

Anexo 2

Pruebas para la obtención de títulos de Técnico y Técnico Superior

Convocatoria correspondiente al curso académico 2025-2026

(Resolución de 12 de diciembre de 2025 de la Dirección General de Educación Secundaria, Formación Profesional y Régimen Especial)

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

Código del ciclo:	Denominación completa del título:
IFCS03	Técnico Superior en Desarrollo de Aplicaciones Web
Clave o código del módulo:	Denominación completa del módulo profesional:
05	Programación

INSTRUCCIONES GENERALES PARA LA REALIZACIÓN DE LA PRUEBA
- El examen tendrá una duración de 2h. - La prueba consta de un examen tipo test con cuatro opciones de las cuales solamente una es correcta. - Cada pregunta se responderá en el espacio dejado al efecto, en la hoja de respuestas, la hoja 2. Se usarán X en los recuadros para señalar la respuesta seleccionada. - Si se quiere rectificar una respuesta contestada, se rellenará toda la casilla de la respuesta incorrecta, tal y como se puede apreciar aquí: a ☐ b ☒ c ☐ d ☐ - Se dispondrá de una hoja para borrador (o de varias si se requieren), que será proporcionada por el centro. Esa hoja se entregará obligatoriamente al final junto con el examen, si bien nada de lo escrito en la hoja de borrador se valorará en la corrección. - Sólo se utilizará bolígrafo negro o azul, no permitiéndose usar bolígrafo rojo, lapicero, Tipp-Ex, etc. - Por supuesto, tampoco se podrá emplear ningún dispositivo electrónico. - Cualquier tachadura o borrón en una respuesta podrá invalidar toda la puntuación de esta.

CRITERIOS DE CALIFICACIÓN Y VALORACIÓN
- El test se calificará sobre 10 puntos. Todas las preguntas se calificarán equitativamente con la misma cantidad de puntos. En cada pregunta se plantearán varias respuestas, y se deberá señalar la única que se considere correcta, según el caso. Cada respuesta correcta que se marque se valorará con 0,25 puntos, y si se marca alguna incorrecta, se valorará con una cantidad negativa equivalente a 1/3 de cada respuesta correcta. Es decir, se descontarán 0,08 puntos. Si no se está seguro de si una respuesta es correcta o no, y no se marca, no sumará ni restará puntos. - Calificación final del módulo profesional: - El alumno obtendrá en el módulo profesional una calificación entera entre 1 y 10. Dicha calificación se calculará redondeando la conseguida en la prueba. Si los decimales son inferiores a 0,5 la calificación se redondeará al entero más bajo; si son superiores o iguales a 0,5 al entero más alto. Esta regla tiene una excepción: las notas de examen inferiores a 1 se redondearán a 1.

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

CALIFICACIÓN

RESPUESTAS TEST

1 a ☐ b ☐ c ☐ d ☐	11 a ☐ b ☐ c ☐ d ☐	21 a ☐ b ☐ c ☐ d ☐	31 a ☐ b ☐ c ☐ d ☐
2 a ☐ b ☐ c ☐ d ☐	12 a ☐ b ☐ c ☐ d ☐	22 a ☐ b ☐ c ☐ d ☐	32 a ☐ b ☐ c ☐ d ☐
3 a ☐ b ☐ c ☐ d ☐	13 a ☐ b ☐ c ☐ d ☐	23 a ☐ b ☐ c ☐ d ☐	33 a ☐ b ☐ c ☐ d ☐
4 a ☐ b ☐ c ☐ d ☐	14 a ☐ b ☐ c ☐ d ☐	24 a ☐ b ☐ c ☐ d ☐	34 a ☐ b ☐ c ☐ d ☐
5 a ☐ b ☐ c ☐ d ☐	15 a ☐ b ☐ c ☐ d ☐	25 a ☐ b ☐ c ☐ d ☐	35 a ☐ b ☐ c ☐ d ☐
6 a ☐ b ☐ c ☐ d ☐	16 a ☐ b ☐ c ☐ d ☐	26 a ☐ b ☐ c ☐ d ☐	36 a ☐ b ☐ c ☐ d ☐
7 a ☐ b ☐ c ☐ d ☐	17 a ☐ b ☐ c ☐ d ☐	27 a ☐ b ☐ c ☐ d ☐	37 a ☐ b ☐ c ☐ d ☐
8 a ☐ b ☐ c ☐ d ☐	18 a ☐ b ☐ c ☐ d ☐	28 a ☐ b ☐ c ☐ d ☐	38 a ☐ b ☐ c ☐ d ☐
9 a ☐ b ☐ c ☐ d ☐	19 a ☐ b ☐ c ☐ d ☐	29 a ☐ b ☐ c ☐ d ☐	39 a ☐ b ☐ c ☐ d ☐
10 a ☐ b ☐ c ☐ d ☐	20 a ☐ b ☐ c ☐ d ☐	30 a ☐ b ☐ c ☐ d ☐	40 a ☐ b ☐ c ☐ d ☐

