

PLANTILLA DE ENTREGA DE INFORME

**MÓDULO: PRO401-9524-225081-ONL-TALLER DE APLICACIONES MÓVILES**

**SEMANA: 3**

Docente: Iván Ayala

Estudiante: Karla Pesce, Jaime Codoceo y Sergio Molina

Índice

[Introducción 3](#_Toc207623836)

[Desarrollo 4](#_Toc207623837)

[Proceso de compilación 5](#_Toc207623838)

[Proceso de ejecución 5](#_Toc207623839)

[Ejemplo de la ejecución 5](#_Toc207623840)

[Documentación en GitHub 6](#_Toc207623841)

# Introducción

El desarrollo de aplicaciones en Java constituye una de las primeras aproximaciones al mundo de la programación estructurada y orientada a objetos. En esta actividad se busca comprender el proceso completo de creación, compilación y ejecución de un programa en Java sin utilizar un IDE, con el fin de fortalecer el dominio de la línea de comandos y afianzar la lógica básica.

El ejercicio práctico consiste en elaborar un programa sencillo que solicite al usuario datos específicos de un vehículo —marca, modelo, cilindrada, tipo de combustible y capacidad en pasajeros— utilizando la clase Scanner para la entrada de datos y variables de tipo String e int para su almacenamiento. Posteriormente, el programa debe mostrar en consola los valores ingresados, reforzando el ciclo entrada–proceso–salida fundamental en cualquier aplicación.

Más allá del código, la actividad promueve la documentación del proceso completo en un repositorio de GitHub, incluyendo requerimientos funcionales y no funcionales, historias de usuario y un cronograma de trabajo mediante GitHub Projects. Finalmente, se solicita la entrega de un informe que evidencie la práctica realizada, explique el proceso de compilación por consola y presente reflexiones sobre el aprendizaje adquirido.

# Desarrollo

Para esta actividad se creo un block de notas, guardado con el nombre VehiculoApp.java, ya que en java en nombre del archivo debe estar relacionado con la clase creada.

Este código utiliza la clase Scanner para poder leer los datos desde el teclado y variables de tipo String e int para guardar la información.

Código Utilizado.

|  |
| --- |
| import java.util.Scanner; // Para leer desde teclado  class VehiculoApp {  public static void main(String[] args) {  Scanner sc = new Scanner(System.in);  // Entradas (String e int según corresponde)  System.out.print("Marca: ");  String marca = sc.nextLine();  System.out.print("Modelo: ");  String modelo = sc.nextLine();  System.out.print("Cilindrada (cc): ");  int cilindrada = sc.nextInt();  sc.nextLine(); // limpiar salto de línea pendiente  System.out.print("Tipo de Combustible (gasolina/diesel/híbrido/eléctrico): ");  String combustible = sc.nextLine();  System.out.print("Capacidad en pasajeros: ");  int capacidad = sc.nextInt();  // Salidas  System.out.println("\nLa marca que ha ingresado es: " + marca);  System.out.println("El modelo que ha ingresado es: " + modelo);  System.out.println("La cilindrada que ha ingresado es: " + cilindrada + " cc");  System.out.println("El tipo de combustible es: " + combustible);  System.out.println("Tiene una capacidad de " + capacidad + " pasajeros.");  sc.close();  }  } |

## Proceso de compilación

El archivo se compilo desde la línea de comandos (CMD) utilizando el compilador de java (javac), dentro de la carpeta donde esta el archivo se ejecutó el siguiente comando:

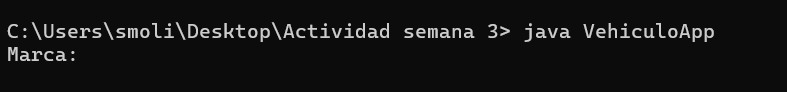
Texto

El contenido generado por IA puede ser incorrecto.

Este comando generó un archivo VehiculoApp.class, que contiene el bytecode para ser ejecutado en la JVM (Java Virtual Machine).

## Proceso de ejecución

Una vez compilado se ejecutó el siguiente comando:



El programa pidió al usuario los siguientes datos: Marca, Modelo, Cilindrada, Tipo de Combustible y Capacidad en pasajeros, y luego los mostró en consola.

## Ejemplo de la ejecución

Texto

El contenido generado por IA puede ser incorrecto.

## Documentación en GitHub

Para cumplir con la consigna, se creó un repositorio en GitHub que contiene:

* El código fuente VehiculoApp.java.
* Un archivo README.md con:
  + Instrucciones de compilación y ejecución.
  + Requerimientos funcionales y no funcionales.
  + Evidencias (capturas de pantalla del proceso).
* Historias de usuario documentadas.
* Un cronograma inicial de trabajo creado en GitHub Projects, con las tareas planificadas.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

# Conclusión

El desarrollo de la actividad permitió aplicar los conceptos básicos de Java sin IDE, reforzando el uso de la consola para compilar y ejecutar programas. Se implementó un sistema sencillo que solicita y muestra datos de un vehículo utilizando la clase Scanner, cumpliendo con los requerimientos funcionales y no funcionales definidos.

Además, se fortalecieron buenas prácticas de documentación al crear un README.md estructurado, con instrucciones claras de compilación y ejecución, evidencias gráficas del proceso, así como historias de usuario y un cronograma de trabajo.

Con esta experiencia, se afianzaron competencias clave en programación estructurada, control de versiones con GitHub y gestión de proyectos mediante la organización de entregables.

Bibliografía

En esta sección, declara las fuentes y autores sobre los que has basado tu informe, siguiendo el siguiente modelo de referencia bibliográfica acorde a la norma APA:

**Modelo:**

Apellido, Nombre del autor (año de publicación). Título. Ciudad, País de publicación: Editorial, capítulo y páginas.

**Ejemplo:**

Gutierrez, R. M, (2013). El impacto de la sobrepoblación de invertebrados en un ecosistema selvático . Revista Mundo Natural, capítulo 8, páginas 73-82.