



## Java - Test

**¿Cuál de las siguientes opciones declara correctamente una variable entera en Java?**

- a) int numero = "10";
- b) numero = 10;
- c) int numero = 10;
- d) integer numero = 10;

**Respuesta:** c) int numero = 10;

**¿Qué palabra clave se usa para definir una constante en Java?**

- a) final
- b) const
- c) static
- d) constant

**Respuesta:** a) final

**¿Qué sucede si intentamos asignar un valor decimal a una variable int sin conversión?**

- a) Se redondea automáticamente.
- b) Ocurre un error de compilación.
- c) Se almacena sin la parte decimal.
- d) Se convierte en double automáticamente.

**Respuesta:** b) Ocurre un error de compilación.

**¿Cuál es el tipo de dato adecuado para almacenar un carácter en Java?**

- a) String
- b) char
- c) Character
- d) boolean

**Respuesta:** b) char

**¿Qué clase se usa para leer datos desde la entrada estándar (teclado) en Java?**

- a) Scanner
- b) InputStream
- c) BufferedReader
- d) Console

**Respuesta:** a) Scanner





**¿Cuál de las siguientes líneas permite leer un número entero desde el teclado en Java?**

- a) Scanner sc = new Scanner(System.in); int num = sc.nextInt();
- b) int num = Console.readInt();
- c) Scanner sc = new Scanner(System.in); int num = sc.readInt();
- d) int num = System.in.read();

**Respuesta:** a) Scanner sc = new Scanner(System.in); int num = sc.nextInt();

**¿Cómo se imprime un mensaje en la consola sin salto de línea en Java?**

- a) System.out.print("Hola");
- b) System.out.println("Hola");
- c) System.out.write("Hola");
- d) Console.write("Hola");

**Respuesta:** a) System.out.print("Hola");

**¿Cuál es la estructura de control adecuada para repetir un bloque de código un número determinado de veces?**

- a) if
- b) for
- c) switch
- d) while

**Respuesta:** b) for

**¿Qué palabra clave se usa para salir de un bucle antes de que termine?**

- a) exit
- b) return
- c) break
- d) continue

**Respuesta:** c) break

**¿Qué valor debe tener la condición en un while para que el bucle nunca se ejecute?**

- a) true
- b) false
- c) null
- d) 0

**Respuesta:** b) false



## ¿Qué imprimirá el siguiente código?

```
int x = 5;
if (x > 3) {
    System.out.println("Mayor que 3");
} else if (x > 1) {
    System.out.println("Mayor que 1");
} else {
    System.out.println("Ninguna de las anteriores");
}
```

- a) Mayor que 3
- b) Mayor que 1
- c) Ninguna de las anteriores
- d) Error de compilación

**Respuesta:** a) Mayor que 3

Explica con tus palabras, ¿Cuál es la diferencia entre Print y Println?

Tu respuesta:



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



## ¿Qué imprimirá el siguiente código?

```
public class Test {
    public static void main(String[] args) {
        int a = 10, b = 5;
        a = a + b;
        b = a - b;
        a = a - b;
        System.out.println("a: " + a + ", b: " + b);
    }
}
```

Tu respuesta:

## ¿Cuántas veces se ejecutará la línea "Hola"?

```
public class Test {
    public static void main(String[] args) {
        int i = 1;
        while (i < 5) {
            System.out.println("Hola");
        }
    }
}
```

- a) 4 veces
- b) 5 veces
- c) Infinitamente
- d) Nunca

**Respuesta: c)**

Infinitamente (No hay una instrucción que incremente *i*, por lo que la condición *i < 5* siempre será true).





## ¿Qué sucede si el usuario introduce "10.5" al ejecutar este código?

```
import java.util.Scanner;

public class Test {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Ingrese un número entero: ");
        int num = sc.nextInt();
        System.out.println("Número ingresado: " + num);
    }
}
```

- a) Se imprimirá Número ingresado: 10.5
- b) Se generará una excepción InputMismatchException
- c) Se truncará el número y se imprimirá Número ingresado: 10
- d) El código compilará, pero no imprimirá nada

**Respuesta:** b) Se generará una excepción InputMismatchException porque nextInt() espera un número entero y "10.5" es un número decimal.

## ¿Cuál será la salida de este código si el usuario ingresa "Hola Mundo"?

```
import java.util.Scanner;

public class Test {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Ingrese un texto: ");
        String palabra = sc.next();
        System.out.println("Texto ingresado: " + palabra);
    }
}
```

- a) Texto ingresado: Hola
- b) Texto ingresado: Hola Mundo
- c) Texto ingresado: Mundo
- d) Se generará un error

**Respuesta:** a) Texto ingresado: Hola (next() solo captura la primera palabra antes del primer espacio).





## ¿Cuál será la salida del siguiente código?

```
public class Test {
    public static void main(String[] args) {
        int x = 2;
        switch (x) {
            case 1:
                System.out.println("Uno");
            case 2:
                System.out.println("Dos");
            case 3:
                System.out.println("Tres");
                break;
            default:
                System.out.println("Otro");
        }
    }
}
```

- a) Dos
- b) Dos Tres Otro
- c) Dos Tres
- d) Tres

**Respuesta:** c) Dos Tres  
(Falta break después del  
case 2, por lo que continúa  
ejecutando case 3).

## ¿Qué imprimirá el siguiente código?

```
public class Test {
    public static void main(String[] args) {
        for (int i = 0; i < 5; i += 2) {
            System.out.print(i + " ");
        }
    }
}
```

