Actividades de la Unidad 11: Colecciones.

Actividades propuestas. Realizar las siguientes **12.03, 12.04 y 12.05.**

Actividad propuesta 12.3

Repite la Actividad resuelta 12.6 usando un iterador para eliminar los elementos cuyo valor es 5.

```java
package com.mycompany.ap12_3;

import java.util.*;

public class Ap12_3 {

    public static void main(String[] args) {
        Collection<Integer> lista = new ArrayList<>();

        int n;
        for (int i = 0; i < 100; i++) {
            lista.add((int) (Math.random() * 10 + 1));
        }
        System.out.println("Lista inicial:\n" + lista);
        Iterator<Integer> iterador = lista.iterator();
        while (iterador.hasNext()) {
            n = iterador.next();
            if (n == 5) {
                iterador.remove();
            }
        }
        System.out.println("Lista inicial:\n" + lista);
    }
}

```

```
--- exec:3.1.0:exec (default-cli) @ Ap12_3 ---
Lista inicial:
[8, 8, 5, 1, 5, 6, 5, 4, 4, 8, 3, 8, 2, 8, 8, 6, 4, 5, 3, 2, 4, 5, 3, 7, 9, 2, 3, 6, 9, 7, 6, 10, 1, 4, 6, 5, 1, 1, 4, 9, 4, 6, 7, 3, 5, 7, 4, 9, 9, 10, 4, 6, 10, 9, 4, 2, 9, 2, 6, 4, 2, 6, 1, 5,
Lista inicial:
[8, 8, 1, 6, 4, 4, 8, 3, 8, 2, 8, 8, 6, 4, 3, 2, 4, 3, 7, 9, 2, 3, 6, 9, 7, 6, 10, 1, 4, 6, 1, 1, 4, 9, 4, 6, 7, 3, 7, 4, 9, 9, 10, 4, 6, 10, 9, 4, 2, 9, 2, 6, 4, 2, 6, 1, 7, 1, 10, 6, 6, 6, 1, 9,
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
```

```
5, 7, 1, 10, 6, 6, 6, 5, 1, 9, 8, 8, 4, 2, 5, 9, 2, 6, 10, 9, 6, 5, 9, 10, 9, 6, 8, 9, 7, 7, 8, 3, 3, 9, 5, 1, 6]

, 9, 8, 8, 4, 2, 9, 2, 6, 10, 9, 6, 9, 10, 9, 6, 8, 9, 7, 7, 8, 3, 3, 9, 1, 6]
```

Actividad propuesta 12.4

Implementa una aplicación donde se piden por consola números reales hasta que se introduce un 0. A medida que se leen del teclado, los valores positivos se insertan en una colección y los negativos en otra. Al final, se muestran ambas colecciones y las sumas de los elementos de cada una de ellas. Por último, se eliminan de ambas todos los valores que sean mayores que 10 o menores de −10 y se vuelven a mostrar.

```java
package com.mycompany.ap12_4;

import java.util.*;

public class Ap12_4 {

    public static void main(String[] args) {
        int num, suma = 0;
        Scanner sc = new Scanner(source:System.in);
        List<Integer> positivos = new ArrayList<>();
        List<Integer> negativos = new ArrayList<>();
        do {
            System.out.println(x: "Introduzca un numero, o 0 para terminar:");
            num = sc.nextInt();
            if (num > 0) {
                positivos.add(e: num);
            } else if (num != 0) {
                negativos.add(e: num);
            }

        } while (num != 0);
        System.out.println("--Positivos:\n" + positivos);
        System.out.println("--Negativos:\n" + negativos);
        for (int numero : positivos) {
            suma += numero;
        }
        System.out.println("---Suma de Positivos:\n" + suma);
        suma = 0;
        for (int numero : negativos) {
            suma += numero;
        }
        System.out.println("---Suma de Negativos:\n" + suma);
    }
}

```

```
--- exec:3.1.0:exec (default-cli) @ Ap12_4 ---
Introduzca un numero, o 0 para terminar:
5
Introduzca un numero, o 0 para terminar:
-8
Introduzca un numero, o 0 para terminar:
12
Introduzca un numero, o 0 para terminar:
-74
Introduzca un numero, o 0 para terminar:
3
Introduzca un numero, o 0 para terminar:
50
Introduzca un numero, o 0 para terminar:
0
--Positivos:
[5, 12, 3, 50]
--Negativos:
[-8, -74]
---Suma de Positivos:
70
---Suma de Negativos:
-82
------------------------------------------------------------
BUILD SUCCESS
```

