**Ejemplos de List y ArrayList**

**Ejemplo 1.**

Texto

Descripción generada automáticamente

**Texto

Descripción generada automáticamenteEjemplo 2**

**Texto

Descripción generada automáticamenteEjemplo 3.**

Texto

Descripción generada automáticamente

**Ejemplos de HashMap**

**Ejemplo 1**

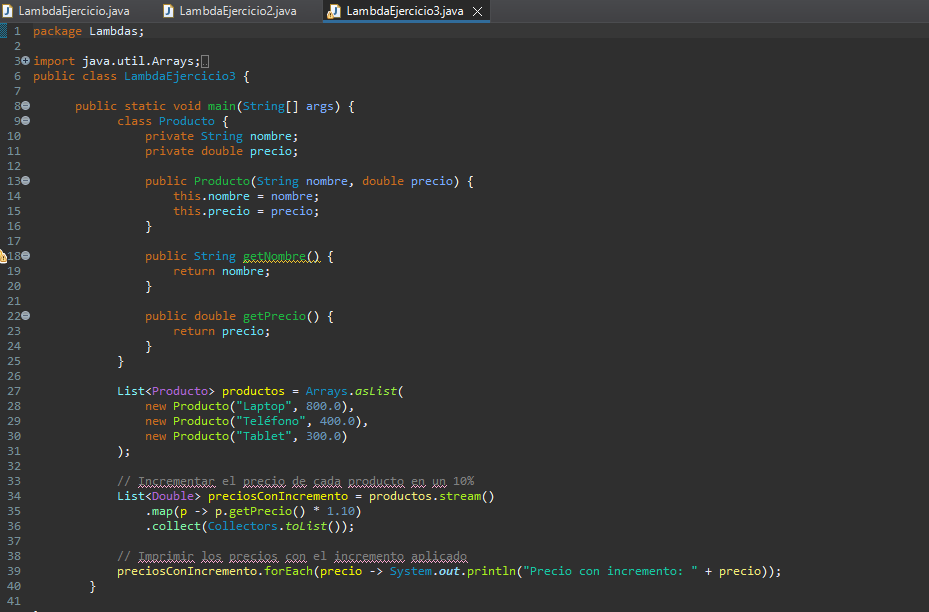
Texto

Descripción generada automáticamente

**Ejemplo 2**

Texto

Descripción generada automáticamente

**Ejemplo 3 **

**Ejemplos de LinkedList**

**Texto

Descripción generada automáticamenteEjemplo** 1

**Clase Persona**

Texto