- a) 0 1 2 3 4
- b) 0 2 4
- c) 0 1 2
- d) 0 2 3 4

**Respuesta:** b) 0 2 4 (i se  
incrementa en 2 en cada  
iteración).

## ¿Qué mostrará el siguiente código?





Cofinanciado por  
la Unión Europea



```
public class Test {
    public static void main(String[] args) {
        int num = 7;
        if (num % 2 == 0)
            System.out.println("Par");
        else
            System.out.println("Impar");
    }
}
```

- a) Par
- b) Impar
- c) Error de compilación
- d) No imprimirá nada

**Respuesta: b) Impar (7 %**  
2 da 1, que no es 0, por  
lo que cae en el else).

### ¿Cuál será la salida de este código?

```
public class Test {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            for (int j = 1; j <= 2; j++) {
                System.out.println("i=" + i + " j=" + j);
            }
        }
    }
}
```

a)

```
ini

i=1 j=1
i=1 j=2
i=2 j=1
i=2 j=2
i=3 j=1
i=3 j=2
```

b)

```
ini

i=1 j=1
i=2 j=2
i=3 j=3
```



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



<p>c)</p> <pre>ini i=1 j=1 i=2 j=1 i=3 j=1</pre>	<p>d)</p> <pre>ini i=1 j=2 i=2 j=2 i=3 j=2</pre>
--	--

**Respuesta:** a) (El bucle anidado imprime todas las combinaciones de i y j).

¿Cuál será la salida de los siguientes códigos y en qué se diferencian?

#### Código 1: Usando while

```
public class Test {
    public static void main(String[] args) {
        int i = 5;
        while (i < 5) {
            System.out.println("Ejecutando while...");
            i++;
        }
    }
}
```

#### Código 2: Usando do-while

```
public class Test {
    public static void main(String[] args) {
        int i = 5;
        do {
            System.out.println("Ejecutando do-while...");
            i++;
        } while (i < 5);
    }
}
```

#### Opciones:

- a) Ambos imprimirán Ejecutando ... una vez.
- b) Solo el while imprimirá Ejecutando while... una vez.



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



- c) Solo el do-while imprimirá Ejecutando do-while... una vez.  
d) Ninguno imprimirá nada.

**Respuesta:**

**c) Solo el do-while imprimirá Ejecutando do-while... una vez.**

- En el primer código (while), la condición `i < 5` es false desde el inicio, por lo que no entra al bucle y no imprime nada.
- En el segundo código (do-while), la instrucción dentro del do se ejecuta al menos una vez antes de evaluar la condición, por lo que imprimirá Ejecutando do-while... una vez antes de salir del bucle.

¿Cuál será la salida de los siguientes códigos y en qué se diferencian?

**Código 1: Usando if-else**

```
public class Test {
    public static void main(String[] args) {
        int num = 3;
        if (num == 1)
            System.out.println("Uno");
        else if (num == 2)
            System.out.println("Dos");
        else if (num == 3)
            System.out.println("Tres");
        else
            System.out.println("Otro");
    }
}
```

**Código 2: Usando switch**

```
public class Test {
    public static void main(String[] args) {
        int num = 3;
        switch (num) {
            case 1:
                System.out.println("Uno");
                break;
            case 2:
                System.out.println("Dos");
                break;
            case 3:
                System.out.println("Tres");
                break;
            default:
                System.out.println("Otro");
        }
    }
}
```

**Opciones:**

- a) Ambos imprimirán Tres.  
b) Solo el if-else imprimirá Tres, el switch imprimirá Otro.  
c) Ambos imprimirán Otro.  
d) Ninguno imprimirá nada.





### Respuesta:

#### a) Ambos imprimirán Tres.

- Ambos códigos evalúan la condición y encuentran que num == 3, por lo que imprimen "Tres".
- La diferencia clave es que en if-else, las condiciones pueden evaluar expresiones más complejas (num > 3, num % 2 == 0, etc.), mientras que switch solo evalúa valores exactos de tipos compatibles (int, char, String, etc.).
- **Nota:** Sin el break, el switch seguiría ejecutando los siguientes case hasta encontrar un break o salir.

¿Qué diferencia hay en la ejecución de estos dos bucles? ¿Cuál imprimirá más veces?

#### Código 1: Usando for

```
public class Test {
    public static void main(String[] args) {
        for (int i = 1; i <= 3; i++) {
            System.out.println("Iteración " + i);
        }
    }
}
```

#### Código 2: Usando while

```
public class Test {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 3) {
            System.out.println("Iteración " + i);
            i++;
        }
    }
}
```

### Opciones:

- Ambos imprimirán exactamente lo mismo.
- for imprimirá más veces que while.
- while imprimirá más veces que for.
- Ninguno imprimirá nada.

### Respuesta:

#### a) Ambos imprimirán exactamente lo mismo.

- Ambos inician i en 1 y lo incrementan hasta 3, por lo que imprimirán:

[Copiar](#)  
[Editar](#)



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



Iteración 1

Iteración 2

Iteración 3

- La diferencia clave es que el **for** agrupa la inicialización, condición e incremento en una sola línea, mientras que el **while** separa la inicialización y el incremento fuera del encabezado.
- En algunos casos, si el incremento de **i++** estuviera dentro de una condición o dentro de un **if**, podrían comportarse de manera diferente.