Correctas _____ Incorrectas _____ No Puntuadas/Sin Contestar _____

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

CONTENIDO DE LA PRUEBA:

1.- Dado el siguiente fragmento de código, ¿cuál es el resultado de ejecutar el comando java Test 9 4?

```
public class Test {
    public static void main(String[] args) {
        int a = Integer.parseInt(args[0]);
        int b = Integer.parseInt(args[1]);
        System.out.println(a % b == 0 ? "Múltiplo" : "No múltiplo");
    }
}
```

- a) Múltiplo
- b) No múltiplo
- c) Error en tiempo de compilación
- d) Error en tiempo de ejecución

2.- En JDBC, ¿cuál es el orden correcto para ejecutar una consulta SQL parametrizada?

- a) DriverManager → ResultSet → PreparedStatement → Connection
- b) Connection → ResultSet → PreparedStatement → DriverManager
- c) DriverManager → Connection → PreparedStatement → ResultSet
- d) Connection → Statement → DriverManager → ResultSet

3.- ¿Qué describe el "contrato" de las colecciones tipo hash (HashSet / HashMap)?

- a) equals() es suficiente; hashCode() solo mejora el rendimiento.
- b) Si equals() es true, hashCode() debe ser idéntico obligatoriamente.
- c) Una colisión de hash lanza una IllegalStateException.
- d) Se prohíbe técnicamente el uso de objetos mutables como claves.

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

4.- ¿Cuál es la salida del siguiente programa?

```
public class Counter {
    static int total = 0;
    int local;

    Counter(int v) {
        local = v;
        total += v;
    }

    public static void main(String[] args) {
        Counter c1 = new Counter(3);
        Counter c2 = new Counter(7);
        Counter c3 = new Counter(c1.local);
        System.out.println(total + " " + c3.local);
    }
}
```

- a) 10 3
- b) 13 3
- c) 10 7
- d) 13 7

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

5.- ¿Cuál es la salida del siguiente código?

```
public class Orden {
    static String s = "A";
    static {
        s += "B";
    }
    String t;
    {
        t = s + "C";
    }
    Orden() {
        t += "D";
    }
    public static void main(String[] args) {
        Orden o = new Orden();
        System.out.println(o.t);
    }
}
```

- a) ACD
- b) ABCD
- c) BCD
- d) ABD

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

6.- ¿Cuál es la salida del siguiente programa?

```
class Base {  
    static String nombre() { return "Base"; }  
    String instancia() { return "base"; }  
}  
class Derivada extends Base {  
    static String nombre() { return "Derivada"; }  
    String instancia() { return "derivada"; }  
}  
public class Test {  
    public static void main(String[] args) {  
        Base obj = new Derivada();  
        System.out.println(obj.nombre() + " " + obj.instancia());  
    }  
}
```

