E. Number of threads is automatically determined based on the load on the Executor instance.

Explicación: Existen varios métodos en Executors que nos devuelven una implementación de Executor, pero el número de threads utilizados depende del método utilizado, por tanto, de la implementación generada.

Pregunta 28

Consider the following code appearing in a module-info.java

```
module com.members{
  requires transitive org.mytype;
}
```

Identify correct statements.

A. Any module that requires the com.members module must also require the org.mytype module.
B. Any module that requires the com.members module can use the org.mytype module without requiring it.
C. Only a module that requires the com.members module can use the org.mytype module.
D. Any module that requires the org.mytype module must also require the com.members module.
E. Any module that requires the com.members module must require the org.mytype module instead of com.members.

Explicación: Cuando un módulo requiere transitivamente a otro, quien requiera al principal requerirá también implicitamente a requerido de forma transitiva.

Pregunta 29

You are creating a xyz.data module that makes its two classes - com. xyz.data.Part1 and com.xyz.data.Part2 - available to all other modules for use.

Which of the following files correctly defines this module?

A. module xyz.data { exports com.xyz.data.*; }
B. module xyz.data { exports com.xyz.data; }
C. module xyz.data { exports com.xyz.data to all; }
D. module xyz.data { exports com.xyz.data.Part1, com. xyz.data.Part2; }
E. module xyz.data { exports com.xyz.data.Part1; exports com. xyz.data.Part2; }

Explicación. Cuando se define un módulo y se quiere hacer accesible un paquete al exterior, la sintaxis es

exports nombre_paquete;

La única respuesta que se ajusta a ese formato es la indicada en B

Pregunta 30

Given:

```
var data = new ArrayList<>();

data.add("Landing");

data.add(30);

data.add("Page");

data.set(1, 25);

data.remove(2);

data.set(2, 200L);

System.out.print(data);
```

What is the output?

A. [Page, 200]

B. [Landing, 30, Page]

C. [Landing, 25, null, 200]

D. An exception is thrown at run time

Explicación. Hay dos elementos en la colección, por lo que la posición 2 no existe. Al intentar establecer un valor en dicha posición, se produce una IndexOutOfBoundsException. Con el método add si podríamos añadir un nuevo valor en la primera posición libre:

```
data.add(2,200L);
```

Pero con set no, ya que este método se utiliza para modificar el contenido de una posición existente

Pregunta 31

Which code fragment prints 100 random numbers?

A. var r=new Random();
   DoubleStream.generate(r::nextDouble).limit(100).forEach(System.out::println);
B. DoubleStream.generate(Random::nextDouble).limit(100).forEach(System.out::println);
C. DoubleStream.generate(Random::nextDouble).skip(100).forEach(System.out::println);
D. DoubleStream.stream(Random::nextDouble).limit(100).forEach(System.out::println);

Explicación. El método generate() de Stream requiere una implementación de Supplier. Si esta implementación consiste en generar números aleatorios y r es la varaible que apunta a un objeto Random, la implementación de Supplier puede ser:

()->r.nextDouble()

o

r::nextDouble

Pregunta 32

Given:

```
public class Painter{
        public Painter(){
        }
        //Painter methods
}
```

You want to use the Painter class in a try-with-resources statement.

Which change will accomplish this?

A. Extend AutoCloseable and override the close method.

B. Implement AutoCloseable and override the autoClose method.

C. Extend AutoCloseable and override the autoClose method.

D. Implement AutoCloseable and override the close method.

Explicación: Para que un objeto pueda ser utilizado en un try con recursos, su clase debe implementar la interfaz AutoCloseable y sobreescribir su único método close()

Pregunta 33

Which is a correct implementation of a BinaryOperator functional interface?

A. (a,b)->System.out.println(a+":"+b)
B. a,b->a+b
C. (a,b)->String.toString(a,b)
D. (var a,b)->a.compareTo(b)
E. (a,b)->a.concat(b)

Explicación. La interfaz BinaryOperator define un método con dos parámetros y un tipo de devolución, los tres del mismo tipo. En este sentido, la respuesta A es incorrecta porque la implementación no devuelve ningún resultado. Las respuestas B y D no son correctas porque la lista de parámetros está mal definida. C no es correcta porque el formato del método toString es incorrecto. La correcta es la E, ya que si a y b son String, el resultado de concat es también un String

Pregunta 34

Given:

```
@Retention(RetentionPolicy.RUNTIME)
public @interface Logging {
   String value() default "";
   String[] params();
   String date();
   int depth() default 10;
}
```

Which of the following options correctly uses the above annotation?

A. @ Logging (date = "2020-04-17", params = { null })
   void process (int index){ }
B. @ Logging (date = "2019", params = "index")
   void process(int index){ }
C. @ Logging (depth = 10, date = "04/10/2005", params = {"index"}, value=" pr ")
   static final String cad = null;
D. @ Logging ({"index"}, "23/01/2022")
   void process (int index){ }
E. @ Logging ("value", params={"index"}, date="30/05/2019")
   void process (int index){ }

Explicación. La respuesta A es incorrecta porque el valor del atributo params no puede ser null. La respuesta B es correcta, los atributos date y params tienen valores correctos y value y depth, que no están asignados, adquieren los valores por defecto. La respuesta C es correcta, en le caso de params, como es de tipo array, se indican los valores entre llaves, aunque si es solo un elemento no sería necesario ponerlas, tal y como ocurre en la respuesta B. La respuesta D es incorrecta porque, si se asignan más de un atributo es necesario indicar explícitamente los nombres y sus valores. E es incorrecta por el mismo motivo que D

Pregunta 35

Consider the following code:

```
import java.util.*;

import java.text.*;

public class TestClass{

   public static void main(String[] args) throws Exception    {

      double amount = 100600.5;

      Locale jp = new Locale("jp", "JP");

      //1 create formatter here.

      System.out.println(  formatter.format(amount)  );

   }

}
```

How will you create formatter using a factory at //1 so that the output is in Japanese Currency format?

