

# Fundamentos de Programación Orientada a Eventos



Universidad  
del Valle

## VALIDACIONES Y EXCEPCIONES EN JAVA

Docente: Ing. JESUS A. GONZALEZ M.

Correo: [jesus.a.gonzalez@correounivalle.edu.co](mailto:jesus.a.gonzalez@correounivalle.edu.co)



# CONTENIDO

- ✓ Validaciones (Controlando errores antes de que ocurran)
- ✓ Excepciones



# ¿Qué son las validaciones?

## Definición:

- Verificación previa a una operación para asegurar que los datos son válidos.





## ¿Por qué validar?

- Evita caídas inesperadas del programa.
- Mejora la experiencia del usuario.
- Protege la lógica del negocio.





# Tipos comunes de validación

-  Null checks: Evitar NullPointerException (acceder a posiciones inexistentes en una estructura de datos)
-  Rango numérico: Edad, precios, cantidades
-  Formato de texto: Correos, contraseñas, códigos
-  Existencia o unicidad: Evitar duplicados o buscar elementos





# Ejemplo 1 – Validar una edad

```
if (edad < 0 || edad > 120) {  
    System.out.println("Edad no válida.");  
}
```

```
if (edad < 18) {  
    System.out.println("Edad no válida.");  
}
```





## Ejemplo 2 – Validación con expresiones regulares

```
public class Prueba{

    public static void main(String arg[]){

        String email = "user@gmail.com";
        if (!email.matches("^[\\w.-]+@[\\w.-]+\\.[a-zA-Z]{2,}$")) {
            System.out.println("Correo no válido");
        }else{
            System.out.println("Correo válido");
        }

    }

}
```



## Ejemplo 2 – Validación con expresiones regulares

Explicación de la Expresión regular.

```
matches("^[\w.-]+@[\w.-]+\.[a-zA-Z]{2,}$")
```

- ^ y \$ indican inicio y fin de la cadena.
- \w representa letras, números o guiones bajos.
- + significa uno o más.
- [a-zA-Z]{2,} requiere al menos 2 letras para el dominio (.com, .org...).



# ¿Qué es una excepción en Java?

Una excepción es un evento que interrumpe el flujo normal del programa.

Permite manejar errores sin que el programa se detenga abruptamente.

Ejemplo: división entre cero, archivos no encontrados, datos inválidos.





# ¿Qué puede causar una excepción?

- Dividir entre cero (ArithmeticException)
- Acceder a un índice inválido (ArrayIndexOutOfBoundsException)
- Usar un objeto null (NullPointerException)
- Leer un archivo inexistente (FileNotFoundException)
- Convertir una cadena de texto en número (NumberFormatException)





# Tipos de excepciones

## 1. Checked Exceptions (**verificadas**):

- El **compilador obliga** a manejarlas.
- Ej: IOException, SQLException

## 2. Unchecked Exceptions (**no verificadas**):

- **No es obligatorio** manejarlas.
- Ej: NullPointerException, IllegalArgumentException





# Estructura básica try-catch-finally

```
try {  
    int resultado = 10 / 0;  
  
} catch (ArithmeticException e) {  
    System.out.println("No puedes dividir entre cero.");  
  
} finally { //Esta parte es opcional  
    System.out.println("Esto siempre se ejecuta.");  
}
```





# ¿Por qué son importantes las excepciones?

- ✓ Permiten que el programa siga funcionando ante errores.
- ✓ Mejoran la experiencia del usuario.
- ✓ Ayudan a detectar y depurar errores.
- ✓ Separan lógica de negocio del manejo de errores.





# Validaciones y Excepciones

## Ejercicio

### Ejercicio en clase

Desarrollar un programa que permita a la secretaría de tránsito de Yumboslavia, administrar los vehículos, asociando cada vehículo con su propietario, mediante un documento llamado tarjeta de propiedad.

De los vehículos se requiere manejar su placa compuesta por 3 letras, seguidas por 3 números, la marca y el año de fabricación del mismo. Los propietarios deben registrar su número del documento de identidad, nombres, apellidos y dirección.

En la tarjeta de propiedad se debe registrar el vehículo, el propietario, el código asociado de la misma, la fecha de expedición de la misma (cuyo formato es dd/mm/aaaa).





# Validaciones y Excepciones

## Ejercicio

Para el ejercicio:

- Realizar el diagrama de clases únicamente para las clases MODELO, no incluya los métodos get y set
- Use la arquitectura MVC
- Realice la validaciones necesarias, es decir que donde el dato sea numérico y que se garantice el formato solicitado donde se requiera.





# Mini Proyecto 3