- a) Derivada derivada
- b) Base derivada
- c) Base base
- d) Derivada base

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

7.- ¿Cuál es la salida del siguiente programa?

```
public class ExTest {  
    public static void main(String[] args) {  
        try {  
            metodo();  
            System.out.print("A");  
        } catch (RuntimeException e) {  
            System.out.print("B");  
        } catch (Exception e) {  
            System.out.print("C");  
        } finally {  
            System.out.print("D");  
        }  
        System.out.print("E");  
    }  
    static void metodo() throws Exception {  
        throw new NullPointerException("npe");  
    }  
}
```

- a) BDE
- b) CDE
- c) BCD
- d) ADE

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

8.- ¿Cuál es la salida del siguiente código?

```
public class WrapEx {
    public static void main(String[] args) {
        try {
            String s = null;
            int n = Integer.parseInt(s);
            System.out.println(n);
        } catch (NumberFormatException e) {
            System.out.println("NFE");
        } catch (NullPointerException e) {
            System.out.println("NPE");
        }
    }
}
```

- a) NFE
- b) NPE
- c) 0
- d) Error de compilación

9.- ¿Cuál es la salida del siguiente código?

```
public class StringTest {
    public static void main(String[] args) {
        String a = "hola";
        String b = "hola";
        String c = new String("hola");
        String d = c.intern();
        System.out.println(a == b);
        System.out.println(a == c);
        System.out.println(a == d);
    }
}
```

- a) true false true
- b) true true true
- c) false false true
- d) true false false

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

10.- Dada la siguiente clase, ¿qué valor tendrá el campo 'password' tras deserializar un objeto previamente serializado?

```
import java.io.Serializable;
public class Usuario implements Serializable {
    private String nombre = "admin";
    private transient String password = "secreto";
}
```

- a) "secreto"
- b) null
- c) "" (cadena vacía)
- d) Error de serialización en tiempo de ejecución

11.- ¿Cuál es la salida del siguiente código?

```
class A {
    A() { System.out.print("A"); }
}
class B extends A {
    B() { System.out.print("B"); }
}
class C extends B {
    C() { System.out.print("C"); }
    public static void main(String[] args) {
        new C();
    }
}
```

- a) ABC
- b) CBA
- c) C
- d) Error de compilación

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

12.- ¿Cuál es la salida del siguiente código?

```
class Demo {
    static boolean activo;
    public static void main(String[] args) {
        int x;
        if (activo == true)
            x = 100;
        else
            x = 200;
        System.out.println(x);
    }
}
```

- a) 100
- b) 200
- c) 0
- d) Error de compilación

13.- ¿Qué diferencia hay entre Class.getMethods() y Class.getDeclaredMethods() en Java?

- a) getMethods() devuelve solo los métodos privados; getDeclaredMethods() devuelve solo los públicos.
- b) getMethods() devuelve todos los métodos públicos incluidos los heredados; getDeclaredMethods() devuelve todos los métodos declarados en la clase (cualquier visibilidad) excluyendo los heredados.
- c) Ambos devuelven exactamente los mismos métodos.
- d) getMethods() devuelve métodos estáticos; getDeclaredMethods() devuelve métodos de instancia.

14.- ¿Qué devuelve Object.class.getSuperclass() en Java?

- a) La clase java.lang.Class
- b) La clase java.lang.Object
- c) null
- d) Lanza una excepción NoSuchMethodException

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

15.- ¿Cuál es la salida del siguiente programa?

```
class Animal {
    String tipo;
    Animal(String tipo) { this.tipo = tipo; }
    public String toString() { return "Animal:" + tipo; }
}
class Perro extends Animal {
    String nombre;
    Perro(String nombre) {
        super("Mamifero");
        this.nombre = nombre;
    }
    public String toString() {
        return super.toString() + "/Perro:" + nombre;
    }
    public static void main(String[] args) {
        System.out.println(new Perro("Rex"));
    }
}
```

- a) Animal:Mamifero
- b) Perro:Rex
- c) Animal:Mamifero/Perro:Rex
- d) Error de compilación