Descripción generada automáticamente

**Ejemplo 2**

Texto

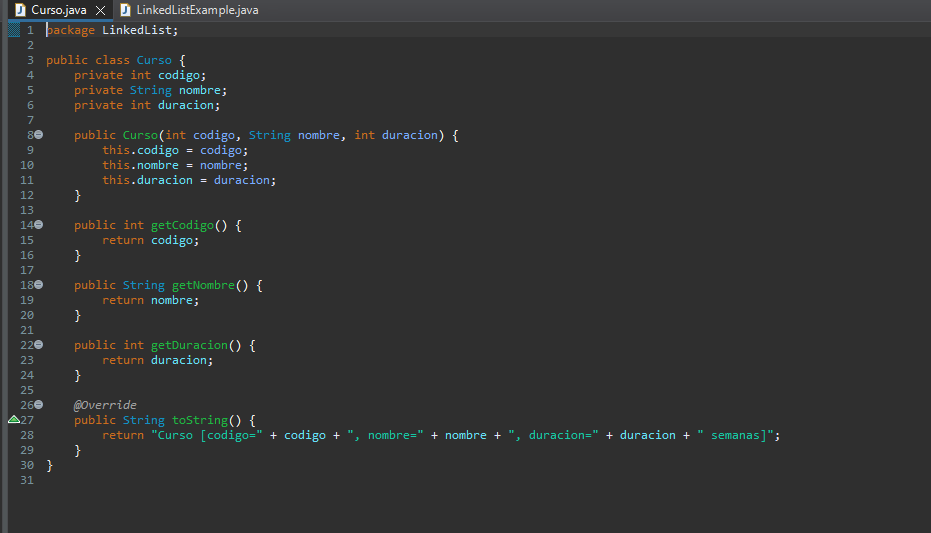
Descripción generada automáticamente

Texto

Descripción generada automáticamente

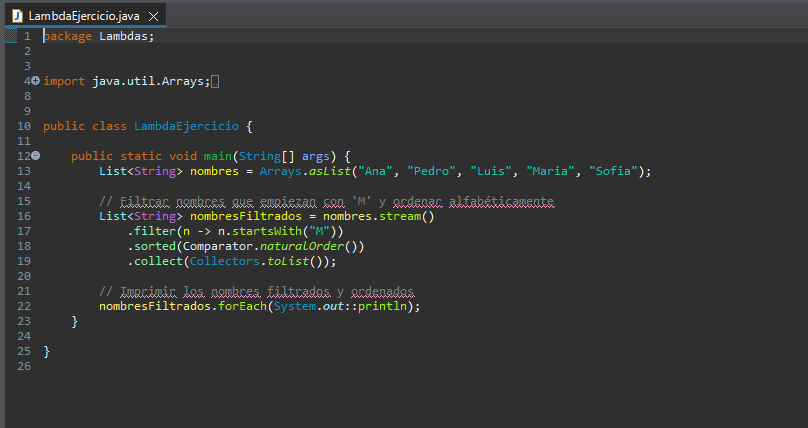
**Ejemplo 3**

**Clase Curso**

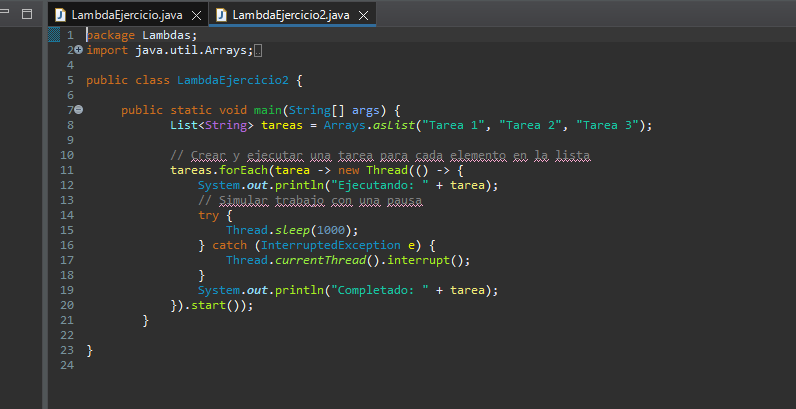


**Ejemplos de Lambdas**

**Ejemplo 1**



**Ejemplo 2**



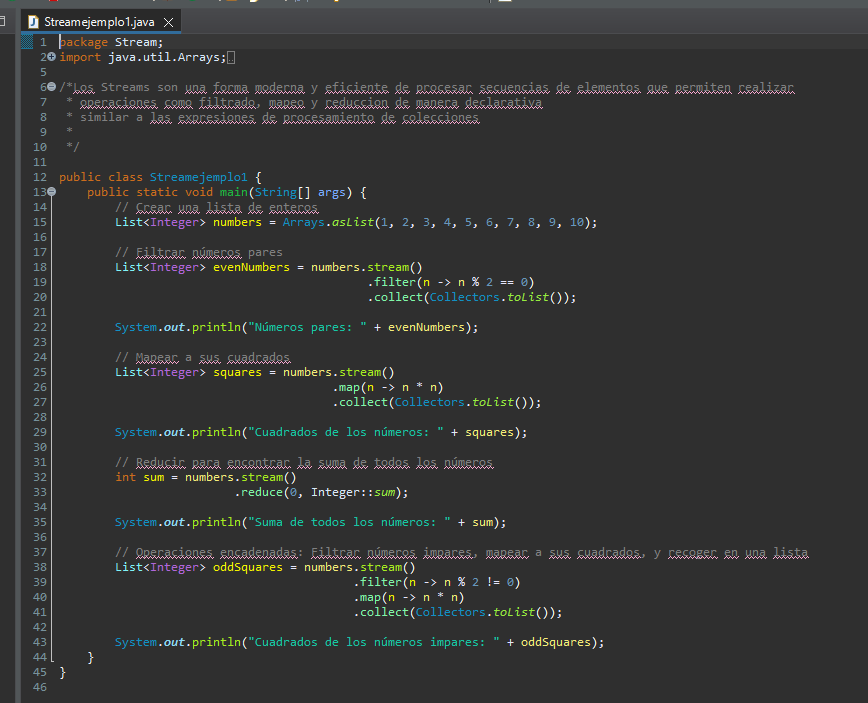
**Ejemplo 3**

Texto

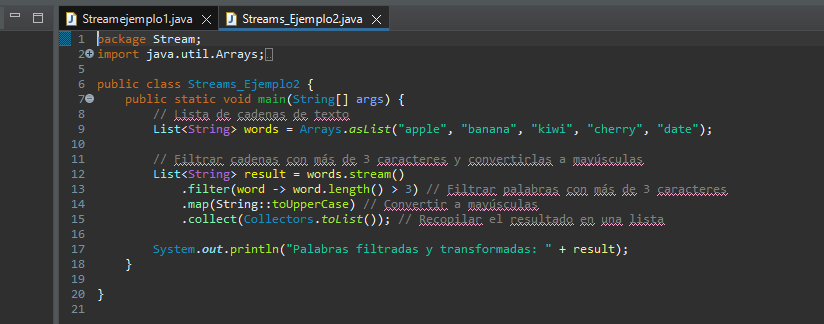
Descripción generada automáticamente

**Ejemplos de Stream**

**Ejemplo 1**



**Ejemplo 2**



Texto

