

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLD-001	Página: 1



GUÍA DE LABORATORIO

(formato docente)

INFORMACIÓN BÁSICA					
ASIGNATURA:	ESTRUCTURA DE DATOS Y ALGORITMOS				
TÍTULO DE LA PRÁCTICA:	POO, HERENCIA, INTERFACES Y GENERICIDAD				
NÚMERO DE PRÁCTICA:	03	AÑO LECTIVO:	2025 – A	NRO. SEMESTRE:	TERCERO III
TIPO DE PRÁCTICA:	INDIVIDUAL	X			
	GRUPAL	—	MÁXIMO DE ESTUDIANTES	00	
FECHA INICIO:	19/05/2025	FECHA FIN:	23/05/2025	DURACIÓN:	90 minutos.
RECURSOS A UTILIZAR: <ul style="list-style-type: none"> • Github. • Lenguaje de Programación Java. • Ide Java Eclipse/Visual Studio Code. 					
DOCENTE(s): <ul style="list-style-type: none"> • Mg. Ing. Rene Alonso Nieto Valencia. 					

OBJETIVOS/TEMAS Y COMPETENCIAS	
OBJETIVOS: <ul style="list-style-type: none"> • Aprenda Herencia, Interfaces y Genericidad. • Aplicar conceptos elementales de programación a resolver utilizando POO en problemas de algoritmos. • Desarrollar pruebas. 	
TEMAS: <ul style="list-style-type: none"> • Introducción. • TAD. • POO, Herencia, Interfaces y Genericidad. 	
COMPETENCIAS	C.a
	C.b
	C.c
	C.d

CONTENIDO DE LA GUÍA

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 2</p>

I. MARCO CONCEPTUAL

- <https://www.w3schools.com/java/>
- <https://docs.oracle.com/javase/7/docs/api/java/util/List.html>
- <https://docs.oracle.com/javase/tutorial/java/generics/types.html>

II. EJERCICIO/PROBLEMA RESUELTO POR EL DOCENTE

En un editor Java, realizar la integración de los siguientes ejercicios, revisar los resultados obtenidos y realizar una explicación del funcionamiento de forma concreta y clara.

1. ArrayList:

```
ArrayList<String> alumnos = new ArrayList<String>();
ArrayList<Integer> notas = new ArrayList<Integer>();
alumnos.add("MARIA");
alumnos.add("DIEGO");
alumnos.add("RENE");
alumnos.add("ALONSO");
System.out.println(alumnos.hashCode());
System.out.println(alumnos.isEmpty());
System.out.println(alumnos.size());
```

2. Iterador:

```
Iterator<String> itA = alumnos.iterator();
while (itA.hasNext()) {
    System.out.println(itA.next());
}
```

3. Clase Animal en Java:

```
public class Animal {
    String nombre;
    boolean genero;

    public Animal(String nombre, boolean genero) {
        super();
        this.nombre = nombre;
        this.genero = genero;
    }

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
}
```

```
public boolean isGenero() {  
    return genero;  
}  
  
public void setGenero(boolean genero) {  
    this.genero = genero;  
}  
}
```

```
ArrayList<Animal> mascotas = new ArrayList<Animal>();  
List<Animal> mascotas2 = new List<Animal>(); /** <- Ver error */  
List<Animal> mascotas2 = new ArrayList<Animal>();
```

III. EJERCICIOS/PROBLEMAS PROPUESTOS

De acuerdo a los ejercicios propuestos desarrollar los algoritmos y mostrar las siguientes indicaciones:

- Enunciado del ejercicio.
- Código en java desarrollado.
- Resultados obtenidos.
- Explicación breve y concreta del código implementado.

