



**UNIVERSIDAD TECNOLÓGICA DE PANAMÁ**



**FACULTAD DE INGENIERÍA DE SISTEMAS  
COMPUTACIONALES**

**LICENCIATURA EN CIBERSEGURIDAD**

**PROGRAMACION 1**

**LABORATORIO 2**

**PREPARADO POR**

**ADRIAN JIMENEZ 4-839-2413**

**JUSTIN HE 8-1045-2230**

**A CONSIDERACIÓN DE:**

**NAPOLEON IBARRA**

**GRUPO: 2S3111**

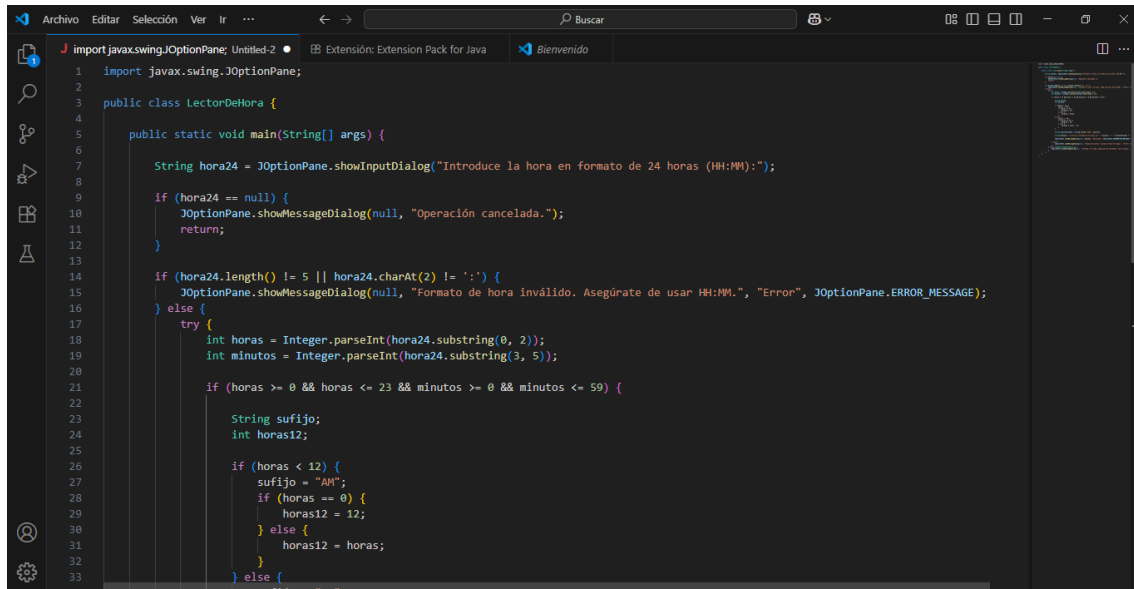
**FECHA: 8-9-25**

## Programa 1: Leer la hora asignada e imprimirla

### Objetivo

Leer la hora que un usuario le asigna al programa desarrollado en notación de 24 horas, para luego imprimirla en 12.

### Código Desarrollado

A screenshot of a Java IDE window titled 'J OptionPane: Untitled-2'. The code is for a class named 'LectorDeHora'. It starts with an import statement for 'javax.swing.JOptionPane'. The 'main' method prompts the user to enter a 24-hour time in HH:MM format. It checks if the input is null, if it's the correct length (5 characters), and if it contains a colon. If any of these checks fail, it shows an error message. If the input is valid, it parses the hours and minutes using 'Integer.parseInt', then converts the hours to 12-hour format and adds 'AM' or 'PM' as a suffix. Finally, it prints the formatted time.

```
1 import javax.swing.JOptionPane;
2 import javax.swing.JOptionPane;
3
4 public class LectorDeHora {
5
6     public static void main(String[] args) {
7
8         String hora24 = JOptionPane.showInputDialog("Introduce la hora en formato de 24 horas (HH:MM:");
9
10        if (hora24 == null) {
11            JOptionPane.showMessageDialog(null, "Operación cancelada.");
12            return;
13        }
14
15        if (hora24.length() != 5 || hora24.charAt(2) != ':') {
16            JOptionPane.showMessageDialog(null, "Formato de hora inválido. Asegúrate de usar HH:MM.", "Error", JOptionPane.ERROR_MESSAGE);
17        } else {
18            try {
19                int horas = Integer.parseInt(hora24.substring(0, 2));
20                int minutos = Integer.parseInt(hora24.substring(3, 5));
21
22                if (horas >= 0 && horas <= 23 && minutos >= 0 && minutos <= 59) {
23
24                    String sufijo;
25                    int horas12;
26
27                    if (horas < 12) {
28                        sufijo = "AM";
29                        if (horas == 0) {
30                            horas12 = 12;
31                        } else {
32                            horas12 = horas;
33                        }
34                    } else {
35                        horas12 = horas - 12;
36                        sufijo = "PM";
37                    }
38
39                    System.out.println(horas12 + ":" + minutos + " " + sufijo);
40                }
41            } catch (NumberFormatException e) {
42                JOptionPane.showMessageDialog(null, "Error: No se puede convertir a número.", "Error", JOptionPane.ERROR_MESSAGE);
43            }
44        }
45    }
46 }
```