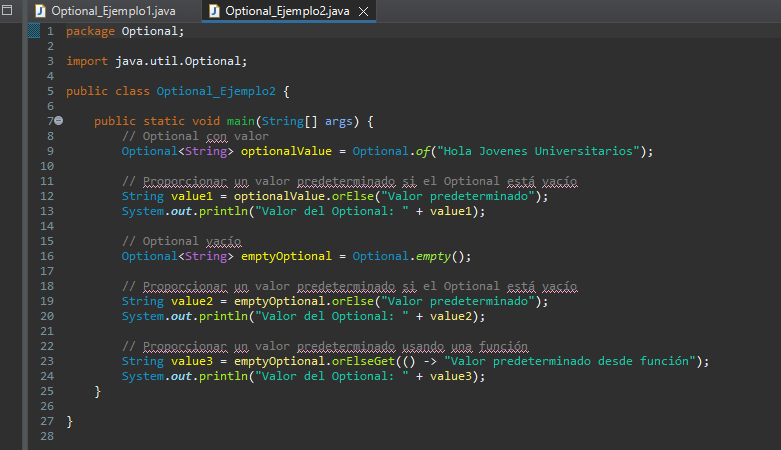
Descripción generada automáticamente**Ejemplo 3**

**Ejemplos de Optional**

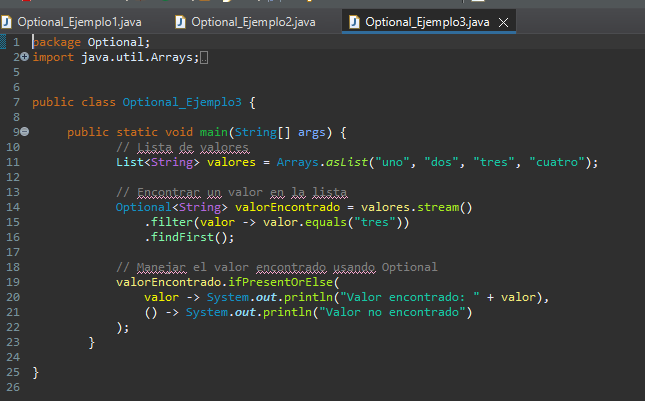
**Ejemplo 1**

Texto

Descripción generada automáticamente

**Ejemplo 2**

**Ejemplo 3**



**Texto

Descripción generada automáticamenteEjercicio de edad actual**

package Edad;

import java.time.LocalDate;

import java.time.Period;

import java.time.format.DateTimeFormatter;

import java.util.Scanner;

public class Edad {

public static void main(String[] args) {

System.***out***.print("Fecha de nacimiento (dd/mm/yyyy): ");

