



Actividad Sumativa : “Java APP”



MÓDULO: Taller de aplicaciones móviles

SEMANA: 3

Docente: Iván Ayala Ayala

Estudiante: Oliver Rubio Rauld

Índice

<i>Introducción</i>	3
<i>Desarrollo</i>	4
<i>Conclusión</i>	9
<i>Bibliografía</i>	10

Introducción

Este documento presenta el desarrollo de una aplicación en Java diseñada para gestionar el cálculo de costos de despacho en una distribuidora de alimentos. Incluye el registro de vehículos de reparto y el cálculo automatizado del costo de envío según el monto de compra y la distancia a recorrer. El código implementa estructuras de control condicional, manejo de entrada de datos mediante la clase Scanner, y muestra un ejemplo práctico de programación orientada a objetos en un contexto de logística y distribución. El objetivo es demostrar la aplicación de conceptos básicos de programación en la resolución de problemas del mundo real.

Desarrollo

Código

// Importar la clase Scanner para leer entrada del usuario

```
import java.util.Scanner;
```

```
/**
```

```
 * Sistema de cálculo de despacho para distribuidora de alimentos
```

```
 */
```

```
public class DistribuidoraAlimentos {
```

```
    /**
```

```
     * Método principal que ejecuta la aplicación
```

```
    */
```

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

```
        // Crear objeto Scanner para leer entrada del usuario
```

```
        Scanner scanner = new Scanner(System.in);
```

```
        // Variables para almacenar datos del vehículo de despacho
```

```
        String marca;           // Almacena la marca del vehículo
```

```
        String modelo;         // Almacena el modelo del vehículo
```

```
        int cilindrada;        // Almacena la cilindrada del motor
```

```
        String combustible;    // Almacena el tipo de combustible
```

```
        int pasajeros;         // Almacena la capacidad de pasajeros
```

```
        // Variables para el cálculo del despacho
```

```
        double totalCompra;    // Almacena el monto total de la compra
```

```
        double distancia;      // Almacena la distancia en kilómetros
```

```
        double costoDespacho;  // Almacena el costo total del despacho
```

```
        // Mostrar mensaje de bienvenida
```

```
        System.out.println("=== SISTEMA DE DISTRIBUCIÓN DE ALIMENTOS ===");
```

```
        System.out.println("=== REGISTRO DE VEHÍCULO DE DESPACHO ===\n");
```

// Solicitar y leer datos del vehículo

System.out.print("Ingrese la marca del vehículo: ");

marca = scanner.nextLine(); *// Leer línea completa para marca*

System.out.print("Ingrese el modelo del vehículo: ");

modelo = scanner.nextLine(); *// Leer línea completa para modelo*

System.out.print("Ingrese la cilindrada del motor (cc): ");

cilindrada = scanner.nextInt(); *// Leer número entero para cilindrada*

scanner.nextLine(); *// Limpiar buffer del scanner*

System.out.print("Ingrese el tipo de combustible: ");

combustible = scanner.nextLine(); *// Leer línea completa para combustible*

System.out.print("Ingrese la capacidad de pasajeros: ");

pasajeros = scanner.nextInt(); *// Leer número entero para pasajeros*

// Mostrar datos ingresados (Salida requerida)

System.out.println("\n=== DATOS DEL VEHÍCULO REGISTRADO ===");

System.out.println("La 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 " + pasajeros + " pasajeros.");

// Sección de cálculo de despacho

System.out.println("\n=== CÁLCULO DE COSTO DE DESPACHO ===");

// Solicitar datos para el cálculo del despacho

System.out.print("Ingrese el total de la compra (\$): ");