### Explicación del código

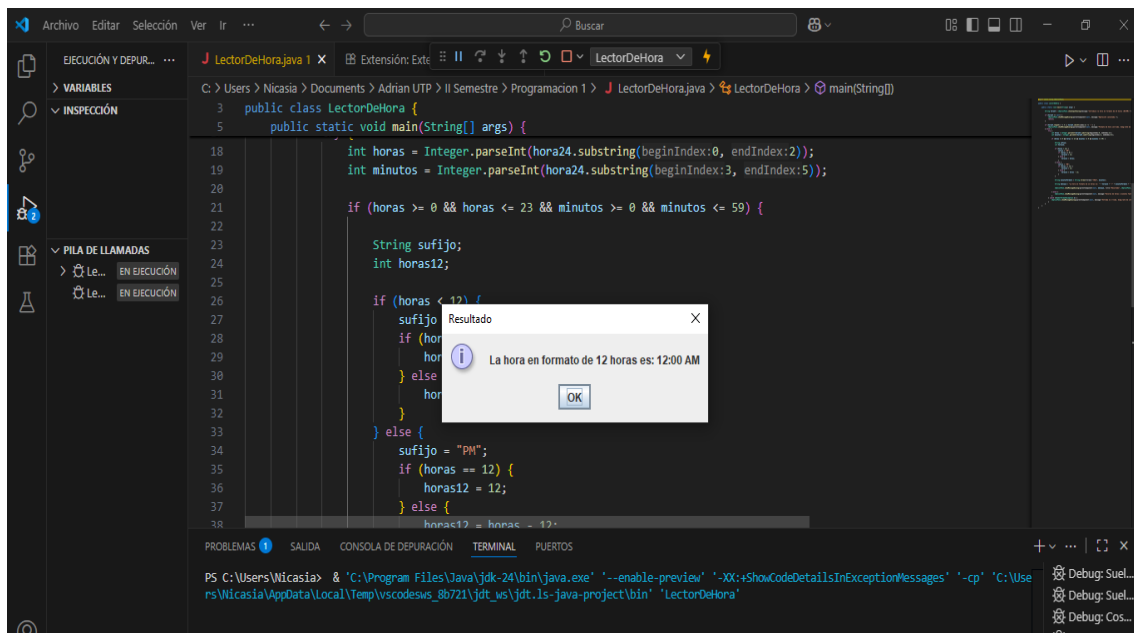
Primeramente se coloca el comando **import javax.swing.JOptionPane** porque es fundamental para se pueda abrir las ventanas a la hora de ejecutar el código.

Se utiliza el comando **JOptionPane.showInputDialog** para pedir la hora al usuario en una ventana. También **JOptionPane.showMessageDialog** para mostrar los mensajes de error en otra ventana si se escribe mal la hora.

También se ingresa el comando **Integer.parseInt** para convertir las horas y los minutos que el usuario escribió a números, para poder hacer los cálculos y **.substring** para cortar el texto de la hora para separar las horas de los minutos.

Y aparte también se utiliza el comando **.length** para verificar que la hora que se ingresó tenga 5 caracteres.

## Ejecución del código



## Programa 2: Costo diario de conducir y ahorro al compartir

### Objetivo

Calcular el costo total diario de usar el automóvil para ir al trabajo y estimar el ahorro por persona si se comparte el viaje. Se consideran combustible, estacionamiento y peajes.