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

16.- ¿Cuál es la salida del siguiente código?

```
import java.util.HashMap;
public class MapTest {
    public static void main(String[] args) {
        HashMap<String, Integer> mapa = new HashMap<>();
        mapa.put("a", 1);
        mapa.put("b", 2);
        mapa.merge("a", 10, Integer::sum);
        mapa.merge("c", 5, Integer::sum);
        System.out.println(mapa.get("a") + " " + mapa.get("c"));
    }
}
```

- a) 10 5
- b) 1 5
- c) 11 5
- d) 11 null

17.- ¿Cuál es la salida del siguiente código?

```
public class Ops {
    public static void main(String[] args) {
        int n = 10;
        int a = n++ + ++n;
        int b = --n + n--;
        System.out.println(a + " " + b + " " + n);
    }
}
```

- a) 22 22 10
- b) 21 22 10
- c) 22 23 10
- d) 22 22 11

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

18.- ¿Cuál es la salida del siguiente programa?

```
public class BoxTest {
    public static void main(String[] args) {
        Integer x = 127;
        Integer y = 127;
        Integer p = 125;
        Integer q = 125;
        System.out.println(x == y);
        System.out.println(p == q);
    }
}
```

- a) true true
- b) false false
- c) true false
- d) false true

19.- ¿Cuál es la salida del siguiente código?

```
public class Sobrecarga {
    static void imprimir(int a, int b) {
        System.out.println("int,int");
    }
    static void imprimir(double a, double b) {
        System.out.println("double,double");
    }
    static void imprimir(int... nums) {
        System.out.println("varargs");
    }
    public static void main(String[] args) {
        imprimir(1, 2);
    }
}
```

- a) int,int
- b) double,double
- c) varargs
- d) Error de compilación por ambigüedad

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

20.- ¿Cuál es la salida del siguiente código?

```
public class MatrizTest {
    public static void main(String[] args) {
        int[][] m = new int[3][];
        m[0] = new int[]{1, 2};
        m[1] = new int[]{3, 4, 5};
        System.out.println(m[1][2] + " " + m.length + " " + m[0].length);
    }
}
```

- a) 5 3 2
- b) 4 3 3
- c) 5 2 2
- d) Error en tiempo de ejecución

21.- ¿Cuál de las siguientes clases convierte un InputStream de bytes en un Reader de caracteres con una codificación específica?

- a) FileReader
- b) BufferedReader
- c) InputStreamReader
- d) CharArrayReader

22.- ¿Cuál de las siguientes afirmaciones sobre FileInputStream y FileReader en Java es correcta?

- a) Ambas clases leen bytes sin ninguna diferencia.
- b) FileInputStream lee bytes crudos; FileReader lee caracteres usando la codificación por defecto del sistema.
- c) FileReader es más eficiente que FileInputStream para archivos binarios.
- d) FileInputStream puede leer archivos de texto pero FileReader no.

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

23.- Dada una clase Sensor que contiene un atributo static int totalSensores y un atributo de instancia double lecturaActual, ¿cuál de las siguientes afirmaciones describe correctamente la composición del "estado" de un objeto específico de dicha clase?

- El estado del objeto incluye tanto a totalSensores como a lecturaActual, ya que el objeto puede acceder y modificar ambos valores en cualquier momento.
- El estado del objeto está definido únicamente por lecturaActual, puesto que los atributos estáticos pertenecen a la clase y no forman parte de la identidad o estado independiente de una instancia concreta.
- El estado del objeto solo existe si los atributos son mutables; si lecturaActual se declarase como final, el objeto pasaría a considerarse un objeto sin estado.
- El estado de un objeto se define exclusivamente por el conjunto de sus métodos públicos (comportamiento), siendo los atributos simplemente un soporte técnico secundario.

24.- ¿Qué ocurre si se deserializa un objeto y la clase ha cambiado su serialVersionUID respecto al que tenía cuando fue serializado?