Actividad propuesta 12.5

A partir de `conjuntoClientes` del ejemplo, crea otro conjunto con los mismos elementos ordenados por edad y otro con los clientes ordenados por nombre.

```java
package com.mycompany.ap12_5;

import java.util.*;

public class Ap12_5 {

    public static void main(String[] args) {
        Set<Cliente> clientes = new TreeSet<>();
        Comparator<Cliente> porEdad = new Comparator<Cliente>() {
            @Override
            public int compare(Cliente uno, Cliente otro) {
                return Integer.compare(x: uno.edad(), y: otro.edad());
            }
        };
        Comparator<Cliente> porNombre = new Comparator<Cliente>() {
            @Override
            public int compare(Cliente uno, Cliente otro) {
                return uno.nombre.compareTo(anotherString:otro.nombre);
            }
        };
        clientes.add(new Cliente(dni:"111", nombre:"Marta", fechaNacimiento: "12/02/2000"));
        clientes.add(new Cliente(dni:"115", nombre:"Jorge", fechaNacimiento: "16/03/1995"));
        clientes.add(new Cliente(dni:"112", nombre:"Carlos", fechaNacimiento: "01/10/2002"));
        System.out.println("--Clientes ordenados por DNI:" + clientes);
        Set<Cliente> clientesPorEdad = new TreeSet<>(comparator:porEdad);
        clientesPorEdad.addAll(c: clientes);
        System.out.println("--Clientes ordenados por edad:" + clientesPorEdad);
        Set<Cliente> clientesPorNombre = new TreeSet<>(comparator:porNombre);
        clientesPorNombre.addAll(c: clientes);
        System.out.println("--Clientes ordenados por nombre:" + clientesPorNombre);
    }
}
```

```java
package com.mycompany.ap12_5;

import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.temporal.ChronoUnit;

public class Cliente implements Comparable<Cliente>{
    String dni, nombre;
    LocalDate fechaNacimiento;

    public Cliente(String dni, String nombre, String fechaNacimiento) {
        this.dni = dni;
        this.nombre = nombre;
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern(pattern: "dd/MM/yyyy");
        this.fechaNacimiento = LocalDate.parse(text: fechaNacimiento, formatter);
    }

    int edad(){
        return (int)fechaNacimiento.until(endExclusive: LocalDate.now(), unit: ChronoUnit.YEARS);
    }
    @Override
    public boolean equals(Object obj) {
        return dni.equals(anObject: ((Cliente) obj).dni);
    }

    @Override
    public int compareTo(Cliente otro){
        return dni.compareTo(anotherString:otro.dni);
    }

    @Override
    public String toString() {
        return "\n***Cliente***:" + "\n-dni=" + dni + "\n-nombre=" + nombre
                + "\n-edad=" + edad() + "\n";
    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Ap12_5 ---
--Clientes ordenados por DNI:[
***Cliente***:
-dni=111
-nombre=Marta
-edad=24
,
***Cliente***:
-dni=112
-nombre=Carlos
-edad=21
,
***Cliente***:
-dni=115
-nombre=Jorge
-edad=28
]
--Clientes ordenados por edad:[
```

```
--Clientes ordenados por edad:[
***Cliente***:
-dni=112
-nombre=Carlos
-edad=21
,
***Cliente***:
-dni=111
-nombre=Marta
-edad=24
,
***Cliente***:
-dni=115
-nombre=Jorge
-edad=28
]
--Clientes ordenados por nombre:[
```

```
--Clientes ordenados por nombre:[
***Cliente***:
-dni=112
-nombre=Carlos
-edad=21
,
***Cliente***:
-dni=115
-nombre=Jorge
-edad=28
,
***Cliente***:
-dni=111
-nombre=Marta
-edad=24
]
------------------------------------------------------------------------
BUILD SUCCESS
```

Actividades de aplicación. Realizar las siguientes **12.16, 12.17, 12.18, 12.19 y 12.24.**