Scanner ingreso = new Scanner (System.***in***);

String fecha = ingreso.nextLine();

try {

DateTimeFormatter formatoFecha = DateTimeFormatter.*ofPattern*("dd/MM/yyyy");

LocalDate fechaNac = LocalDate.*parse*(fecha, formatoFecha);

LocalDate iechaAct = LocalDate.*now*();

Period periodo = Period.*between*(fechaNac, iechaAct);

System.***out***.printf("La eda es: %d años, %d meses y %d dias",

periodo.getYears(), periodo.getMonths(), periodo.getDays());

System.***out***.println();

} catch(Exception e) {

System.***out***.println("La fecha Ingresada no tiene el Formato ddd/mm/aaaa");

}

}

}

**Codigos.txt**

**Ejemplos de List y ArrayList**

**Ejemplo 1.**

package List\_ArrayList;

import java.util.ArrayList;

public class Ejemplo2\_ArrayList {

public static void main(String[] args) {

ArrayList<String> cadenas;

//crear instancia

cadenas = new ArrayList<String>();

//Agregar Elementos

cadenas.add("Hola");

cadenas.add("Buenos dias,");

cadenas.add("Un Placer Saludarte! ;)");

for (int i =0; i< cadenas.size(); i++) {

System.***out***.println(cadenas.get(i));

}

System.***out***.println("");

// bprrar elementos

cadenas.remove(1);

System.***out***.println("----Ejemplo Eliminando una Posicion-------");

for (int i =0; i< cadenas.size(); i++) {

System.***out***.println(cadenas.get(i));

}

}

}

**Ejemplo 2**

package List\_ArrayList;

/\*las listas son un conjunto de elementos relacionados entre nsi que tienen un determinado ordem

\* Orden FiFo Primero en entrar primero en salir

\* LifO el ultimo en entrar el primero en fsalir

\* \*/

public class Ejemplo1\_List {

private int num;

private String nombre;

private int edad;

public Ejemplo1\_List () {

}

public Ejemplo1\_List(int num, String nombre, int edad) {

super();

this.num = num;

this.nombre = nombre;

this.edad = edad;

}

public int getNum() {

return num;

}

public void setNum(int num) {

this.num = num;

}

public String getNombre() {

return nombre;

}

public void setNombre(String nombre) {

this.nombre = nombre;

}

public int getEdad() {

return edad;

}

public void setEdad(int edad) {

this.edad = edad;

}

}

**package List\_ArrayList;**

**import java.util.ArrayList;**

**import java.util.List;**

**public class Arraylist {**

**public static void main(String[] args) {**

**List<Ejemplo1\_List> lista= new ArrayList<Ejemplo1\_List>();**

**lista.add(new Ejemplo1\_List(1,"Anderson", 30));**

**lista.add(new Ejemplo1\_List(2,"Yobany", 20));**

**lista.add(new Ejemplo1\_List(3,"Mateo", 11));**

**System.out.println("----For---");**

**// recore por indice**

**for(int i=0; i<lista.size(); i++) {**

**System.out.println("Prueba" + lista.get(i).getNombre());**

**}**

**System.out.println("----Foreach-------");**

**//recorrido foreach recore uno por uno**

**for(Ejemplo1\_List perso:lista) {**

**System.out.println("prueba:" + perso.getNombre());**

**}**

**}**

**}Ejemplo 3**

**package List\_ArrayList;**

**import java.util.ArrayList;**

**import java.util.List;**

**public class Ejemplo3\_ArrayList {**

**public static void main(String[] args) {**

**List<String> cars = new ArrayList<>();**

**cars.add("Toyota");**

**cars.add("Honda");**

**cars.add("Ford");**

**cars.add("BMW");**

**// Acceso a los elementos de la lista**

**System.out.println("Primer carro: " + cars.get(0));**

**System.out.println("Segundo carro: " + cars.get(1));**

**// Recorrer la lista usando un bucle for-each**

**System.out.println("Lista de carros:");**

**for (String car : cars) {**

**System.out.println(car);**

**}**

**cars.set(2, "Chevrolet");**

**System.out.println("Carro modificado: " + cars.get(2));**

**cars.remove("BMW");**

**System.out.println("Lista de carros después de eliminar BMW:");**

**for (String car : cars) {**

**System.out.println(car);**

**}**

**}**

**}**

**Ejemplos de HashMap**

**Ejemplo 1**

package HashMap;