- Se ignora el cambio y la deserialización continúa normalmente.
- Se lanza una InvalidClassException en tiempo de ejecución.
- Se lanza un ClassNotFoundException en tiempo de compilación.
- El objeto se deserializa pero todos sus campos toman valores por defecto.

25.- ¿Cuál es la jerarquía correcta de clases de excepción para IOException en Java?

- Throwable → Error → IOException
- Throwable → Exception → RuntimeException → IOException
- Throwable → Exception → IOException
- Throwable → IOException → Exception

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

26.- Dadas las clases Nodo y Lista, ¿cuál es la salida al ejecutar el main?

```
class Nodo {
    int valor;
    Nodo(int v) {
        System.out.print("N" + v + " ");
        this.valor = v;
    }
}
class Lista {
    Nodo cabeza;
    Lista(int v) {
        System.out.print("L" + v + " ");
        this.cabeza = new Nodo(v * 2);
    }
    public static void main(String[] args) {
        Lista l1 = new Lista(3);
        Lista l2 = new Lista(5);
    }
}
```

- a) L3 N6 L5 N10
- b) N6 L3 N10 L5
- c) L3 L5 N6 N10
- d) N3 L6 N5 L10

27.- ¿Cuál de las siguientes formas de leer un fichero de texto es más segura en Java moderno (cierre automático de recursos)?

- a) `FileReader fr = new FileReader("f.txt"); BufferedReader br = new BufferedReader(fr); // leer...`
- b) `try (BufferedReader br = new BufferedReader(new FileReader("f.txt"))) { // leer... }`
- c) `BufferedReader br = new BufferedReader(new FileReader("f.txt")); try { // leer... } finally { br.close(); }`
- d) Las opciones a) y c) son equivalentes en seguridad a la b).

28.- ¿Cuál de los siguientes fragmentos causará un error de compilación en Java por posible pérdida de precisión?

- a) `int i = (int) 3.14;`
- b) `double d = 5;`
- c) `long l = 2147483648L;`
- d) `float f = 3.14;`

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

29.- ¿Cuál es la principal ventaja de usar PreparedStatement frente a Statement en JDBC?

- a) PreparedStatement permite consultas sin conexión a la base de datos.
- b) PreparedStatement solo funciona con bases de datos Oracle.
- c) PreparedStatement previene inyección SQL y puede reutilizarse con diferentes parámetros de forma más eficiente.
- d) PreparedStatement retorna los resultados más rápido porque evita el uso de ResultSet.

30.- ¿Cuál es el resultado de System.out.println(12 & 10);?

- a) 2
- b) 8
- c) 22
- d) 6

31.- ¿Cuál es la salida del siguiente código si existe el fichero 'datos.txt'?

```
import java.io.*;
public class LeerFichero {
    public static void main(String[] args) {
        try {
            BufferedReader br = new BufferedReader(new FileReader("datos.txt"));
            System.out.print("Abierto");
            br.close();
        } catch (FileNotFoundException e) {
            System.out.print("No encontrado");
        } catch (IOException e) {
            System.out.print("IOException");
        } finally {
            System.out.print(" Fin");
        }
    }
}
```

- a) No encontrado Fin
- b) Abierto Fin
- c) IOException Fin
- d) Abierto

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

32.- ¿Cuál es la salida del siguiente código?

```
public class PasoValor {  
    static int cuenta = 0;  
    public static void incrementar(int n) {  
        n += 5;  
        cuenta += 5;  
    }  
    public static void main(String[] args) {  
        int local = 10;  
        incrementar(local);  
        System.out.println(local + " " + cuenta);  
    }  
}
```

- a) 15 5
- b) 10 5
- c) 15 0
- d) 10 0

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

33.- ¿Cuál es la salida del siguiente programa?

```
public class Anidado {
    public static void metodoA() {
        try {
            int[] arr = new int[2];
            arr[5] = 1;
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.print("A");
            throw new RuntimeException("re", e);
        }
    }
    public static void main(String[] args) {
        try {
            metodoA();
        } catch (RuntimeException e) {
            System.out.print("B");
        } finally {
            System.out.print("C");
        }
    }
}
```

- a) ABC
- b) AC
- c) BC
- d) AB

34.- En JDBC, ¿qué método se debe llamar para desactivar el modo autocommit y gestionar transacciones manualmente?