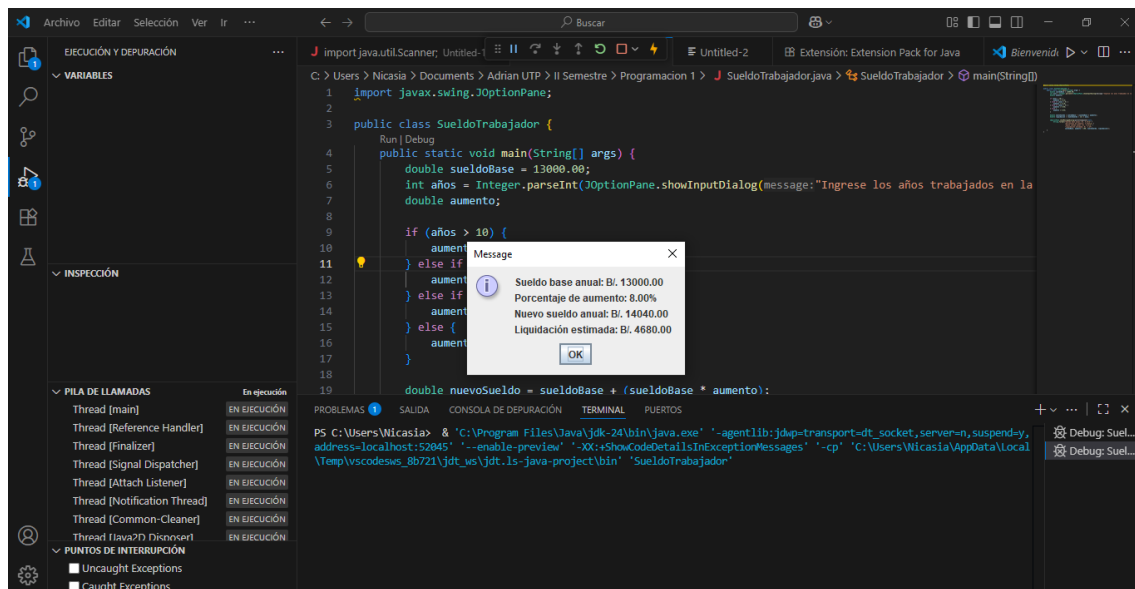
### Código fuente: CostoAuto.java



## Explicación del código

- Se piden los datos al usuario con `JOptionPane.showInputDialog(...)`.
- Se convierten a `double/int` con `Double.parseDouble(...)` y `Integer.parseInt(...)`.
- Se calculan `litrosConsumidos`, `costoCombustible` y `costoTotal`.
- Si hay más de 1 persona, se divide el `costoTotal` entre personas para obtener `costoCompartido`.
- Se calcula el ahorro como la diferencia entre manejar solo y el costo por persona compartiendo.
- Se muestran los resultados con 2 decimales usando `String.format("%.2f")`.