import java.util.Date;

public class Ejemplo2\_hashEstudiante {

private int id;

private String apellido;

private int edad;

private Date fechaIngreso;

public Ejemplo2\_hashEstudiante(int id, String apellido, int edad, Date fechaIngreso) {

super();

this.id = id;

this.apellido = apellido;

this.edad = edad;

this.fechaIngreso = fechaIngreso;

}

public Ejemplo2\_hashEstudiante () {

}

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getApellido() {

return apellido;

}

public void setApellido(String apellido) {

this.apellido = apellido;

}

public int getEdad() {

return edad;

}

public void setEdad(int edad) {

this.edad = edad;

}

public Date getFechaIngreso() {

return fechaIngreso;

}

public void setFechaIngreso(Date fechaIngreso) {

this.fechaIngreso = fechaIngreso;

}

*@Override*

public String toString() {

// **TODO** Auto-generated method stub

return "Ejemplo2\_hashEstudiante [id= "+ id + ", apellido="+ apellido + ", edad= "+ edad+ ", fechaIngreso= "+ fechaIngreso +"]";

}

}

**Ejemplo 2**

package HashMap;

import java.util.Date;

import java.util.HashMap;

public class HashMpa {

public static void main(String[] args) {

// Crear el HashMap para almacenar estudiantes

HashMap<Integer, Ejemplo2\_hashEstudiante> estudiantes = new HashMap<>();

algunos estudiantes

Ejemplo2\_hashEstudiante estudiante1 = new Ejemplo2\_hashEstudiante(1, "García", 20, new Date());

Ejemplo2\_hashEstudiante estudiante2 = new Ejemplo2\_hashEstudiante(2, "Martínez", 22, new Date());

Ejemplo2\_hashEstudiante estudiante3 = new Ejemplo2\_hashEstudiante(3, "López", 21, new Date());

// Agregar estudiantes al HashMap

estudiantes.put(estudiante1.getId(), estudiante1);

estudiantes.put(estudiante2.getId(), estudiante2);

estudiantes.put(estudiante3.getId(), estudiante3);

for (Ejemplo2\_hashEstudiante estudiante : estudiantes.values()) {

System.out.println(estudiante);

}

// Obtener un estudiante por su ID

Ejemplo2\_hashEstudiante estudianteObtenido = estudiantes.get(2);

if (estudianteObtenido != null) {

System.out.println("Estudiante con ID 2: " + estudianteObtenido);

} else {

System.out.println("No se encontró el estudiante con ID 2.");

}

// Eliminar un estudiante por su ID

estudiantes.remove(1);

System.out.println("Después de eliminar el estudiante con ID 1:");

for (Ejemplo2\_hashEstudiante estudiante : estudiantes.values()) {

System.out.println(estudiante);

}

}

}

**Ejemplo 3**

package HashMap;

import java.util.Date;

import java.util.HashMap;

public class Main\_producto {

public static void main(String[] args) {

// Crear el HashMap para almacenar productos

HashMap<Integer, Ejemplo3\_Producto> inventario = new HashMap<>();

// Crear algunos productos

Ejemplo3\_Producto producto1 = new Ejemplo3\_Producto(1, "Manzana", 0.50, new Date());

Ejemplo3\_Producto producto2 = new Ejemplo3\_Producto(2, "Naranja", 0.75, new Date());

Ejemplo3\_Producto producto3 = new Ejemplo3\_Producto(3, "Plátano", 0.30, new Date());

// Agregar productos al HashMap

inventario.put(producto1.getId(), producto1);

inventario.put(producto2.getId(), producto2);

inventario.put(producto3.getId(), producto3);