A. NumberFormat formatter = NumberFormat.getCurrencyFormatter(jp);
B. NumberFormat formatter = new DecimalFormat(jp);
C. NumberFormat formatter = NumberFormat.getCurrencyInstance(jp);
D. NumberFormat formatter = NumberFormat.getInstance(jp);
E. NumberFormat formatter = new DecimalFormat("#.00");

Explicación. Dado que se trata de crear una implementación de NumberFormat para formateado de moneda, se debe utilizar el método estático getCurrencyInstance() para su creación

Pregunta 36

Given:

```
public class Employ{
        private Map<String, Double> info=new HashMap<>();
        private Lock lock=new ReentrantLock();
        public void setData(String name, Double salary){
                //Line 1
                try{
                        info.put(name,salary);
                }finally{
                        //Line 2
                }
        }

}
```

What should be inserted at line 1 and line 2?

A. lock.adquired(); and lock.release();
B. lock.lock(); and lock.unlock();
C. lock.lock().adquire(); and lock.lock().release();
D. lock.readLock(); and lock.writeLock();

Explicación: Los métodos de Lock para bloquear y desbloquear el acceso a un bloque de código a otros hilos son lock() y unlock(), respectivamente. El primero antes de la instrucción de llamada al método del objeto compartido y el segundo en el bloque finally para garantizar su ejecución

Pregunta 37

Given

```
public class Base{

        public <T> Collection<T> mybase(Collection<T> args){...}

}
```

and

```
public class Child extends Base{...}
```

Which two statements are true if the method is added to Child?(choose two)

A. public Collection<String> mybase(Collection<String> arg) { ... } overrides Base.mybase.

B. public <T> Collection<T> mybase(Stream<T> arg) { ... } overloads Base.mybase.

C. public <T> List<T> mybase(Collection<T> arg) { ... } overrides Base.mybase.

D. public <T> Collection<T> mybase(Collection<T> arg) { ... } overloads Base.mybase.

E. public <T> Collection<T> mychild(Collection<T> arg) { ... } overloads Base.mybase.

F. public <T> Iterable<T> mybase(Collection<T> arg) { ... } overrides Base.mybase.

Explicación: La respuesta A es incorrecta porque a la hora de sobrescribir un método con tipos genéricos no se puede indicar un tipo específico. La respuesta B es correcta, ya el tipo del parámetro del nuevo método es distinto al del método de la clase Base y, por tanto, estamos ante un caso de sobrecarga. La sobrescritura de la respuesta C es correcta dado que List es un subtipo de Collection. D es incorrecta porque el nuevo método es idéntico al heredado, así que sería una sobrescritura y no sobrecarga. La respuesta E es incorrecta porque se trataría de dos métodos con nombre diferente, por tanto, no es sobrecarga. La respuesta F es incorrecta porque se cambia el tipo de devolución del método en la sobrescritura y esto solo puede hacerse si el nuevo tipo es subclase del tipo de devolución del original, como Iterable no es subtipo de Collection, es incorrecto

Pregunta 38

Given

```
int[] secA={2,4,6,8,10};

int[] secB={2,4,8,6,10};

int res1=Arrays.mismatch(secA, secB);

int res2=Arrays.compare(secA, secB);

System.out.print(res1+" : "+res2);
```

What is the result?

A. -1:2
B. 2:-1
C. 2:3
D. 3:0

Explicación: La primera diferencia entre ambos arrays se da en la posición 2, por tanto, el método mismatch() devuelve 2. Como el primer array es menor que el segundo, la llamada a compare() devuelve -1

Pregunta 39

Given:

List<Integer> nums=List.of(34, 56,19,80,25,100);

System.out.println(/*Insert code here*/);

Which of the following options will correctly print then numbers of elements that are greater than 50?

A. nums.toStream().reduce(n->n>50).length()
B. nums.stream().map(n->n>50).count()
C. nums.asStream().filter(n->n>50).forEach()
D. nums.stream(n->n>50).size()
E. nums.stream().filter(n->n>50).count()

Explicación: Para obtener un stream a partir de la lista y poder filtrarla se debe utilizar el método stream(), por lo que las respuestas A y C son descartadas. La respuesta B tampoco es correcta porque el método map realiza una transformación de elementos, no un filtrado. El método stream() no recibe ningún argumento, por lo que D tampoco es válida

Pregunta 40

Given:

```
public class Person{
        private String name;
        private String street;
        private int age;
        public Person(String n, String s, int a){
                name=n;street=s;age=a;
        }
        //only getters
}
```

Which of the following actions implements the Java SE Secure guidelines to achieve this class to be inmutable?

A. Make the class final
B. Create private setters methods
C. Make the fields final
D. Make getter methods synchronized

Explicación: Para que los objetos de la clase sean inmutables, se debe impedir que los atributos puedan ser modificados una vez que se han establecido. Esto se consigue definiéndolos como final. Hacer la clase final no garantiza su inmutabilidad, solamente que no se puede heredar.