#### Completar código (Rellenar espacios en blanco)

Completa el código para que imprima los números pares del 1 al 10 usando **for**.

```
public class Test {  
  
    public static void main(String[] args) {  
  
        for (_____ i = _____; i _____; i _____) {  
            if (_____ % 2 == 0) {  
                System.out.println(_____);  
            }  
        }  
    }  
}
```

◊ **Respuesta esperada:**



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



Cofinanciado por  
la Unión Europea



```
for (int i = 1; i <= 10; i++) {
    if (i % 2 == 0) {
        System.out.println(i);
    }
}
```

¿Qué imprimirá el siguiente código?

```
public class Test {
    public static void main(String[] args) {
        int a = 10, b = 20;
        while (a < b) {
            a += 5;
        }
        System.out.println("Valor final de a: " + a);
    }
}
```

Tu respuesta:

◇ **Respuesta esperada:**

Valor **final** de a: **20**

Modifica el código para que en lugar de contar del 1 al 5, cuente del 5 al 1.

```
for (int i = 1; i <= 5; i++) {
    System.out.println(i);
}
```

Tu respuesta:



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG

**12**



◊ Respuesta esperada:

```
for (int i = 5; i >= 1; i--) {
    System.out.println(i);
}
```

¿Qué error tiene el siguiente código?

```
public class Test {
    public static void main(String[] args) {
        String texto = "Hola mundo";
        texto[0] = 'h';
        System.out.println(texto);
    }
}
```

Tu respuesta:



Cofinanciado por  
la Unión Europea



**Completa el siguiente código para recorrer un array de enteros e imprimir cada elemento.**

```
public class Test {
    public static void main(String[] args) {
        int[] numeros = {1, 2, 3, 4, 5};

        for (int i = ____; i ____; i____) {
            System.out.println(____);
        }
    }
}
```

Respuesta esperada:

```
for (int i = 0; i < numeros.length; i++) {
    System.out.println(numeros[i]);
}
```

**¿Cuál será la salida del siguiente código?**

```
public class Test {
    public static void main(String[] args) {
        String s1 = "Hola";
        String s2 = new String("Hola");

        System.out.println(s1 == s2);
        System.out.println(s1.equals(s2));
    }
}
```

Tu respuesta:



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



Cofinanciado por  
la Unión Europea



MINISTERIO  
DE TRABAJO  
Y ECONOMÍA SOCIAL



Fondos Europeos



Junta de Andalucía  
Instituto Andaluz del  
Deporte y la Juventud  
INSTITUTO ANDALUZ DEL DEPORTE Y LA JUVENTUD



Ayuda  
en Acción  
Impulsa Empleo Joven

Respuesta esperada:

false  
true

**Explicación:**

- `s1 == s2` es `false` porque `s1` y `s2` apuntan a diferentes objetos en memoria.
- `s1.equals(s2)` es `true` porque compara el contenido de las cadenas, no las referencias.

**Explica qué hace el siguiente código y cuál será la salida.**

```
public class Test {
    public static void main(String[] args) {
        String[] palabras = {"Java", "es", "genial"};
        String resultado = "";

        for (String palabra : palabras) {
            resultado += palabra + " ";
        }

        System.out.println(resultado.trim());
    }
}
```

◊ **Respuesta esperada:**

- **Salida:** "Java es genial"
- **Explicación:** Recorre un array de Strings y concatena sus elementos en una sola cadena separada por espacios. `.trim()` elimina el espacio final extra.

Tu respuesta:



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.ª ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



Cofinanciado por  
la Unión Europea



### ¿Qué error tiene el siguiente código?

```
public class Test {
    public static void main(String[] args) {
        System.out.println(suma(5, 3));
    }

    public static int suma(int a, int b) {
        int resultado = a + b;
    }
}
```

#### ◊ Respuesta esperada:

- **Error:** El método suma no devuelve ningún valor, aunque está declarado con int.
- **Corrección:** Se debe agregar return resultado;

```
public static int suma(int a, int b) {
    int resultado = a + b;
    return resultado;
}
```

**Completa el siguiente código para que la función multiplica devuelva el producto de dos números.**



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



```

public class Test {
    public static void main(String[] args) {
        int resultado = multiplica(4, 5);
        System.out.println("El resultado es: " + resultado);
    }

    public static _____ multiplica(_____ a, _____ b) {
        return _____;
    }
}

```

◊ **Respuesta esperada:**

```

public static int multiplica(int a, int b) {
    return a * b;
}

```

¿Qué imprimirá el siguiente código?

```

public class Test {
    public static void main(String[] args) {
        System.out.println(mensaje("Juan"));
        System.out.println(mensaje("Ana"));
    }

    public static String mensaje(String nombre) {
        return "Hola, " + nombre + "!";
    }
}

```

Hola, Juan!

Hola, Ana!

◊ **Respuesta esperada:**

**Explicación:**

El método mensaje recibe un nombre como parámetro y devuelve un saludo



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.ª ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



concatenando "Hola, " con el nombre.

### Explica qué hace el siguiente código y cuál será la salida.

```
public class Test {
    public static void main(String[] args) {
        int[] valores = {2, 4, 6};
        System.out.println(sumarArray(valores));
    }

    public static int sumarArray(int[] numeros) {
        int suma = 0;
        for (int num : numeros) {
            suma += num;
        }
        return suma;
    }
}
```

#### ◊ Respuesta esperada:

- **Salida:** 12
- **Explicación:** Se define un array de enteros {2, 4, 6} y se llama a la función sumarArray(), que recorre el array sumando sus elementos.

¿Cuál de las siguientes opciones devuelve la longitud de la cadena String texto = "Java";?