// Imprimir todos los productos

for (Ejemplo3\_Producto producto : inventario.values()) {

System.out.println(producto);

}

// Obtener un producto por su ID

Ejemplo3\_Producto productoObtenido = inventario.get(2);

if (productoObtenido != null) {

System.out.println("Producto con ID 2: " + productoObtenido);

} else {

System.out.println("No se encontró el producto con ID 2.");

}

// Eliminar un producto por su ID

inventario.remove(1);

System.out.println("Después de eliminar el producto con ID 1:");

for (Ejemplo3\_Producto producto : inventario.values()) {

System.out.println(producto);

}

}

}

**Ejemplos de LinkedList**

**Ejemplo 1**

package LinkedList;

import java.util.LinkedList;

import java.util.List;

public class LinkedListCursoExample {

public static void main(String[] args) {

List<Curso> listaCursos = new LinkedList<>()

// Agregar algunos objetos Curso a la lista

listaCursos.add(new Curso(101, "Introducción a la Programación", 8));

listaCursos.add(new Curso(102, "Estructuras de Datos", 12));

listaCursos.add(new Curso(103, "Bases de Datos", 10));

System.out.println("---- Foreach -------");

for (Curso curso : listaCursos) {

System.out.println("Curso: " + curso.getNombre() + ", Duración: " + curso.getDuracion() + " semanas");

}

System.out.println("");

Curso cursoObtenido = listaCursos.get(1);

System.out.println("Curso en el índice 1: " + cursoObtenido.getNombre());

System.out.println("");

listaCursos.remove(0);

System.out.println("Después de eliminar el primer curso:");

for (Curso curso : listaCursos) {

System.out.println("Curso: " + curso.getNombre() + ", Duración: " + curso.getDuracion() + " semanas");

}

System.out.println("");

listaCursos.add(0, new Curso(104, "Ingeniería de Software", 14));

System.out.println("Después de agregar un curso al inicio:");

for (Curso curso : listaCursos) {

System.out.println("Curso: " + curso.getNombre() + ", Duración: " + curso.getDuracion() + " semanas");

}

}

}

**Ejemplo 2**

package LinkedList;

import java.util.LinkedList;

import java.util.List;

public class LinkedListExample {

// las LinkedList solo se pueden recorer por foreach porque solo se pueden mostar uno por uno porque no tiene un id

public static void main(String[] args) {

List<Persona> lista = new LinkedList<>();

lista.add(new Persona(1, "Anderson", 30));

lista.add(new Persona(2, "Yobany", 20));

lista.add(new Persona(3, "Mateo", 11));

System.out.println("----Foreach-------");

// recorrido foreach recorre uno por uno

for (Persona perso : lista) {

System.out.println("prueba: " + perso.getNombre());

}

}

}

**Ejemplo 3**

package LinkedList;

import java.util.LinkedList;

import java.util.List;

/\*\*

\* Ejemplo de uso de LinkedList con objetos Persona.

\* Es una lista doblemente enlazada (ida y vuelta).

\* Permite duplicados y mantiene el orden de inserción.

\*/

public class LinkedListExample1 {

public static void main(String[] args) {

// Crear el LinkedList para almacenar objetos Persona

List<Persona> lista = new LinkedList<>();

// Agregar algunos objetos Persona a la lista

lista.add(new Persona(1, "Anderson", 30));

lista.add(new Persona(2, "Yobany", 20));

lista.add(new Persona(3, "Mateo", 11));

// Imprimir todos los elementos de la lista

System.out.println("---- Foreach -------");

for (Persona perso : lista) {

System.out.println("Nombre: " + perso.getNombre() + ", Edad: " + perso.getEdad());

}

// Obtener un elemento específico por su Indice

Persona personaObtenida = lista.get(1);

System.out.println("Persona en el índice 1: " + personaObtenida.getNombre());

// Eliminar un elemento por su índice

lista.remove(0);

System.out.println("Después de eliminar el primer elemento:");

for (Persona perso : lista) {

System.out.println("Nombre: " + perso.getNombre() + ", Edad: " + perso.getEdad());