## Ejemplo de prueba rápida

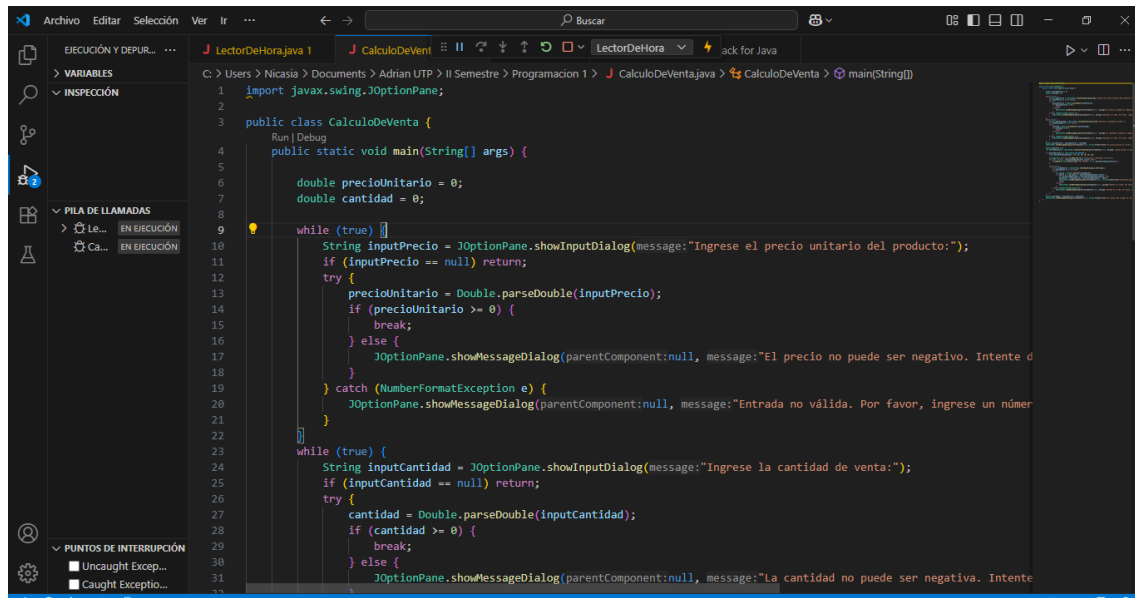


## Programa 3: Calcular precio de un producto

### Objetivo

Calcular el precio de un producto, y agregar un descuento solo si el vendedor lo requiera.

### Código

The image shows a screenshot of an IDE (likely IntelliJ IDEA) with a Java file named 'CalculoDeVenta.java'. The code is as follows:

```
1 import javax.swing.JOptionPane;
2
3 public class CalculoDeVenta {
4     public static void main(String[] args) {
5
6         double precioUnitario = 0;
7         double cantidad = 0;
8
9         while (true) {
10             String inputPrecio = JOptionPane.showInputDialog(message:"Ingrese el precio unitario del producto:");
11             if (inputPrecio == null) return;
12             try {
13                 precioUnitario = Double.parseDouble(inputPrecio);
14                 if (precioUnitario >= 0) {
15                     break;
16                 } else {
17                     JOptionPane.showMessageDialog(parentComponent:null, message:"El precio no puede ser negativo. Intente d
18                 }
19             } catch (NumberFormatException e) {
20                 JOptionPane.showMessageDialog(parentComponent:null, message:"Entrada no válida. Por favor, ingrese un númer
21             }
22         }
23
24         while (true) {
25             String inputCantidad = JOptionPane.showInputDialog(message:"Ingrese la cantidad de venta:");
26             if (inputCantidad == null) return;
27             try {
28                 cantidad = Double.parseDouble(inputCantidad);
29                 if (cantidad >= 0) {
30                     break;
31                 } else {
32                     JOptionPane.showMessageDialog(parentComponent:null, message:"La cantidad no puede ser negativa. Intente
33             }
34         }
```

### Explicación

Usamos **JOptionPane.showInputDialog** para pedir el precio y la cantidad en una ventana emergente. Este método siempre devuelve un **String** o **null** si el usuario cancela.

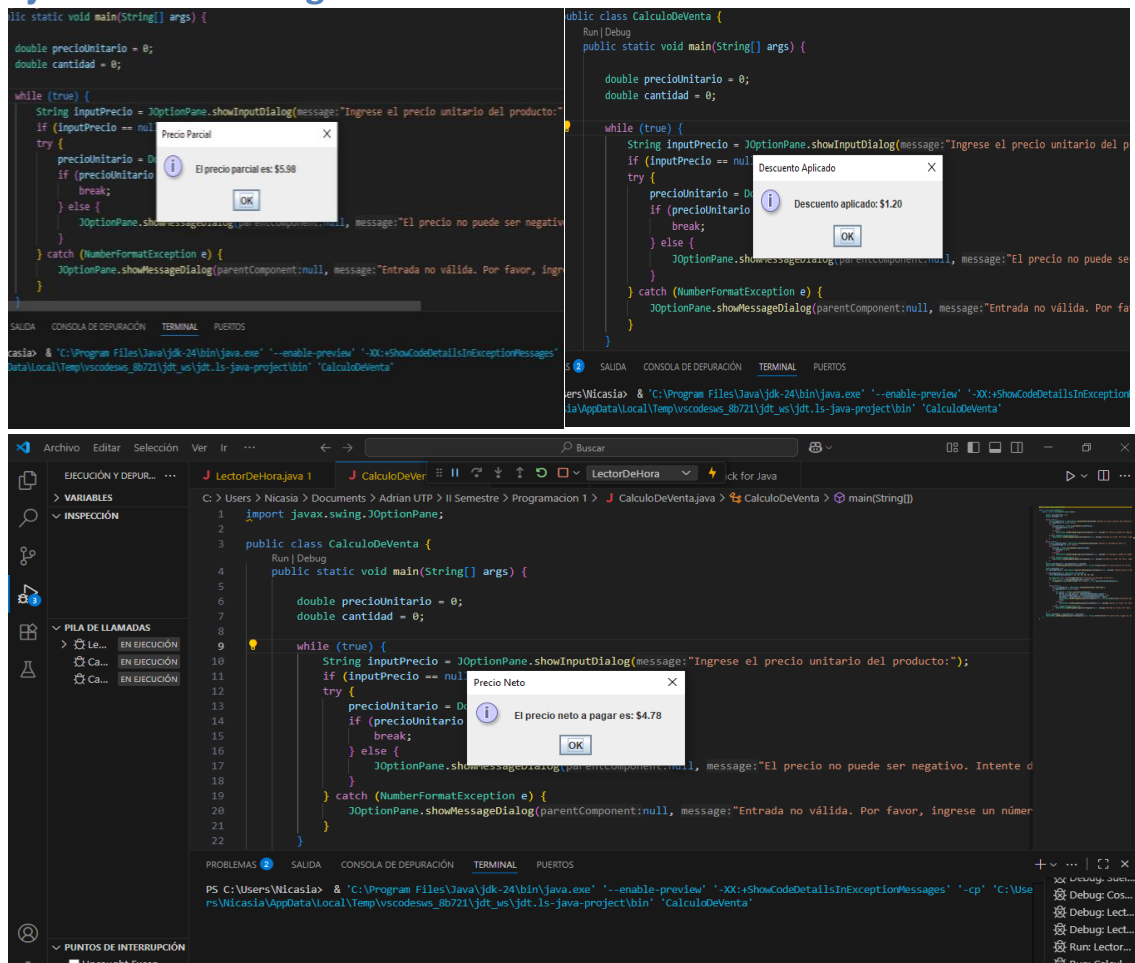
Para mostrar resultados o errores, usamos **JOptionPane.showMessageDialog**. Esto presenta la información en una ventana emergente.