12.16. Implementa una aplicación que gestione los socios de un club usando la clase `Socio` implementada en la Actividad resuelta 12.11. En particular, se deberán ofrecer las opciones de alta, baja y modificación de los datos de un socio. Además, se listarán los socios por nombre o por antigüedad en el club.

```java
package com.mycompany.aa12_16;
import java.util.*;

public class Aa12_16 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(source:System.in);
        int opcion;
        String dni, nombre, fecha;
        Socio socio;
        Set<Socio> socios = new TreeSet<>();
        do {
            menu();
            opcion = sc.nextInt();
            sc.skip(pattern: "\n");
            switch (opcion) {
                case (1) -> { //Alta
                    System.out.println(x: "Introduzca dni (no se podrá modificar):");
                    dni = sc.nextLine();
                    System.out.println(x: "Nombre:");
                    nombre = sc.nextLine();
                    System.out.println(x: "Fecha alta(dd/MM/yyyy):");
                    fecha = sc.nextLine();
                    socios.add(new Socio(dni, nombre, alta: fecha));
                }
                case (2) -> { //modificar
                    System.out.println(x: "Introduzca dni socio a modificar;");
                    dni = sc.next();
                    for (Socio c : socios) {
                        if (c.dni.equals(anObject: dni)) {
                            sc.skip(pattern: "\n");
                            System.out.println("Socio encontrado\n"
                                    + "Introduzca nuevo nombre:");
                            c.nombre = sc.nextLine();
                            System.out.println(x: "Fecha alta(dd/MM/yyyy):");
                            c.setfechaAlta(alta: sc.nextLine());
                        }
                    }
                }
                case (3) -> { //dar de baja
                    System.out.println(x: "Introduzca id socio a dar de baja;");
                    dni = sc.next();
                    Iterator<Socio> iterador = socios.iterator();
                    while (iterador.hasNext()) {
                        socio = iterador.next();
                        if (socio.equals(new Socio(dni, nombre:"", alta: ""))) {
                            iterador.remove();
                        }
                    }
                    System.out.println(x: "\nSocio eliminado\n");
                }
                case (4) -> { //Listar por nombre
                    Comparator<Socio> porNombre = new Comparator<Socio>() {
                        @Override
                        public int compare(Socio uno, Socio otro) {
                            return uno.nombre.compareTo(anotherString:otro.nombre);
                        }
                    };
                    Set<Socio> sociosPorNombre = new TreeSet<>(comparator:porNombre);
                    sociosPorNombre.addAll(c: socios);
                    System.out.println("--Socios ordenados por nombre:" + sociosPorNombre);
                }
                case (5) -> { //Listar por antigüedad
                    Comparator<Socio> porAntiguedad = new Comparator<Socio>() {
                        @Override
                        public int compare(Socio uno, Socio otro) {
                            return uno.fechaAlta.compareTo(other: otro.fechaAlta);
                        }
                    };
                    Set<Socio> sociosPorAntiguedad = new TreeSet<>(comparator:porAntiguedad);
                    sociosPorAntiguedad.addAll(c: socios);
                    System.out.println("--Socios ordenados por nombre:" + sociosPorAntiguedad);
                }
                case (0) -> {
                }
                default -> {
                    System.out.println(x: "Opción errónea");
                }
            }
        } while (opcion != 0);
    }

    private static void menu() {
        System.out.println(x: """
                              ********Men\u00fa Socios********
                              1 Alta socio
                              2 Modificar socio
                              3 Dar de baja socio
                              4 Listar socios por nombre
                              5 Listar socios por antigüedad
                              0 Salir:""");
    }
}
```

