



Contents

| | |
|---|---|
| Raíces del Código | 3 |
| Fundamentos de Java a través de la gestión de viveros forestales..... | 3 |
| I. Prólogo | 4 |
| II. Introducción..... | 5 |
| V. Ejercicio 0: Hola Bosque..... | 6 |
| VI. Ejercicio 1: Área de Plantación..... | 6 |
| VII. Ejercicio 2: Conteo de Brotes..... | 7 |
| VIII. Ejercicio 3: Control de Altura..... | 7 |
| IX. Ejercicio 4: Alerta de Sequía | 7 |
| X. Ejercicio 5: Registro de Crecimiento..... | 8 |

Raíces del Código

Java es un lenguaje de programación **orientado a objetos (POO)** que se usa para crear desde aplicaciones de escritorio, móviles (como aplicaciones Android) hasta servidores y sistemas empresariales. Su filosofía principal es “**escribe una vez, corre en cualquier lugar**” porque funciona sobre la **Java Virtual Machine (JVM)**, que traduce el código Java a un lenguaje que cualquier sistema operativo puede ejecutar.

Fundamentos de Java a través de la gestión de viveros forestales

Descubre la sintaxis y la semántica fundamentales de Java analizando datos de un vivero forestal. Aprende métodos, variables y flujo de control mientras contribuyes a la reforestación tecnológica.

Instrucciones Generales

- ✓ **Versión:** Java 17+ (OpenJDK).
- ✓ **Estructura:** Una clase por archivo (sin package). **Método *public static void*.**
- ✓ **Linter Obligatorio:** Checkstyle (Google Java Style). Errores de estilo = KO.

- ✓ **Comentarios:** Obligatorios, solo en inglés y siguiendo formato Javadoc.
- ✓ **IDE:** IntelliJ IDEA (recomendado), VS Code, Eclipse, etc.
- ✓ **Compilación:** Vía terminal: `javac Clase.java` y `java Clase`.

I. Prólogo

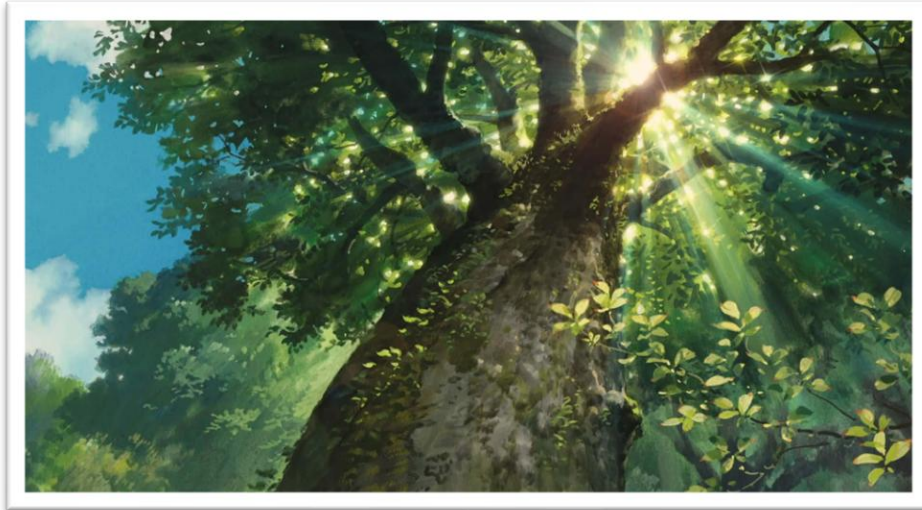
En los viveros forestales, cada semilla representa el futuro de un ecosistema. El seguimiento preciso de la germinación y el crecimiento es vital para asegurar que los bosques del mañana sean fuertes.

Java, al igual que un bosque de secuoyas, es conocido por su estructura, su altura y su capacidad para perdurar en el tiempo. A través de este proyecto, aprenderás que programar, al igual que la silvicultura, requiere paciencia, estructura y una atención meticulosa a los detalles.



II. Introducción

*¡Te damos la bienvenida a **Raíces del Código!***



Trabajarás con ejercicios prácticos que introducen conceptos de Java en un contexto de gestión forestal.

IMPORTANTE: En cada ejercicio, debes entregar una **Clase** que contenga el método estático solicitado.