- A) texto.longitud();
- B) texto.length();
- C) texto.size();
- D) texto.count();

**Respuesta correcta: B) texto.length();**





## ¿Qué imprimirá el siguiente código?

```
String s1 = "Hola";
String s2 = "Hola";
String s3 = new String("Hola");

System.out.println(s1 == s2);
System.out.println(s1 == s3);
System.out.println(s1.equals(s3));
```

- A) true false true
- B) false true false
- C) true true true
- D) false false true

**Respuesta correcta: A) true false true**

### Explicación:

- `s1 == s2` es true porque ambos apuntan al mismo objeto en el **pool de Strings**.
- `s1 == s3` es false porque `new String("Hola")` crea un nuevo objeto en memoria.
- `s1.equals(s3)` es true porque compara el contenido, no las referencias.

## ¿Cuál de las siguientes opciones es la forma correcta de declarar un array de enteros en Java?

- A) int[] numeros = new int(5);
- B) int numeros[] = {1, 2, 3, 4, 5};
- C) int[] numeros = new int[5] {1, 2, 3, 4, 5};
- D) int numeros = [1, 2, 3, 4, 5];

**Respuesta correcta: B) int numeros[] = {1, 2, 3, 4, 5};**

## ¿Cuál será la salida del siguiente código?

<pre>int[] valores = {10, 20, 30, 40, 50}; System.out.println(valores[2]);</pre>	<p>A) 10 B) 20 C) 30 D) 40</p>
--	--

- A) 10
- B) 20
- C) 30
- D) 40

**Respuesta correcta: C) 30**

 **Explicación:** Los índices en un array comienzan en **0**, por lo que `valores[2]` devuelve





el tercer elemento (30).

### ¿Qué ocurrirá si ejecutamos el siguiente código?

```
int[] numeros = new int[3];
System.out.println(numeros[5]);
```

- A) Imprimirá 0
- B) Imprimirá null
- C) Ocurrirá un **ArrayIndexOutOfBoundsException**
- D) Imprimirá 5

**Respuesta correcta: C) Ocurrirá un ArrayIndexOutOfBoundsException**

 **Explicación:** El array tiene solo 3 elementos (índices: 0, 1, 2), pero intentamos acceder a numeros[5], que está fuera del rango.

### ¿Cuál de las siguientes declaraciones de métodos en Java es correcta?

- A) public void sumar(int a, int b) { return a + b; }
- B) public static int sumar(int a, int b) { return a + b; }
- C) public int sumar(int a, int b) { System.out.println(a + b); return; }
- D) public void sumar(int a, int b) { return; }

**Respuesta correcta: B) public static int sumar(int a, int b) { return a + b; }**

 **Explicación:**

- Opción A es incorrecta porque un método void no puede devolver un valor.
- Opción C es incorrecta porque return; no tiene sentido en un método con int.
- Opción D es incorrecta porque un método void no necesita return;.

### ¿Qué imprimirá el siguiente código?





```
public class Test {
    public static void main(String[] args) {
        int resultado = sumar(2, 3, 4);
        System.out.println(resultado);
    }

    public static int sumar(int a, int b) {
        return a + b;
    }
}
```

- A) 5
- B) 9
- C) Error de compilación
- D) No imprimirá nada

**Respuesta correcta: C) Error de compilación**

 **Explicación:**

El método sumar(int a, int b) solo recibe **dos parámetros**, pero en sumar(2, 3, 4) estamos enviando **tres**. Esto provoca un error en la compilación.

**¿Qué ocurre si un método en Java no tiene return pero está declarado como int?**

- A) Retorna null automáticamente.
- B) Retorna 0 automáticamente.
- C) Ocurre un error de compilación.
- D) No pasa nada, pero no devuelve un valor.

**Respuesta correcta: C) Ocurre un error de compilación.**

 **Explicación:** Un método con un **tipo de retorno distinto de void** está **obligado** a devolver un valor del tipo especificado. Si falta return, el código no compilará.

**¿Cómo se llama una función dentro de una clase en Java?**

- A) Procedimiento
- B) Método
- C) Función
- D) Subrutina

**Respuesta correcta: B) Método**

 **Explicación:** En Java, las funciones que pertenecen a una clase se llaman **métodos**.





## ¿Para qué se utilizan las excepciones en Java?

- A) Para mejorar el rendimiento del programa.
- B) Para manejar errores en tiempo de ejecución de manera controlada.
- C) Para crear interfaces gráficas de usuario.
- D) Para hacer que el código sea más complejo.

**Respuesta correcta: B) Para manejar errores en tiempo de ejecución de manera controlada.**

## ¿Qué hace un bloque **finally** en las excepciones de Java?

- A) Se ejecuta solo si no hay excepciones.
- B) Se ejecuta solo si ocurre una excepción.
- C) Siempre se ejecuta, independientemente de si hay o no excepciones.
- D) Se ejecuta solo antes del bloque catch.