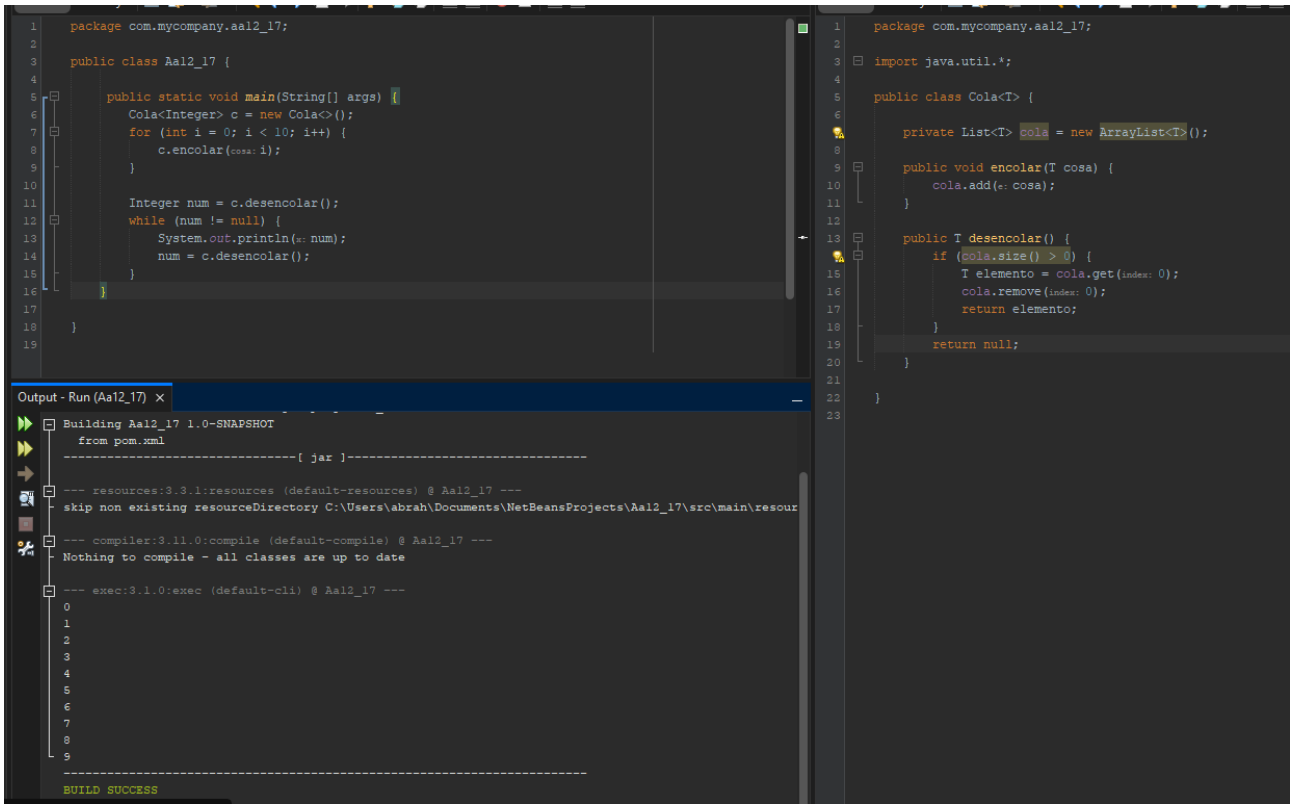
```java
package com.mycompany.aa12_16;

import java.io.Serializable;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.time.temporal.ChronoUnit;

public class Socio implements Comparable<Socio>, Serializable {

    String dni;
    String nombre;
    LocalDate fechaAlta;

    public Socio(String dni, String nombre, String alta) {
        this.dni = dni;
        this.nombre = nombre;
        DateTimeFormatter f = DateTimeFormatter.ofPattern("dd/MM/yyyy");
        this.fechaAlta = LocalDate.parse(alta, f);
    }

    public void setfechaAlta(String alta){
        DateTimeFormatter f = DateTimeFormatter.ofPattern("dd/MM/yyyy");
        this.fechaAlta = LocalDate.parse(alta, f);
    }

    public Socio(String dni) {
        this.dni = dni;
    }

    int antiguedad() {
        return (int) fechaAlta.until(LocalDate.now(), ChronoUnit.YEARS);
    }

    @Override
    public int compareTo(Socio o) {
        return dni.compareTo(o.dni);
    }


    @Override
    public boolean equals(Object o) {
        return dni.equals(((Socio) o).dni);
    }

    @Override
    public String toString() {
        return "Socio(" + "dni=" + dni + ", nombre=" + nombre
                + ", antiguedad=" + antiguedad() + ")\n";
    }
}
```

```
--- exec:3.1.0:exec (default-cli) @ Aal2_16 ---
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
1
Introduzca dni (no se podrá modificar):
111
Nombre:
Lucas Pato
Fecha alta(dd/MM/yyyy):
05/03/2009
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
1
Introduzca dni (no se podrá modificar):
222
Nombre:
Abraham Q
Fecha alta(dd/MM/yyyy):
08/07/1987
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
1
Introduzca dni (no se podrá modificar):
333
Nombre:
Jose G
Fecha alta(dd/MM/yyyy):
```

```
Fecha alta(dd/MM/yyyy):
09/05/2000
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
4
--Socios ordenados por nombre:[Socio{dni=222, nombre=Abraham Q, antiguedad=36}
, Socio{dni=333, nombre=Jose G, antiguedad=23}
, Socio{dni=111, nombre=Lucas Pato, antiguedad=15}
]
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
5
--Socios ordenados por antiguedad:[Socio{dni=222, nombre=Abraham Q, antiguedad=36}
, Socio{dni=333, nombre=Jose G, antiguedad=23}
, Socio{dni=111, nombre=Lucas Pato, antiguedad=15}
]
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
2
Introduzca dni socio a modificar;
111
Socio encontrado
Introduzca nuevo nombre:
Antonio Santana
Fecha alta(dd/MM/yyyy):
02/02/2022
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
4
--Socios ordenados por nombre:[Socio{dni=222, nombre=Abraham Q, antiguedad=36}
, Socio{dni=111, nombre=Antonio Santana, antiguedad=2}
, Socio{dni=333, nombre=Jose G, antiguedad=23}
]
********Menú Socios********
1 Alta socio
```

```
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
5
--Socios ordenados por antiguedad:[Socio{dni=222, nombre=Abraham Q, antiguedad=36}
, Socio{dni=333, nombre=Jose G, antiguedad=23}
, Socio{dni=111, nombre=Antonio Santana, antiguedad=2}
]
********Menú Socios********
1 Alta socio
2 Modificar socio
3 Dar de baja socio
4 Listar socios por nombre
5 Listar socios por antigüedad
0 Salir:
0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  02:30 min
```

