CESUR
Tu Centro Oficial de FP

EXAMEN PARCIAL – UNIDAD 7		
TRIMESTRE: SEGUNDO		**Fecha:**
CICLO: Desarrollo de Aplicaciones Web *MÓDULO: Programación*	*CURSO: 1º* *Turno: Mañana*	*CALIFICACIÓN:*
Nombre:	***Apellidos:***	

Instrucciones: *Esta prueba tiene como finalidad evaluar los aprendizajes de Programación. Lee atentamente y responde escribiendo el código más adecuado.*

Si las instrucciones no se siguen como se especifican el examen no será evaluado

PARTE PRÁCTICA. TIPO A.

- El examen práctico tiene una puntuación máxima de 10 puntos.
- Para superar la parte práctica se requiere alcanzar un mínimo de 5 puntos.

1. **(5 puntos)** Se debe desarrollar un programa en Java que realice las siguientes tareas:
   a) Generación de un array de números aleatorios enteros. Se realizará en un **método estático** generarArrayAleatorio() que devolverá el array dentro de la clase **BuscadorArray**:
      - Se generará un array cuyo tamaño será un número aleatorio entre 50 y 100.
      - Cada elemento del array será un número entero aleatorio entre 1 y 100, asegurando que no se repitan valores.
   b) Búsqueda de valores en el array. Se realizará en un **método estático** buscarElemento(int[] array, int valor) dentro de la clase **BuscadorArray**, que devolverá la posición del array en la que se encuentra el número. Si el número **no está en el array**, se lanzará la excepción **ElementoNoEncontradoException**:
      - Se intentará buscar hasta 5 números en el array generado, que se pedirán por teclado. Se debe comprobar que el número introducido esté entre 1 y 100.
      - Para cada número, se verificará si está presente en el array.
      - Si después de 5 intentos ningún número fue encontrado, se mostrará un mensaje indicando que ninguno de los números buscados estaba presente en el array.
   c) Uso de Excepciones:
      - Se debe definir una excepción personalizada llamada **ElementoNoEncontradoException**, que será lanzada cuando un número no se encuentre en el array.

Prueba tu programa con la clase Test:

```
public class Test {
    public static void main(String[] args) {
        Random random = new Random();
        Scanner sc = new Scanner(System.in);
        int valorBuscado = -1;
        // Generar el array de números aleatorios
        int[] numeros = BuscadorArray.generarArrayAleatorio();
        // Intentar buscar 5 números
        boolean encontrado = false;
        int intento=1;
        do{
            do {
                System.out.print("Escribe un numero (1-100): ");
                valorBuscado = sc.nextInt();
            }while(valorBuscado<1 || valorBuscado>100);
            System.out.println("Intento " + intento + ": Buscando el número " + valorBuscado);
            try {
                int indice = BuscadorArray.buscarElemento(numeros, valorBuscado);
                System.out.println("¡Éxito! El número " + valorBuscado + " se encuentra en el índice: " + indice);
                encontrado = true;
            } catch (ElementoNoEncontradoException e) {
                System.out.println("Excepción: " + e.getMessage());
            }
            intento++;
        }while (!encontrado && intento<=5);
        if (!encontrado) {
            System.out.println("No se encontró ninguno de los 5 números buscados.");
        }
        System.out.println("Números: "+Arrays.toString(numeros));
    }
}
```

Ejemplo de ejecución:

```
Tamaño del array generado aleatoriamente: 81
Escribe un numero (1-100): 230
Escribe un numero (1-100): 54
Intento 1: Buscando el número 54
Excepción: El número 54 no se encuentra en el array.
Escribe un numero (1-100): 67
Intento 2: Buscando el número 67
Excepción: El número 67 no se encuentra en el array.
Escribe un numero (1-100): 50
Intento 3: Buscando el número 50
¡Éxito! El número 50 se encuentra en el índice: 53
Números: [99, 82, 52, 93, 74, 86, 7, 32, 60, 18, 8, 62, 3, 16, 19, 56, 59, 70, 35, 87, 64, 29, 25, 48, 28, 36, 72, 17, 21, 53, 46, 95, 97, 3
```