**Respuesta correcta: C) Siempre se ejecuta, independientemente de si hay o no excepciones.**

## ¿El siguiente código está correcto? Si no, ¿qué está mal?

```
public class Test {
    public static void main(String[] args) {
        try {
            String str = null;
            System.out.println(str.length());
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

- A) Sí, el código está correcto y manejará cualquier excepción.
- B) No, el tipo de excepción debe ser NullPointerException, no Exception.
- C) No, debería haber un bloque finally.
- D) No, falta un bloque catch.

**Respuesta correcta: A) Sí, el código está correcto y manejará cualquier excepción.**





**Explicación:**

El bloque catch (Exception e) maneja todas las excepciones, incluyendo NullPointerException. Aunque sería más específico capturar NullPointerException, este código es válido porque la clase Exception es la superclase de todas las excepciones.

**¿Qué es un objeto en Java?**

- A) Un conjunto de instrucciones que realizan una tarea específica.
- B) Una instancia de una clase.
- C) Un tipo de dato primitivo.
- D) Un tipo de variable que solo puede almacenar enteros.

**Respuesta correcta: B) Una instancia de una clase.**

**Explicación:** Un **objeto** es una instancia de una **clase**. Cada objeto tiene sus propios valores y comportamientos definidos por la clase.

**¿Qué palabra clave se usa para crear una clase en Java?**

- A) class
- B) object
- C) new
- D) create

**Respuesta correcta: A) class**

**Explicación:** En Java, se usa la palabra clave **class** para declarar una clase.

**¿Cómo se crea un objeto en Java?**

- A) Object obj = new Object();
- B) Object obj = new MyClass();
- C) Object obj = new Object("MyClass");
- D) MyClass obj = new Object();

**Respuesta correcta: B) Object obj = new MyClass();**

**Explicación:** Para crear un objeto, se utiliza la palabra clave new junto con el nombre de la clase. **MyClass** es la clase de la que estamos creando un objeto.

**¿Qué ocurre si no proporcionas un constructor en una clase?**

- A) El compilador generará automáticamente un constructor predeterminado.
- B) El programa no compilará.





Cofinanciado por  
la Unión Europea



- C) La clase no podrá tener objetos.
- D) El constructor predeterminado generará un error de ejecución.

**Respuesta correcta: A) El compilador generará automáticamente un constructor predeterminado.**

 **Explicación:** Si no defines un constructor explícitamente, Java proporciona automáticamente un **constructor predeterminado** sin parámetros para la clase.

**¿Cuál de las siguientes afirmaciones es correcta acerca de las clases y los objetos en Java?**

- A) Una clase puede ser instanciada varias veces, pero un objeto solo puede pertenecer a una clase.
- B) Un objeto es una copia exacta de la clase.
- C) Una clase define las características y comportamientos, mientras que un objeto es una instancia concreta de una clase.
- D) Un objeto solo puede tener un método, mientras que una clase puede tener muchos.

**Respuesta correcta: C) Una clase define las características y comportamientos, mientras que un objeto es una instancia concreta de una clase.**

 **Explicación:** Las **clases** definen la estructura y comportamiento, mientras que los **objetos** son instancias de esas clases y tienen datos concretos.

**¿Qué imprimirá el siguiente código?**



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



```

class Animal {
    String nombre;

    Animal(String nombre) {
        this.nombre = nombre;
    }
}

class Perro extends Animal {
    Perro(String nombre) {
        super(nombre);
    }

    void saludar() {
        System.out.println("¡Hola, soy " + nombre + "!");
    }
}

public class Test {
    public static void main(String[] args) {
        Animal a = new Perro("Rex");
        a.saludar();
    }
}

```

- A) ¡Hola, soy Rex!
- B) ¡Hola, soy null!
- C) Error de compilación.
- D) ¡Hola, soy Perro!

**Respuesta correcta: C) Error de compilación.**

 **Explicación:**

El método saludar() no está definido en la clase Animal, y Java no puede llamar un método de una clase derivada en una referencia de tipo base (en este caso Animal). Para evitar el error, el tipo de la referencia a debe ser Perro, no Animal.

**¿Qué es la sobrecarga de métodos en Java?**

- A) Cuando un método tiene el mismo nombre, pero con diferentes tipos de retorno.
- B) Cuando un método tiene el mismo nombre y parámetros en una subclase.
- C) Cuando un método tiene el mismo nombre, pero con diferentes parámetros.
- D) Cuando un método tiene el mismo nombre y la misma cantidad de parámetros.

**Respuesta correcta: C) Cuando un método tiene el mismo nombre, pero con diferentes parámetros.**





 **Explicación:** La **sobrecarga de métodos** ocurre cuando se define más de un método con el mismo nombre en una clase, pero con diferentes tipos o números de parámetros.

### ¿Qué es la herencia en Java?

- A) Es cuando un objeto hereda atributos y comportamientos de una clase.
- B) Es cuando una clase hereda atributos y comportamientos de un objeto.
- C) Es cuando una clase puede invocar los métodos de otra clase.
- D) Es un proceso donde un objeto se convierte en una clase.

 **Respuesta correcta:** A) Es cuando un objeto hereda atributos y comportamientos de una clase.

 **Explicación:** La **herencia** es un principio de la programación orientada a objetos donde una clase hereda atributos y métodos de otra clase.

### ¿Qué ocurrirá si ejecutamos este código?

```
class Vehiculo {
    String marca;

    Vehiculo(String marca) {
        this.marca = marca;
    }

    void mostrarMarca() {
        System.out.println("Marca: " + marca);
    }
}

public class Test {
    public static void main(String[] args) {
        Vehiculo v = new Vehiculo("Toyota");
        v.mostrarMarca();
    }
}
```

- A) Imprimirá "Marca: Toyota"
- B) Error de compilación, falta un constructor.
- C) Imprimirá "Marca: null"
- D) El código no compilará debido a un error en el uso del this.

 **Respuesta correcta:** A) Imprimirá "Marca: Toyota"

 **Explicación:**

El constructor Vehiculo(String marca) se llama correctamente al crear el objeto v y asigna el valor de "Toyota" a la variable marca. Luego, el método mostrarMarca() imprime ese valor.





**¿Desde dónde se puede ejecutar los siguientes métodos y acceder a la variable?**

Ten en cuenta que las llamadas a los métodos en “public static void main(String[] args) ”  
No son todas correctas.