1. Listas, Implementar una Lista usando **POO** con **clases** y **métodos genéricos** siguiendo los estándares de Java. (Los métodos para una lista)

- Referencia: <https://docs.oracle.com/javase/7/docs/api/java/util/List.html>
- Puede **ignorar** los siguientes métodos:
 - ✓ **hashCode()**
 - ✓ **iterator()**
 - ✓ **listIterator()**
 - ✓ **listIterator(int index)**
 - ✓ **retainAll(Collection<?> c)**
 - ✓ **toArray()**
 - ✓ **toArray(T[] a)**
- Implemente una clase **Node<T>** donde **T** es un **tipo genérico**, esta clase debe contener al menos dos propiedades.
 - ✓ <https://docs.oracle.com/javase/tutorial/java/generics/types.html>
 - ✓ T data: la información almacenada en el nodo Node<T> nextNode: una referencia al siguiente nodo
 - ✓
- Implementar una clase List<T> esta clase debe contener al menos esta propiedad
 - ✓ Node<T> root: la referencia sobre el nodo inicial

2. Calculadora Genérica, Cree un nuevo proyecto en Java: **CalculadoraGenerica**.

- Implementar las clases Genérica **Operador<T>**, para declarar sus **atributos** (valor1 y valor2), **constructor** (Operador).

```
public class Operador<T extends Number> {  
  
    private T valor1;  
    private T valor2;  
  
    public Operador(T valor1, T valor2) {  
        this.valor1 = valor1;  
        this.valor2 = valor2;  
    }  
}
```

- Escribir la clase **Main**, e implementar el método genérico suma con la siguiente estructura.

```
public class Main {  
  
    static <T extends Number> double suma(T valor1, T valor2) {  
        return (valor1.doubleValue() + valor2.doubleValue());  
    }  
  
    public static void main(String[] args) {  
        Scanner scanner = new Scanner(System.in);  
        int opcion;  
    }  
}
```

- Escribir los métodos genéricos: **resta, producto, división, potencia, raíz cuadrada y raíz cubica.**
- Para poder probar las clases y métodos genéricos implementar mediante un menú de opciones las operaciones, mostrando los resultados:



Menú de Operaciones Clases Genéricas:

1. Suma.
2. Resta.
3. Producto.
4. División.
5. Potencia.
6. Raíz Cuadrada.
7. Raíz Cubica.
8. Salir del Programa.

- **Nota:** El programa debe permitir validar entre valores o tipo de dato (**integer o double**) para poder utilizar los métodos genéricos, el programa no termina hasta escoger la opción **SALIR**.

IV. CUESTIONARIO

1. ¿Cuáles fueron las dificultades que encontraste al desarrollar los ejercicios propuestos? por ejemplo, poca documentación, complejidad del lenguaje, etc.
2. ¿Qué diferencia hay entre un List y un ArrayList en Java?
3. ¿Qué beneficios y oportunidades ofrecen las clases genéricas en Java?

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLD-001</p>	<p>Página: 5</p>

V. REFERENCIAS Y BIBLIOGRAFÍA RECOMENDADAS:

- Weiss M., Data Structures & Problem Solving Using Java, 2010, Addison-Wesley.
- Weiss M., Data Structures and Algorithms Analysis in Java, 2012, Addison-Wesley.
- Cormen T., Leiserson C., Rivest R., Stein C., Introduction to Algorithms, 2022, The MIT Press
- The Java™ Tutorials - <https://docs.oracle.com/javase/tutorial/>
- Sedgewick, R., Algorithms in Java, Parts 1-4: Fundamentals, Data Structures, Sorting, Searching, Part 5:
- Graph Algorithms, Addison-Wesley.
- Malik D., Data Structures Using C++, 2003, Thomson Learning.
- Knuth D., The Art of Computer Programming, Vol. 1 y 3, Addison - Wesley.

TÉCNICAS E INSTRUMENTOS DE EVALUACIÓN	
<p>TÉCNICAS: <i>Actividades Resueltas</i> <i>Ejercicios Propuestos</i></p>	<p>INSTRUMENTOS: <i>Rubricas</i></p>
<p>CRITERIOS DE EVALUACIÓN Los criterios de evaluación se encuentran en el silabo DUFA ANEXO en la sección EVOLUCIÓN CONTINUA</p>	