totalCompra = scanner.nextDouble(); *// Leer número decimal para compra*

```

System.out.print("Ingrese la distancia a recorrer (km): ");
distancia = scanner.nextDouble(); // Leer número decimal para distancia

// Calcular costo de despacho según reglas de negocio
if (totalCompra > 50000) {
    // Despacho gratis para compras sobre 50,000 pesos
    costoDespacho = 0;
    System.out.println("\n¡Despacho gratuito! Compra sobre $50,000");
}
else if (totalCompra >= 25000 && totalCompra <= 49999) {
    // Tarifa de $150 por km para compras entre 25,000-49,999
    costoDespacho = 150 * distancia;
    System.out.println("\nTarifa aplicada: $150 por km");
}
else {
    // Tarifa de $300 por km para compras menores a 25,000
    costoDespacho = 300 * distancia;
    System.out.println("\nTarifa aplicada: $300 por km");
}

// Mostrar resumen del cálculo
System.out.println("Total compra: $" + totalCompra);
System.out.println("Distancia: " + distancia + " km");
System.out.println("Costo despacho: $" + costoDespacho);
System.out.println("Total a pagar: $" + (totalCompra + costoDespacho));

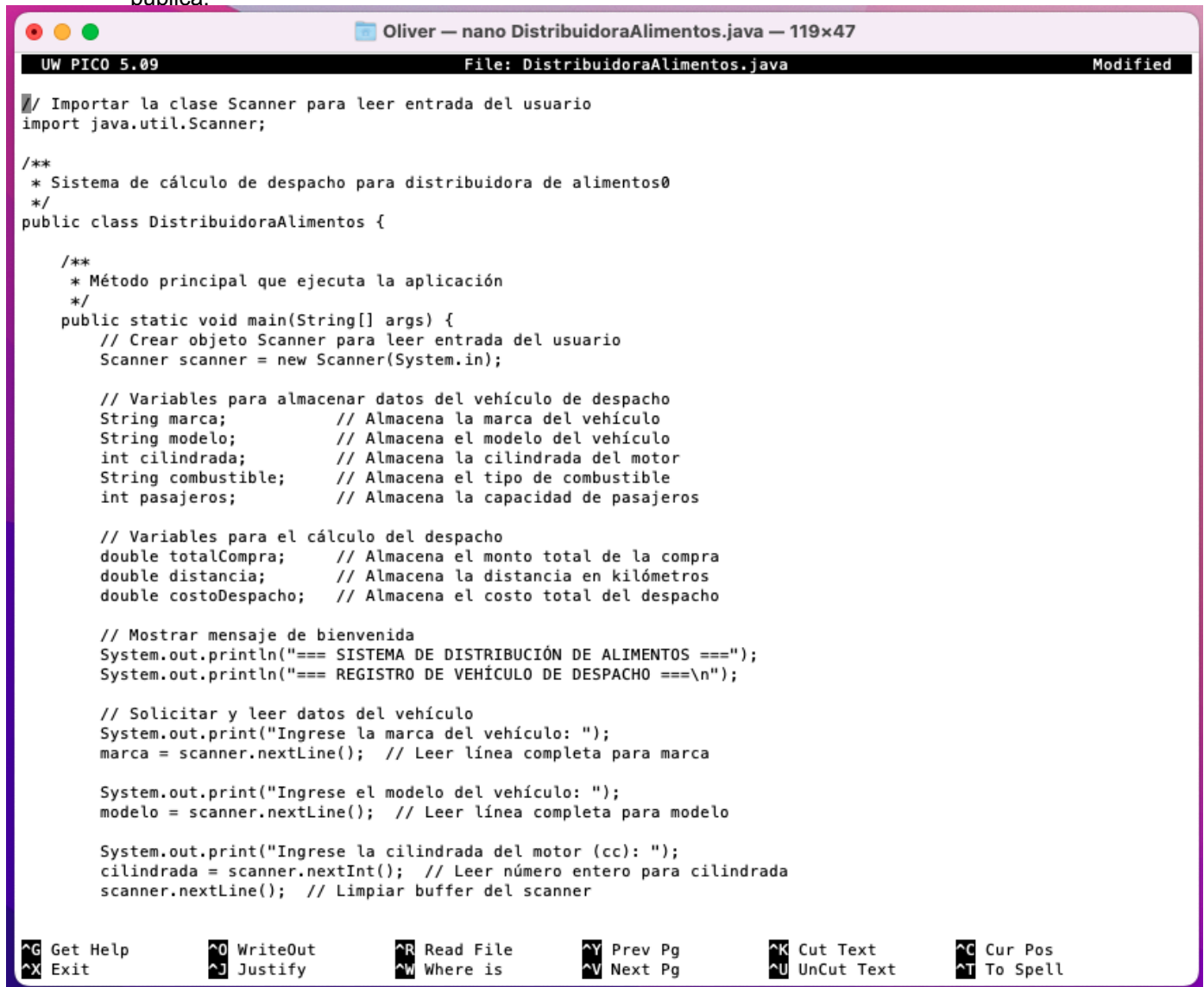
// Cerrar el scanner para liberar recursos
scanner.close();

System.out.println("\n=== PROCESO COMPLETADO ===");
}
}