```
class ClaseA {
    public static void metodoPublico() {
        System.out.println("Método público está ejecutado.");
    }

    private int numeroPrivado = 10;

    protected void metodoProtegido() {
        System.out.println("Método protegido ejecutado.");
    }

    int metodoPorDefecto() {
        System.out.println("Método con acceso por defecto ejecutado.");
    }
}

public class Test {
    public static void main(String[] args) {
        ClaseA c = new ClaseA();
        c.metodoPublico();
        c.metodoProtegido();
        c.metodoPorDefecto();
        System.out.println(c.numeroPrivado);
    }
}
```

- A) **metodoPublico()** es accesible desde cualquier clase y **numeroPrivado** es accesible solo dentro de ClaseA.
- B) **metodoPublico()** es accesible desde cualquier clase, **metodoProtegido()** es accesible solo dentro de la misma clase y subclases, y **metodoPorDefecto()** es accesible solo dentro del mismo paquete.
- C) **metodoPublico()** es accesible solo dentro de ClaseA y **metodoPorDefecto()** es accesible desde cualquier clase.
- D) **metodoProtegido()** solo es accesible dentro del mismo paquete y por subclases.

**Respuestas correctas:**

- A) **metodoPublico()** es accesible desde cualquier clase y **numeroPrivado** es accesible solo dentro de ClaseA.
- B) **metodoPublico()** es accesible desde cualquier clase, **metodoProtegido()** es accesible solo dentro de la misma clase y subclases, y **metodoPorDefecto()** es accesible solo dentro del mismo paquete.





### Explicación de las respuestas:

- **public static void metodoPublico():** Al ser **public**, este método es accesible desde cualquier clase en cualquier paquete, incluso si es estático (static).
- **private int numeroPrivado:** Al ser **private**, esta variable solo es accesible dentro de ClaseA. No puede ser accesada desde fuera de la clase.
- **protected void metodoProtegido():** El modificador **protected** hace que el método sea accesible dentro de la clase que lo declara, y también en sus subclases, incluso si estas están en otro paquete.
- **int metodoPorDefecto():** Al no tener modificador de acceso, el método tiene **acceso por defecto**, lo que significa que solo es accesible dentro del mismo paquete.

### ¿Desde dónde se puede acceder al siguiente método y variable?

```
class MiClase {
    public int numeroPublico = 5;
    private int numeroPrivado = 10;

    public void metodoPublico() {
        System.out.println("Método público ejecutado.");
    }

    private void metodoPrivado() {
        System.out.println("Método privado ejecutado.");
    }
}

public class Test {
    public static void main(String[] args) {
        MiClase obj = new MiClase();
        obj.metodoPublico();
        // obj.metodoPrivado(); // Error de compilación
        System.out.println(obj.numeroPublico);
        // System.out.println(obj.numeroPrivado); // Error de compilación
    }
}
```

- A) El **metodoPublico()** y la **variable numeroPublico** son accesibles desde cualquier clase, mientras que el **metodoPrivado()** y la **variable numeroPrivado** son solo accesibles dentro de la clase MiClase.
- B) Solo **metodoPrivado()** y **numeroPrivado** son accesibles dentro de MiClase.
- C) **metodoPublico()** es accesible solo dentro de MiClase, y **numeroPublico** solo dentro de la clase Test.



D) **metodoPublico()** y **numeroPublico** son accesibles solo dentro del paquete de MiClase.

Respuesta correcta:

**A) El método metodoPublico() y la variable numeroPublico son accesibles desde cualquier clase, mientras que el metodoPrivado() y la variable numeroPrivado son solo accesibles dentro de la clase MiClase.**

**Explicación de la respuesta:**

- **public int numeroPublico:** La **variable pública** es accesible desde cualquier clase, incluso fuera del paquete de MiClase.
- **private int numeroPrivado:** La **variable privada** solo es accesible dentro de la clase MiClase, no desde fuera de ella.
- **public void metodoPublico():** El **método público** es accesible desde cualquier clase.
- **private void metodoPrivado():** El **método privado** solo es accesible dentro de la clase MiClase

**¿Para qué se utiliza la anotación `@Override` en Java?**

- A) Para indicar que un método no puede ser modificado.  
 B) Para que el compilador verifique que un método está siendo sobrescrito correctamente.  
 C) Para declarar un método que puede ser ejecutado en cualquier clase.  
 D) Para crear un nuevo método en una clase.

**Respuesta correcta: B) Para que el compilador verifique que un método está siendo sobrescrito correctamente.**

 **Explicación:**

La anotación `@Override` se utiliza para indicar que un método en una clase hija está sobrescribiendo (modificando) un método de la clase padre. El compilador verificará que realmente estás sobrescribiendo un método de la clase base y no creando un método nuevo por error.

**¿Qué hace el modificador `extends` en Java?**



- A) Define que una clase es una interfaz.
- B) Permite que una clase herede los métodos y atributos de otra clase.
- C) Define un tipo de variable que puede almacenar clases.
- D) Es un modificador de acceso para restringir la visibilidad de un método.

**Respuesta correcta: B) Permite que una clase herede los métodos y atributos de otra clase.**

 **Explicación:**

El modificador **extends** se usa en Java para que una **subclase** herede los métodos y atributos de una **superclase**. Esto permite la reutilización del código y la creación de jerarquías de clases.

**¿Cuál es la diferencia entre sobrescribir y sobrecargar un método?**

- A) **Sobrescribir** un método cambia su implementación en la clase hija, mientras que **sobrecargar** un método crea métodos con el mismo nombre pero diferentes parámetros en la misma clase.
- B) **Sobrescribir** un método crea una nueva versión del método, mientras que **sobrecargar** un método lo elimina de la clase base.
- C) **Sobrescribir** es cuando un método cambia su tipo de retorno, mientras que **sobrecargar** es cuando un método tiene el mismo nombre pero diferente visibilidad.
- D) No hay diferencia, ambos significan lo mismo en Java.

**Respuesta correcta: A) Sobrescribir un método cambia su implementación en la clase hija, mientras que sobrecargar un método crea métodos con el mismo nombre pero diferentes parámetros en la misma clase.**

 **Explicación:**

- **Sobrescribir** (override) se refiere a cuando una clase hija proporciona una implementación diferente de un método que ya existe en la clase padre.
- **Sobrecargar** (overload) es cuando una clase tiene varios métodos con el mismo nombre, pero con diferentes parámetros (número o tipo de parámetros).

**¿Cuál es el resultado de este código?**