- a) connection.setManualCommit(true)
- b) connection.beginTransaction()
- c) connection.setAutoCommit(false)
- d) connection.startTransaction(false)

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

35.- ¿Cuál es la salida del siguiente código?

```
public class Conversion {
    public static void main(String[] args) {
        byte b = (byte) 200;
        System.out.println(b);
    }
}
```

- a) 200
- b) -56
- c) 0
- d) Error de compilación

36.- ¿Cuál es la salida del siguiente código?

```
class Figura {
    double area() { return 0; }
}
class Circulo extends Figura {
    double radio;
    Circulo(double r) { this.radio = r; }
    double area() { return Math.PI * radio * radio; }
}
public class Test {
    public static void main(String[] args) {
        Figura f = new Circulo(1);
        System.out.println(f instanceof Circulo);
        System.out.println(((Circulo) f).radio);
    }
}
```

- a) true 1.0
- b) false 1.0
- c) true Error en tiempo de ejecución
- d) Error de compilación

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

37.- Dada la siguiente clase Producto, ¿cuál de las siguientes sentencias instancia correctamente un objeto?

```
public class Producto {  
    private String nombre;  
    private double precio;  
    public Producto(String nombre, double precio) {  
        this.nombre = nombre;  
        this.precio = precio;  
    }  
}
```

- a) Producto p = new Producto('Libro', 9.99);
- b) Producto p = new Producto("Libro", 9.99);
- c) Producto p = new Producto("Libro", "9.99");
- d) Producto p = Producto("Libro", 9.99);

DATOS DEL ASPIRANTE			FIRMA
APELLIDOS:			
Nombre:	D.N.I. N.I.E. o Pasaporte:	Fecha:	

38.- ¿Cuál es la salida del siguiente programa?

```
class SaldoInsuficiente extends Exception {
    double monto;
    SaldoInsuficiente(double m) { super("Saldo insuficiente"); monto = m; }
}

public class Banco {
    static double saldo = 100;
    static void retirar(double m) throws SaldoInsuficiente {
        if (m > saldo) throw new SaldoInsuficiente(m);
        saldo -= m;
    }
    public static void main(String[] args) {
        try {
            retirar(50);
            System.out.print(saldo + " ");
            retirar(80);
            System.out.print(saldo);
        } catch (SaldoInsuficiente e) {
            System.out.print(e.getMessage() + " " + e.monto);
        }
    }
}
```

- a) 50.0 Saldo insuficiente 80.0
- b) 100.0 Saldo insuficiente 80.0
- c) 50.0 30.0
- d) Saldo insuficiente 80.0

39.- ¿Cuál de las siguientes afirmaciones sobre modificadores de acceso en Java es INCORRECTA?

- a) Un miembro public es accesible desde cualquier clase.
- b) Un miembro private es accesible solo dentro de su propia clase.
- c) Un miembro protected es accesible dentro del mismo paquete y en subclases aunque estén en otro paquete.
- d) Un miembro sin modificador (package-private) es accesible desde cualquier clase dentro y fuera del paquete.

40.- ¿Cuál de las siguientes afirmaciones sobre la sobreescritura (override) de métodos en Java es correcta?

- a) Un método final puede ser sobreescrito en subclases directas.
- b) El método sobreescrito en la subclase puede lanzar excepciones comprobadas más amplias que las declaradas en el método de la superclase.
- c) El modificador de acceso del método sobreescrito puede ser más restrictivo que el de la superclase.
- d) Un método sobreescrito en la subclase puede ampliar el modificador de acceso respecto al método de la superclase.