```

Compilación (para MacOS)

1. Guardar el código en un archivo con extensión *.java* usando editor de texto *nano* dentro de la aplicación *Terminal*. El nombre del archivo debe coincidir exactamente con el nombre de la clase pública.



```
Oliver — nano DistribuidoraAlimentos.java — 119x47
UW PICO 5.09 File: DistribuidoraAlimentos.java Modified

// Importar la clase Scanner para leer entrada del usuario
import java.util.Scanner;

/**
 * Sistema de cálculo de despacho para distribuidora de alimentos
 */
public class DistribuidoraAlimentos {

    /**
     * Método principal que ejecuta la aplicación
     */
    public static void main(String[] args) {
        // Crear objeto Scanner para leer entrada del usuario
        Scanner scanner = new Scanner(System.in);

        // Variables para almacenar datos del vehículo de despacho
        String marca;           // Almacena la marca del vehículo
        String modelo;          // Almacena el modelo del vehículo
        int cilindrada;          // Almacena la cilindrada del motor
        String combustible;     // Almacena el tipo de combustible
        int pasajeros;          // Almacena la capacidad de pasajeros

        // Variables para el cálculo del despacho
        double totalCompra;     // Almacena el monto total de la compra
        double distancia;       // Almacena la distancia en kilómetros
        double costoDespacho;   // Almacena el costo total del despacho

        // Mostrar mensaje de bienvenida
        System.out.println("=== SISTEMA DE DISTRIBUCIÓN DE ALIMENTOS ===");
        System.out.println("=== REGISTRO DE VEHÍCULO DE DESPACHO ===\n");

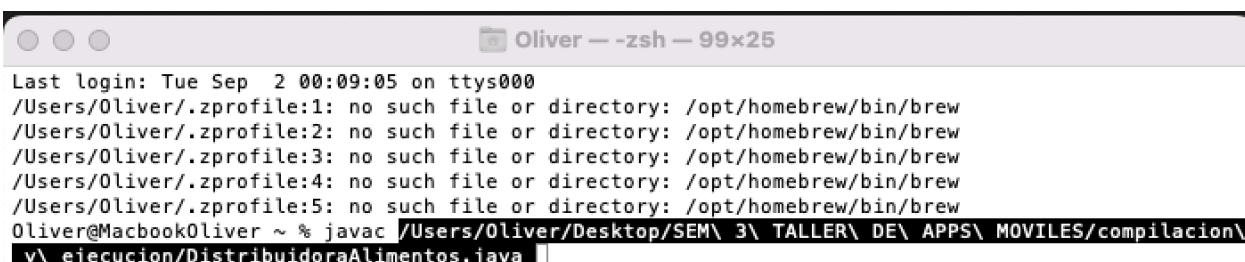
        // Solicitar y leer datos del vehículo
        System.out.print("Ingrese la marca del vehículo: ");
        marca = scanner.nextLine(); // Leer línea completa para marca

        System.out.print("Ingrese el modelo del vehículo: ");
        modelo = scanner.nextLine(); // Leer línea completa para modelo

        System.out.print("Ingrese la cilindrada del motor (cc): ");
        cilindrada = scanner.nextInt(); // Leer número entero para cilindrada
        scanner.nextLine(); // Limpiar buffer del scanner

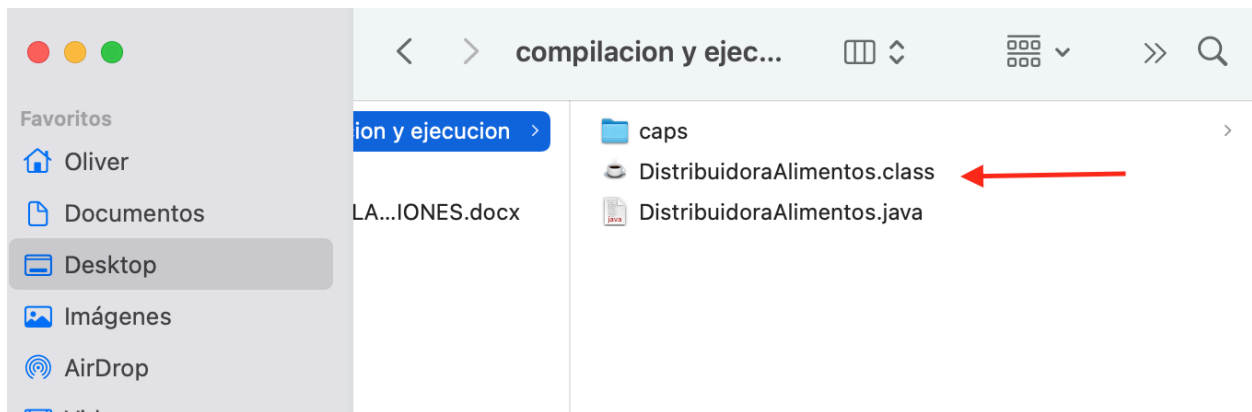
        ^G Get Help      ^O WriteOut      ^R Read File      ^Y Prev Pg      ^K Cut Text      ^C Cur Pos
        ^X Exit          ^J Justify       ^W Where is       ^V Next Pg      ^U UnCut Text    ^T To Spell
```