```

class Animal {
    public void hacerSonido() {
        System.out.println("Sonido de animal");
    }
}

class Perro extends Animal {
    @Override
    public void hacerSonido() {
        System.out.println("Guau");
    }

    public void hacerSonido(int veces) {
        for (int i = 0; i < veces; i++) {
            System.out.println("Guau");
        }
    }
}

public class Test {
    public static void main(String[] args) {
        Perro p = new Perro();
        p.hacerSonido();
        p.hacerSonido(3);
    }
}

```

- A) Imprime "Sonido de animal" y luego "Guau" tres veces.  
 B) Imprime "Guau" una vez y luego "Guau" tres veces.  
 C) Imprime "Guau" una vez y lanza un error.  
 D) Imprime "Sonido de animal" dos veces.

**Respuesta correcta: B) Imprime "Guau" una vez y luego "Guau" tres veces.**

 **Explicación:**

- El método **hacerSonido()** de la clase Perro sobrescribe el método de la clase Animal.
- El método **hacerSonido(int veces)** de la clase Perro es una **sobrecarga** del método, ya que tiene el mismo nombre pero con parámetros diferentes.

**¿Qué imprimirá el siguiente código?**



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
 DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
 AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
 C/DOS ACERAS 23, 29012  
 MÁLAGA | (+34) 952 300 500  
 ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
 TR.ª ALAMEDA DE SOLANO, 32, 11130  
 CHICLANA DE LA FRONTERA | (+34) 956 900 312  
 CHICLANA@ARRABALEMPLEO.ORG



```

class Vehiculo {
    public void arrancar() {
        System.out.println("Vehículo arrancando...");
    }
}

class Coche extends Vehiculo {
    @Override
    public void arrancar() {
        System.out.println("Coche arrancando...");
    }
}