2. **(5 puntos)** Desarrolla un sistema bancario con la clase **CuentaBancaria**, que permita realizar depósitos y reintegros de dinero. El sistema bancario tendrá las siguientes funcionalidades:
    - Método **depositar(double cantidad)**: Permite depositar dinero; si la cantidad es cero o negativa se lanza la excepción DepositoInvalidoException.
    - Método **reintegro(double cantidad)**: Permite retirar dinero; si la cantidad es cero o negativa se lanza la excepción ReintegroInvalidoException y, si la cantidad supera el saldo disponible, se lanza la excepción SaldoInsuficienteException.
    - Método **transferir(Cuenta destino, double cantidad)**: Permite transferir dinero a otra cuenta, realizando internamente un reintegro de la cuenta origen y un depósito en la cuenta destino.
    - Historial de transacciones: Se registra cada operación (depósito, retiro y transferencia) en un array de transacciones (de capacidad fija, por ejemplo, 100 transacciones). Crear una clase **Transaccion** que tenga como atributos el tipo de operación, la cantidad (en euros) y la fecha y hora en que se realizó. Solo tendrá el constructor y el método toString().

Para almacenar la fecha utilizaremos de la clase LocalDateTime el método LocalDateTime.now()

El método toString() de la clase Transaccion, puede ser este:

```
@Override
public String toString() {
    // Formateamos la fecha y la cantidad para su visualización
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
    return fechaHora.format(formatter) + " - " + tipo + " de " + String.format("%.2f€", cantidad);
}
```

Prueba tu programa con la clase Test:

```
public class Test {
    public static void main(String[] args) {
        // Creación de dos cuentas bancarias
        CuentaBancaria cuenta1 = new CuentaBancaria( titular: "Ana García",  saldoInicial: 1000);
        CuentaBancaria cuenta2 = new CuentaBancaria( titular: "Luis Martínez",  saldoInicial: 500);
        try {
            // Realizar operaciones:
            cuenta1.depositar( cantidad: 300);          // Depósito de 300€ en la cuenta1
            cuenta1.reintegrar( cantidad: 200);         // Reintegro de 200€ en la cuenta1
            cuenta1.transferir(cuenta2,  cantidad: 400);  // Transferencia de 400€ de cuenta1 a cuenta2

            // Probamos con valores inválidos para observar el manejo de excepciones:
            //cuenta1.depositar(0);
            //cuenta1.reintegrar(-50);
            //cuenta1.reintegrar(2000);
        } catch (DepositoInvalidoException | ReintegroInvalidoException | SaldoInsuficienteException e) {
            System.out.println("Error: " + e.getMessage());
        }
        System.out.println();
        System.out.printf("Saldo de %s : %.2f€%n", cuenta1.getTitular(),cuenta1.getSaldo());
        System.out.printf("Saldo de %s : %.2f€%n", cuenta2.getTitular(),cuenta2.getSaldo());
        System.out.println();
        // Mostrar el historial de transacciones de cada cuenta
        cuenta1.mostrarHistorial();
        System.out.println();
        cuenta2.mostrarHistorial();
    }
}
```

Ejemplo de ejecución:

```
Depósito de 300,00€ realizado con éxito.
Reintegro de 200,00€ realizado con éxito.
Reintegro de 400,00€ realizado con éxito.
Depósito de 400,00€ realizado con éxito.
Transferencia de 400,00€ de Ana García a Luis Martínez realizada con éxito.

Saldo de Ana García : 700,00€
Saldo de Luis Martínez : 900,00€

Historial de transacciones de Ana García:
2025-02-12 09:26:42 - Depósito de 300,00€
2025-02-12 09:26:42 - Reintegro de 200,00€
2025-02-12 09:26:42 - Reintegro de 400,00€
2025-02-12 09:26:42 - Transferencia enviada de 400,00€

Historial de transacciones de Luis Martínez:
2025-02-12 09:26:42 - Depósito de 400,00€
2025-02-12 09:26:42 - Transferencia recibida de 400,00€
```