- ✓ No incluyas el método **main** en tus archivos de entrega.
- ✓ La gestión de entrada y salida (**System.out** y **Scanner**) debe ocurrir dentro del método.

IV. Instrucciones Generales

- ✓ Usa Java 17+.
- ✓ No utilices packages.
- ✓ Cada archivo debe llamarse exactamente como la clase (ej: FtHelloForest.java).
- ✓ El método debe ser **public static void**.
- ✓ Para leer datos, usa **import java.util.Scanner;**

V. Ejercicio 0: Hola Bosque

- **Directorio:** ex00/
- **Archivo:** FtHelloForest.java
- **Método:** ft_hello_forest()

Escribe un método que muestre un mensaje de bienvenida al sistema de gestión forestal.

```
>>> FtHelloForest.ft_hello_forest();  
Welcome to the Forest Management System!
```

VI. Ejercicio 1: Área de Plantación

- **Directorio:** ex01/
- **Archivo:** FtPlantingArea.java
- **Método:** ft_planting_area()

Solicita la longitud y el ancho (en metros) de una sección de reforestación y muestra el área total.

```
Enter length: 10  
Enter width: 20  
Total planting area: 200 m2
```

VII. Ejercicio 2: Conteo de Brotes

- **Directorio:** ex02/
- **Archivo:** FtSproutCount.java
- **Método:** ft_sprout_count()

Un guardabosques cuenta los brotes nuevos en 3 parcelas distintas. Solicita los 3 valores y muestra la suma total.

```
Plot 1 sprouts: 50
Plot 2 sprouts: 30
Plot 3 sprouts: 20
Total sprouts counted: 100
```

VIII. Ejercicio 3: Control de Altura

- **Directorio:** ex03/
- **Archivo:** FtTreeHeight.java
- **Método:** ft_tree_height()

Solicita la altura de un ejemplar en centímetros. Si mide más de 100 cm, indica que está listo para ser trasplantado al bosque.

```
Enter tree height (cm): 120
Ready for transplanting!
```

IX. Ejercicio 4: Alerta de Sequía

- **Directorio:** ex04/
- **Archivo:** FtDroughtAlert.java
- **Método:** ft_drought_alert()

Solicita los litros de lluvia caídos esta semana. Si es menor a 15 litros, ¡muestra "Activate irrigation system!", de lo contrario "Moisture levels optimal".

```
Rainfall (liters): 10
Activate irrigation system!
```

X. Ejercicio 5: Registro de Crecimiento

- **Directorio:** ex05/
- **Archivo:** FtGrowthLog.java
- **Métodos:** ft_growth_log_iterative() y ft_growth_log_recursive(int n)

Ambos métodos deben imprimir el progreso día a día hasta llegar al objetivo dado por el usuario.

```
Target day: 3
Day 1: Growing...
Day 2: Growing...
Day 3: Growing...
Target reached!
```

XI. Ejercicio 6: Ficha del Vivero

- **Directorio:** ex06/
- **Archivo:** FtNurserySummary.java
- **Método:** ft_nursery_summary()

Solicita el nombre del vivero y la especie principal de árbol, mostrando un resumen formateado.

```
>>> FtNurserySummary.ft_nursery_summary()
Enter nursery name: Valle Verde
Enter main species: Roble
Nursery: Valle Verde | Species: Roble | Status: Active
```


XII. Ejercicio 7: Inventario de Semillas

- **Directorio:** ex07/
- **Archivo:** FtForestSeeds.java
- **Método:** ft_forest_seeds(String type, int qty, String unit)

El método recibe parámetros y debe validar la unidad ("kilograms", "seeds", "bags").

- Si la unidad no es válida, mostrar "Unknown unit".
- Formatear el tipo de semilla para que empiece en mayúscula.

```
>>> FtForestSeeds.ft_forest_seeds("pine", 50, "kilograms")
Pine: 50 kilograms in stock.

>>> FtForestSeeds.ft_forest_seeds("OAK", 200, "seeds")
Oak: 200 seeds ready for planting.

>>> FtForestSeeds.ft_forest_seeds("fir", 10, "bags")
Fir: 10 bags of seeds available.

>>> FtForestSeeds.ft_forest_seeds("cedar", 5, "boxes")
Unknown unit.
```