public class Test {
    public static void main(String[] args) {
        Vehiculo v = new Coche();
        v.arrancar();
    }
}

```

- A) "Vehículo arrancando..."
- B) "Coche arrancando..."
- C) Error de compilación.
- D) No imprime nada.

**Respuesta correcta: B) "Coche arrancando..."**

 **Explicación:**

El método **arrancar()** de la clase **Coche** sobrescribe el método de la clase **Vehiculo**, por lo que cuando se crea un objeto de tipo **Coche** y se invoca el método **arrancar()** a través de una referencia de tipo **Vehiculo**, se ejecuta la implementación sobrescrita en la clase **Coche**.

**¿Cuál es la principal diferencia entre una **List** y un **Set** en Java?**

- A) Una **List** puede contener elementos duplicados, mientras que un **Set** no permite duplicados.
- B) Una **List** no permite elementos nulos, pero un **Set** sí.
- C) Una **List** es más rápida que un **Set** en la búsqueda de elementos.
- D) Un **Set** mantiene el orden de inserción de los elementos, mientras que una **List** no lo hace.

**Respuesta correcta: A) Una List puede contener elementos duplicados, mientras que un Set no permite duplicados.**

**¿Qué tipo de colección deberías usar si necesitas almacenar un grupo de objetos que no pueden repetirse?**



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.<sup>a</sup> ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG



- A) List
- B) Set
- C) Queue
- D) Map

**Respuesta correcta: B) Set**

 **Explicación:**

El **Set** es una colección que no permite elementos duplicados. Es ideal cuando se necesita garantizar que no haya repeticiones.

¿Qué hace el método **put(key, value)** en un **Map** en Java?

- A) Asocia una clave con un valor, añadiendo un nuevo par clave-valor al mapa o actualizando el valor si la clave ya existe.
- B) Añade un valor al mapa, sin necesidad de asociarlo con una clave.
- C) Elimina una clave y su valor asociado del mapa.
- D) Muestra todos los elementos almacenados en el mapa.

**Respuesta correcta: A) Asocia una clave con un valor, añadiendo un nuevo par clave-valor al mapa o actualizando el valor si la clave ya existe.**

¿Qué tipo de colección puedes usar para almacenar pares de claves y valores en Java?

- A) List
- B) Set
- C) Map
- D) Queue

**Respuesta correcta: C) Map**

 **Explicación:**

El **Map** almacena elementos en pares de clave-valor, donde cada clave está asociada a un único valor.

¿Cuál de las siguientes opciones es verdadera respecto a las colecciones en Java?

- A) Un **List** puede contener elementos duplicados y mantiene el orden de inserción.
- B) Un **Set** puede contener elementos duplicados y no mantiene el orden.
- C) Un **Map** no puede contener claves duplicadas, pero sí valores duplicados.
- D) Un **Queue** no permite la inserción de elementos duplicados.





- Respuesta correcta:** A) Un List puede contener elementos duplicados y mantiene el orden de inserción.  
 C) Un Map no puede contener claves duplicadas, pero sí valores duplicados.

¿Cuál de los siguientes métodos se utiliza para comprobar si un elemento está presente en un Set?

- A) containsKey()
- B) contains()
- C) get()
- D) indexOf()

- Respuesta correcta:** B) contains()

 **Explicación:**

El método **contains()** se utiliza para verificar si un elemento existe en un Set.

¿Qué ocurre si intentas agregar un elemento duplicado a un Set en Java?

- A) El elemento se añade al conjunto, pero con un valor diferente.
- B) El conjunto lanza una excepción.
- C) El elemento no se añade al conjunto y no genera ningún error.
- D) El conjunto elimina el elemento duplicado automáticamente.

- Respuesta correcta:** C) El elemento no se añade al conjunto y no genera ningún error.

 **Explicación:**

En un Set, los elementos duplicados no se permiten, pero la operación de añadir un elemento duplicado simplemente no lo agregará, sin generar ningún error.

¿Cuál de las siguientes opciones describe mejor la función de un Map?

- A) Almacena elementos en un orden aleatorio.
- B) Asocia una clave única a un valor específico.
- C) Almacena los elementos de manera ordenada y sin duplicados.
- D) Permite una sola clave, pero múltiples valores para esa clave.

- Respuesta correcta:** B) Asocia una clave única a un valor específico.

¿Qué método se utiliza en un List para obtener un elemento en una posición específica?

- A) get(index)
- B) put(index, value)





- C) **add(value)**
- D) **contains(value)**

**Respuesta correcta: A) get(index)**

**Explicación:**

El método **get(index)** se usa para obtener el elemento en una posición específica de la lista.

**¿Cuál es la diferencia principal entre un HashSet y un TreeSet en Java?**

- A) Un **HashSet** mantiene el orden de inserción de los elementos, mientras que un **TreeSet** no lo hace.
- B) Un **HashSet** no garantiza ningún orden, mientras que un **TreeSet** mantiene los elementos en orden ascendente.
- C) Un **TreeSet** permite elementos duplicados, mientras que un **HashSet** no.
- D) Un **HashSet** es más rápido que un **TreeSet** para operaciones de búsqueda.

**Respuesta correcta: B) Un HashSet no garantiza ningún orden, mientras que un TreeSet mantiene los elementos en orden ascendente.**

**¿Qué método se usa para establecer una conexión con la base de datos en JDBC?**

- A) **connect()**
- B) **getConnection()**
- C) **createConnection()**
- D) **establishConnection()**

**Respuesta correcta: B) getConnection()**

**Explicación:**

El método **getConnection()** de la clase DriverManager se utiliza para establecer una conexión con la base de datos.

**¿Qué tipo de objeto se utiliza para ejecutar una consulta SQL en JDBC?**

- A) **ResultSet**
- B) **PreparedStatement**
- C) **Statement**
- D) **Connection**

**Respuesta correcta: C) Statement**



**Explicación:**

El objeto **Statement** se utiliza para ejecutar consultas SQL simples en la base de datos.

**¿Qué método se usa para ejecutar una consulta de selección (SELECT) en JDBC?**

- A) executeUpdate()
- B) executeQuery()
- C) execute()
- D) runQuery()

**Respuesta correcta: B) executeQuery()**

**Explicación:**

El método **executeQuery()** se usa para ejecutar consultas que devuelven resultados, como las consultas **SELECT**.

**¿Qué tipo de objeto devuelve el método executeQuery() al ejecutar una consulta?**

- A) Statement
- B) Connection
- C) ResultSet
- D) PreparedStatement

**Respuesta correcta: C) ResultSet**

**Explicación:**

El método **executeQuery()** devuelve un objeto **ResultSet**, que contiene los resultados de la consulta **SELECT**.

**¿Qué clase de JDBC se utiliza para ejecutar consultas SQL preparadas con parámetros?**

- A) Statement
- B) PreparedStatement
- C) CallableStatement
- D) ResultSet

**Respuesta correcta: B) PreparedStatement**

**Explicación:**

El objeto **PreparedStatement** se usa para ejecutar consultas SQL que pueden tener parámetros. Es más eficiente y seguro que el **Statement** en muchos casos.



Cofinanciado por  
la Unión Europea



MINISTERIO  
DE TRABAJO  
Y ECONOMÍA SOCIAL



Fondos Europeos



Junta de Andalucía  
Instituto Andaluz de  
Investigación y  
Difusión del  
Empleo Joven



Ayuda  
en Acción  
Impulsa Empleo Joven



REGISTRO NACIONAL DE ASOCIACIONES N°611922  
DECLARADA ENTIDAD DE UTILIDAD PÚBLICA ESTATAL  
AGENCIA DE COLOCACIÓN: ID 0100000017

CENTRO EN MÁLAGA  
C/DOS ACERAS 23, 29012  
MÁLAGA | (+34) 952 300 500  
ARRABAL@ARRABALEMPLEO.ORG

CENTRO EN CÁDIZ  
TR.º ALAMEDA DE SOLANO, 32, 11130  
CHICLANA DE LA FRONTERA | (+34) 956 900 312  
CHICLANA@ARRABALEMPLEO.ORG

37