12.17. Implementa la clase `Cola` genérica utilizando un objeto `ArrayList` para guardar los elementos.



```java
package com.mycompany.aa12_17;

public class Aa12_17 {

    public static void main(String[] args) {
        Cola<Integer> c = new Cola<>();
        for (int i = 0; i < 10; i++) {
            c.encolar(cosa: i);
        }

        Integer num = c.desencolar();
        while (num != null) {
            System.out.println(x: num);
            num = c.desencolar();
        }
    }

}
```

```
Output - Run (Aa12_17)
Building Aa12_17 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ Aa12_17 ---
skip non existing resourceDirectory C:\Users\abrah\Documents\NetBeansProjects\Aa12_17\src\main\resour

--- compiler:3.11.0:compile (default-compile) @ Aa12_17 ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ Aa12_17 ---
0
1
2
3
4
5
6
7
8
9
------------------------------------------------------------------------
BUILD SUCCESS
```

```java
package com.mycompany.aa12_17;

import java.util.*;

public class Cola<T> {

    private List<T> cola = new ArrayList<T>();

    public void encolar(T cosa) {
        cola.add(e: cosa);
    }

    public T desencolar() {
        if (cola.size() > 0) {
            T elemento = cola.get(index: 0);
            cola.remove(index: 0);
            return elemento;
        }
        return null;
    }

}
```

12.18. Implementa la clase `Pila` genérica utilizando un objeto `ArrayList` para guardar los elementos.

```java
package com.mycompany.aa12_18;


public class Aa12_18 {
        public static void main(String[] args) {
        Pila<Integer> p=new Pila<>();
        for (int i = 0; i < 10; i++) {
            p.apilar(cosa: i);
        }

        Integer num=p.desapilar();
        while(num!=null){
            System.out.println(x: num);
            num=p.desapilar();
        }
    }
}
```

```java
package com.mycompany.aa12_18;

import java.util.ArrayList;
import java.util.List;

public class Pila<T> {
   private List<T> pila = new ArrayList<T>();

    public void apilar(T cosa) {
        pila.addFirst(e: cosa);
    }

    public T desapilar() {
        if (pila.size() > 0) {
            return pila.removeFirst();
        }
        return null;
    }

}
```

```
Output - Run (Aa12_18)
   from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ Aa12_18 ---
skip non existing resourceDirectory C:\Users\abrah\Documents\NetBeansProjects\Aa12_18\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ Aa12_18 ---
Changes detected - recompiling the module! :source
Compiling 2 source files with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ Aa12_18 ---
9
8
7
6
5
4
3
2
1
0
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.212 s
```