Para preguntar si el usuario quiere un descuento, se utiliza **JOptionPane.showConfirmDialog**, que muestra una ventana con las opciones "Sí" y "No".

Se usa **StringBuilder** para construir el texto del menú de descuentos.

Se mantienen los bucles **while** y los bloques **try-catch** para asegurar que las entradas de precio, cantidad y la opción de descuento sean números válidos.

## Ejecución del código



## Programa 4: Sueldo con aumento y liquidación

### Objetivo

Calcular el nuevo sueldo anual de un trabajador que tiene un sueldo base de B/. 13,000 anuales, aplicando un aumento según la antigüedad y estimar una liquidación simple.

### Criterios de aumento

- Más de 10 años: +15%
- Más de 5 y hasta 10 años: +10%
- Más de 3 y hasta 5 años: +8%
- 3 años o menos: +5%

### Suposición para la liquidación (simplificada)

Se asume 1 sueldo mensual por cada año trabajado: liquidación =  $(\text{nuevoSueldo} / 12) \times \text{años}$ . Esta fórmula es didáctica y simplificada para el ejercicio.

## Código fuente: SueldoTrabajador.java

```
1 import javax.swing.JOptionPane;
2
3 public class SueldoTrabajador {
4     public static void main(String[] args) {
5         double sueldoBase = 13000.00;
6         int años = Integer.parseInt(JOptionPane.showInputDialog("Ingrese los años trabajados en la empresa:"));
7         double aumento;
8
9         if (años > 10) {
10             aumento = 0.15;
11         } else if (años > 5) {
12             aumento = 0.10;
13         } else if (años > 3) {
14             aumento = 0.08;
15         } else {
16             aumento = 0.05;
17         }
18
19         double nuevoSueldo = sueldoBase + (sueldoBase * aumento);
20         double liquidacion = (nuevoSueldo / 12) * años;
21
22         JOptionPane.showMessageDialog(null,
23             String.format("Sueldo base anual: B/. %.2f\n" +
24                 "Porcentaje de aumento: %.2f%%\n" +
25                 "Nuevo sueldo anual: B/. %.2f\n" +
26                 "Liquidación estimada: B/. %.2f",
27                 sueldoBase, aumento * 100, nuevoSueldo, liquidacion));
28     }
29 }
30
```

## Explicación pasó a paso

- Se solicita la antigüedad (años) vía JOptionPane.
- Según el valor de años, se asigna el porcentaje de aumento con if / else-if / else.
- Se calcula el nuevo sueldo anual:  $\text{nuevoSueldo} = \text{sueldoBase} + (\text{sueldoBase} \times \text{aumento})$ .
- Se estima la liquidación:  $(\text{nuevoSueldo} / 12) \times \text{años}$ .
- Se muestran los resultados formateados a 2 decimales.

## Ejemplo de prueba rápida