}

// Agregar un elemento al inicio de la lista

lista.add(0, new Persona(4, "Carlos", 25));

System.out.println("Después de agregar un elemento al inicio:");

for (Persona perso : lista) {

System.out.println("Nombre: " + perso.getNombre() + ", Edad: " + perso.getEdad());

}

}

}

**Ejemplos de Lambdas**

**Ejemplo 1**

package Lambdas;

import java.util.Arrays;

import java.util.Comparator;

import java.util.List;

import java.util.stream.Collectors;

public class LambdaEjercicio {

public static void main(String[] args) {

List<String> nombres = Arrays.asList("Ana", "Pedro", "Luis", "Maria", "Sofia");

// Filtrar nombres que empiezan con 'M' y ordenar alfabéticamente

List<String> nombresFiltrados = nombres.stream()

.filter(n -> n.startsWith("M"))

.sorted(Comparator.naturalOrder())

.collect(Collectors.toList());

// Imprimir los nombres filtrados y ordenados

nombresFiltrados.forEach(System.out::println);

}

}

**Ejemplo 2**

package Lambdas;

import java.util.Arrays;

import java.util.List;

public class LambdaEjercicio2 {

public static void main(String[] args) {

List<String> tareas = Arrays.asList("Tarea 1", "Tarea 2", "Tarea 3");

// Crear y ejecutar una tarea para cada elemento en la lista

tareas.forEach(tarea -> new Thread(() -> {

System.out.println("Ejecutando: " + tarea);

// Simular trabajo con una pausa

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

Thread.currentThread().interrupt();

}

System.out.println("Completado: " + tarea);

}).start());

}

}

**Ejemplo 3**

package Lambdas;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

public class LambdaEjercicio3 {

public static void main(String[] args) {

class Producto {

private String nombre;

private double precio;

public Producto(String nombre, double precio) {

this.nombre = nombre;

this.precio = precio;

}

public String getNombre() {

return nombre;

}

public double getPrecio() {

return precio;

}

}

List<Producto> productos = Arrays.asList(

new Producto("Laptop", 800.0),

new Producto("Teléfono", 400.0),

new Producto("Tablet", 300.0)

);

// Incrementar el precio de cada producto en un 10%

List<Double> preciosConIncremento = productos.stream()

.map(p -> p.getPrecio() \* 1.10)

.collect(Collectors.toList());

// Imprimir los precios con el incremento aplicado

preciosConIncremento.forEach(precio -> System.out.println("Precio con incremento: " + precio));

}

}

**Ejemplos de Stream**

**Ejemplo 1**

package Stream;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

/\*Los Streams son una forma moderna y eficiente de procesar secuencias de elementos que permiten realizar

\* operaciones como filtrado, mapeo y reduccion de manera declarativa

\* similar a las expresiones de procesamiento de colecciones

\*

\*/

public class Streamejemplo1 {

public static void main(String[] args) {

// Crear una lista de enteros

List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

// Filtrar números pares

List<Integer> evenNumbers = numbers.stream()

.filter(n -> n % 2 == 0)

.collect(Collectors.toList());

System.out.println("Números pares: " + evenNumbers);

// Mapear a sus cuadrados

List<Integer> squares = numbers.stream()

.map(n -> n \* n)

.collect(Collectors.toList());

System.out.println("Cuadrados de los números: " + squares);

// Reducir para encontrar la suma de todos los números

int sum = numbers.stream()

.reduce(0, Integer::sum);

System.out.println("Suma de todos los números: " + sum);

// Operaciones encadenadas: Filtrar números impares, mapear a sus cuadrados, y recoger en una lista

List<Integer> oddSquares = numbers.stream()

.filter(n -> n % 2 != 0)

.map(n -> n \* n)

.collect(Collectors.toList());

System.out.println("Cuadrados de los números impares: " + oddSquares);

}

}

**Ejemplo 2**

package Stream;

import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

public class Streams\_Ejemplo2 {

public static void main(String[] args) {

// Lista de cadenas de texto

List<String> words = Arrays.asList("apple", "banana", "kiwi", "cherry", "date");