12.19. Escribe un programa donde se introduzca por consola una frase que conste exclusivamente de palabras separadas por espacios. Las palabras de la frase se almacenarán en una lista. Finalmente, se mostrarán por pantalla las palabras que estén repetidas y, a continuación, las que no lo estén.

```java
package com.mycompany.aa12_19;

import java.util.*;

public class Aa12_19 {

    public static void main(String[] args) {
        Scanner sc = new Scanner(source:System.in);
        List<String> texto = new ArrayList<>();
        Set<String> sinRepeticion = new LinkedHashSet<>();
        Set<String> repetidas = new LinkedHashSet<>();
        Map<String, Boolean> mapa = new LinkedHashMap<>();
        System.out.println(x: "Introduzca una frase:");
        String frase = sc.nextLine();
        texto.addAll(c: Arrays.asList(a: frase.split(regex: " ")));

        for (String palabra: texto) {
            mapa.put(key:palabra, value: mapa.containsKey(key:palabra));
        }
        for(Map.Entry<String, Boolean> elemento: mapa.entrySet()){
            if(elemento.getValue()){
                repetidas.add(e: elemento.getKey());
            }else{
                sinRepeticion.add(e: elemento.getKey());
            }
        }

        System.out.println("Palabras repetidas:" + repetidas);
        System.out.println("Palabras que no se repiten:" + sinRepeticion);
    }
}

```

```
Output - Run (Aa12_19)
--- resources:3.3.1:resources (default-resources) @ Aa12_19 ---
skip non existing resourceDirectory C:\Users\abrah\Documents\NetBeansProjects\Aa12_19\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ Aa12_19 ---
Changes detected - recompiling the module! :source
Compiling 1 source file with javac [debug target 21] to target\classes

--- exec:3.1.0:exec (default-cli) @ Aa12_19 ---
Introduzca una frase:
hola le dijo el perro a el gato y hola le contesto el gato a el perro
Palabras repetidas:[hola, le, el, perro, a, gato]
Palabras que no se repiten:[dijo, y, contesto]
------------------------------------------------------------------------
BUILD SUCCESS
```

12.24. Implementa una función a la que se le pasen dos listas de enteros ordenadas en sentido creciente y nos devuelva una única lista, también ordenada, fusión de las dos anteriores. Desarrolla el algoritmo de forma no destructiva, es decir, que las listas utilizadas como parámetros de entrada se mantengan intactas.

```java
package com.mycompany.aa12_24;

import java.util.*;

public class Aa12_24 {
    public static void main(String[] args) {
        List<Integer> numeros1 = new ArrayList(c: List.of(e1: 1,e2: 5,e3: 6,e4: 8,e5: 9,e6: 43));
        List<Integer> numeros2 = new ArrayList(c: List.of(e1: 2,e2: 3,e3: 10,e4: 18,e5: 25,e6: 50));
        Set<Integer> fusion =  new TreeSet();
        fusion.addAll(c: numeros1);
        fusion.addAll(c: numeros2);
        System.out.println("---Primera lista:" + numeros1);
        System.out.println("---Segunda lista:" + numeros2);
        System.out.println("--Lista fusionada:" + fusion);
    }
}
```

Output - Run (Aa12_24)

```
cd C:\Users\abrah\Documents\NetBeansProjects\Aa12_24; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd
Scanning for projects...

------------------------< com.mycompany:Aa12_24 >-------------------------
Building Aa12_24 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ Aa12_24 ---
skip non existing resourceDirectory C:\Users\abrah\Documents\NetBeansProjects\Aa12_24\src\main\resourc

--- compiler:3.11.0:compile (default-compile) @ Aa12_24 ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ Aa12_24 ---
---Primera lista:[1, 5, 6, 8, 9, 43]
---Segunda lista:[2, 3, 10, 18, 25, 50]
--Lista fusionada:[1, 2, 3, 5, 6, 8, 9, 10, 18, 25, 43, 50]
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  0.793 s
```