2. Abrir *Terminal* y navegar al directorio donde se aloja el archivo, luego compilar.



```
Oliver — -zsh — 99x25
Last login: Tue Sep  2 00:09:05 on ttys000
/Users/Oliver/.zprofile:1: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:2: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:3: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:4: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:5: no such file or directory: /opt/homebrew/bin/brew
Oliver@MacbookOliver ~ % javac /Users/Oliver/Desktop/SEM\ 3\ TALLER\ DE\ APPS\ MOVILES\compilacion\
y\ ejecucion\DistribuidoraAlimentos.java
```

- Una vez compilado, se creará el archivo *DistribuidoraAlimentos.class*. Luego, en la *Terminal* navegar hasta el directorio que aloja el archivo *.class* y ejecutar.



```
SEM 3 TALLER DE APPS MOVILES - zsh - 87x37
Last login: Tue Sep  2 00:48:52 on ttys000
/Users/Oliver/.zprofile:1: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:2: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:3: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:4: no such file or directory: /opt/homebrew/bin/brew
/Users/Oliver/.zprofile:5: no such file or directory: /opt/homebrew/bin/brew
Oliver@MacbookOliver ~ % cd "/Users/Oliver/Desktop/SEM 3 TALLER DE APPS MOVILES/"
Oliver@MacbookOliver SEM 3 TALLER DE APPS MOVILES % java DistribuidoraAlimentos
=== SISTEMA DE DISTRIBUCIÓN DE ALIMENTOS ===
=== REGISTRO DE VEHÍCULO DE DESPACHO ===

Ingrese la marca del vehículo: Mazda
Ingrese el modelo del vehículo: 2
Ingrese la cilindrada del motor (cc): 1500
Ingrese el tipo de combustible: Bencina
Ingrese la capacidad de pasajeros: 4

=== DATOS DEL VEHÍCULO REGISTRADO ===
La marca que ha ingresado es: Mazda
El modelo que ha ingresado es: 2
La cilindrada que ha ingresado es: 1500 cc
El tipo de combustible es: Bencina
Tiene una capacidad de 4 pasajeros.

=== CÁLCULO DE COSTO DE DESPACHO ===
Ingrese el total de la compra ($): 45000
Ingrese la distancia a recorrer (km): 8

Tarifa aplicada: $150 por km
Total compra: $45000.0
Distancia: 8.0 km
Costo despacho: $1200.0
Total a pagar: $46200.0

=== PROCESO COMPLETADO ===
Oliver@MacbookOliver SEM 3 TALLER DE APPS MOVILES %
```


Conclusión

La realización de esta actividad permitió fortalecer habilidades en programación Java, especialmente en el manejo de entradas de usuario, estructuras condicionales y diseño de flujos de trabajo aplicados a un caso concreto. Además, fomentó la capacidad de analizar requerimientos, traducirlos en código funcional y validar su correcta ejecución. Este ejercicio no solo enriquece la formación académica y técnica, sino que también acerca al estudiante a escenarios profesionales donde la automatización y la eficiencia operativa son clave en el desarrollo de software.

Bibliografía

- Liang, Y. D. (2019). *Introducción a la programación con Java* (3ª ed.). Boston, Estados Unidos: Prentice Hall, capítulo 2, páginas 38-60.
- Deitel, P., & Deitel, H. (2017). *Java: Cómo programar* (10ª ed.). Ciudad de México, México: Pearson Educación, capítulo 3, páginas 67-95.
- REPOSITORIO:
<https://github.com/Ziguratspa/DistribuidoraAlimentos.git>