// Filtrar cadenas con más de 3 caracteres y convertirlas a mayúsculas

List<String> result = words.stream()

.filter(word -> word.length() > 3) // Filtrar palabras con más de 3 caracteres

.map(String::toUpperCase) // Convertir a mayúsculas

.collect(Collectors.toList()); // Recopilar el resultado en una lista

System.out.println("Palabras filtradas y transformadas: " + result);

}

}

**Ejemplo 3**

package Stream;

import java.util.Arrays;

import java.util.Comparator;

import java.util.List;

import java.util.OptionalDouble;

import java.util.Optional;

class Persona {

String nombre;

int edad;

Persona(String nombre, int edad) {

this.nombre = nombre;

this.edad = edad;

}

public String getNombre() {

return nombre;

}

public int getEdad() {

return edad;

}

@Override

public String toString() {

return nombre + " (" + edad + " años)";

}

}

public class Streams\_Ejemplo3 {

public static void main(String[] args) {

// Crear una lista de personas

List<Persona> personas = Arrays.asList(

new Persona("Ana", 23),

new Persona("Juan", 45),

new Persona("Luis", 30),

new Persona("María", 20),

new Persona("Carlos", 40)

);

// Calcular el promedio de las edades

OptionalDouble averageAge = personas.stream()

.mapToInt(Persona::getEdad) // Convertir el Stream a IntStream con las edades

.average(); // Calcular el promedio

averageAge.ifPresent(avg -> System.out.println("Promedio de edades: " + avg));

// Encontrar la persona más joven

Optional<Persona> youngestPerson = personas.stream()

.min(Comparator.comparingInt(Persona::getEdad)); // Encontrar la persona con la edad mínima

youngestPerson.ifPresent(youngest -> System.out.println("Persona más joven: " + youngest));

}

}

**Ejemplos de Optional**

**Ejemplo 1**

package Optional;

import java.util.Optional;

public class Optional\_Ejemplo1 {

public static void main(String[] args) {

// Crear un Optional que contiene un valor

Optional<String> optionalValue = Optional.*of*("Hello, World!");

// Verificar si el Optional contiene un valor

if (optionalValue.isPresent()) {

System.***out***.println("El Optional contiene: " + optionalValue.get());

} else {

System.***out***.println("El Optional está vacío");

}

// Crear un Optional vacío

Optional<String> emptyOptional = Optional.*empty*();

// Verificar si el Optional está vacío

if (emptyOptional.isPresent()) {

System.***out***.println("El Optional contiene: " + emptyOptional.get());

} else {

System.***out***.println("El Optional está vacío");

}

}

}

**Ejemplo 2**

package Optional;

import java.util.Optional;

public class Optional\_Ejemplo2 {

public static void main(String[] args) {

// Optional con valor

Optional<String> optionalValue = Optional.*of*("Hola Jovenes Universitarios");

// Proporcionar un valor predeterminado si el Optional está vacío

String value1 = optionalValue.orElse("Valor predeterminado");

System.***out***.println("Valor del Optional: " + value1);

// Optional vacío

Optional<String> emptyOptional = Optional.*empty*();

// Proporcionar un valor predeterminado si el Optional está vacío

String value2 = emptyOptional.orElse("Valor predeterminado");

System.***out***.println("Valor del Optional: " + value2);

// Proporcionar un valor predeterminado usando una función

String value3 = emptyOptional.orElseGet(() -> "Valor predeterminado desde función");

System.***out***.println("Valor del Optional: " + value3);

}

}

**Ejemplo 3**

package Optional;

import java.util.Arrays;

import java.util.List;

import java.util.Optional;

public class Optional\_Ejemplo3 {

public static void main(String[] args) {

// Lista de valores

List<String> valores = Arrays.asList("uno", "dos", "tres", "cuatro");

// Encontrar un valor en la lista

Optional<String> valorEncontrado = valores.stream()

.filter(valor -> valor.equals("tres"))

.findFirst();

// Manejar el valor encontrado usando Optional

valorEncontrado.ifPresentOrElse(

valor -> System.out.println("Valor encontrado: " + valor),

() -> System.out.println("Valor no encontrado")

);

}

}