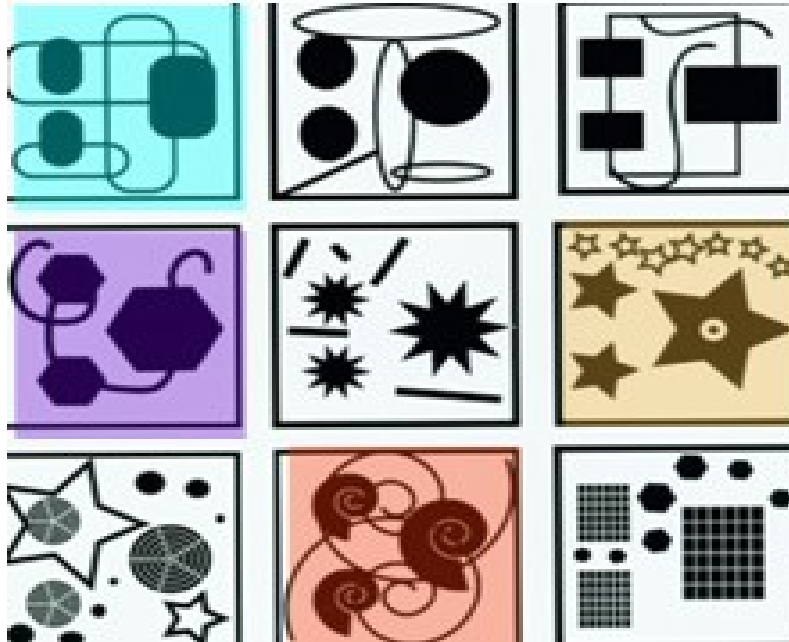


# Tema 4



**Conceptos complementarios  
en Orientación a Objetos**

## Lección 4.3

# **Manejo de excepciones**

# 1. Excepciones

## Excepción:

Situación provocada por un problema durante la ejecución que implica que el programa termine de forma anormal, a no ser que se maneje la excepción.

Ejemplos:

- El usuario introduce un valor erróneo en una variable.
- Se necesita un fichero que no existe.
- Un objeto no puede responder un mensaje en su estado actual.

## Tratamiento/manejo de la excepción:

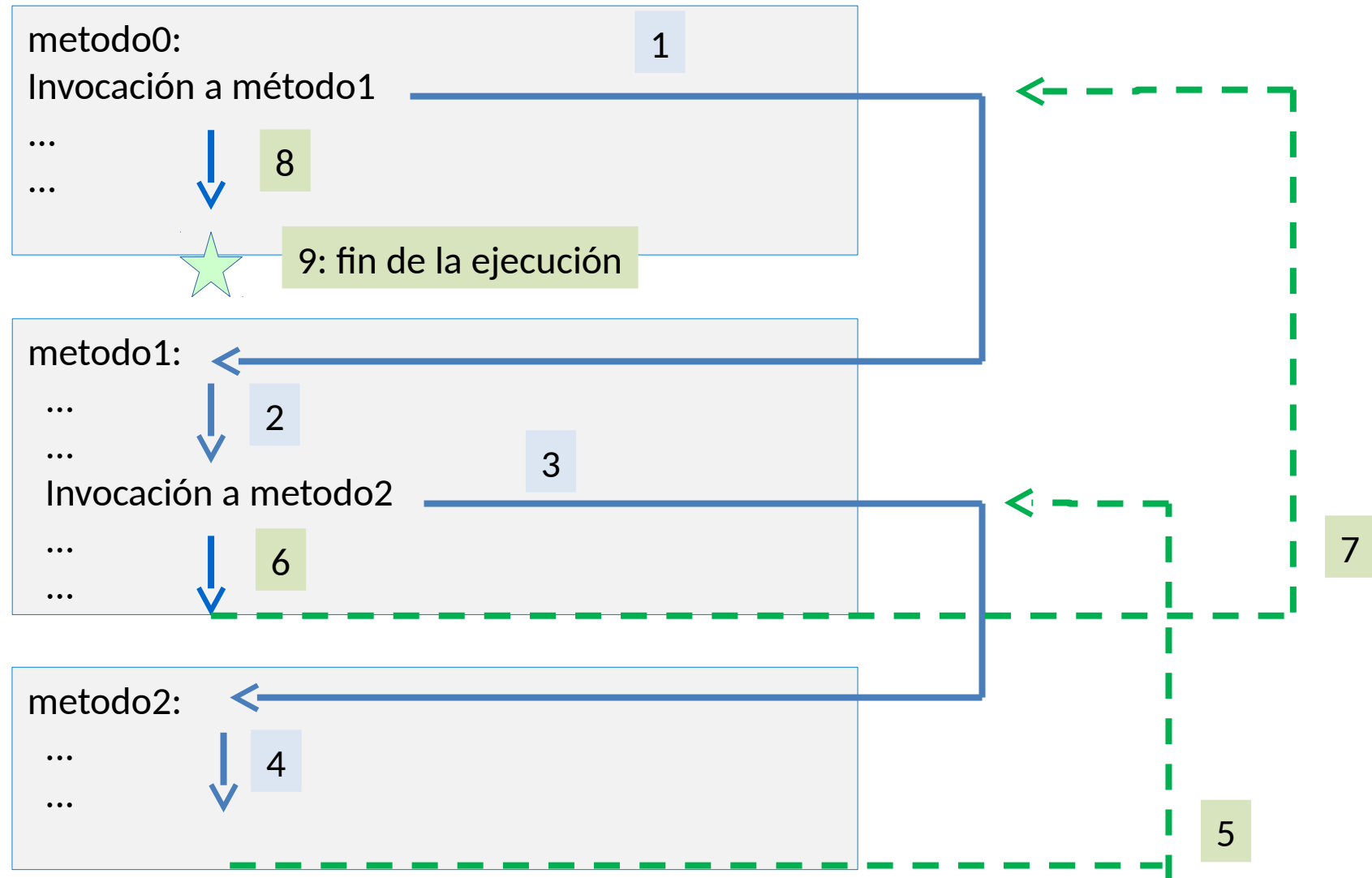
Evita la terminación anormal, detectando el problema y dando una terminación alternativa para recuperarse del error.

Ejemplos:

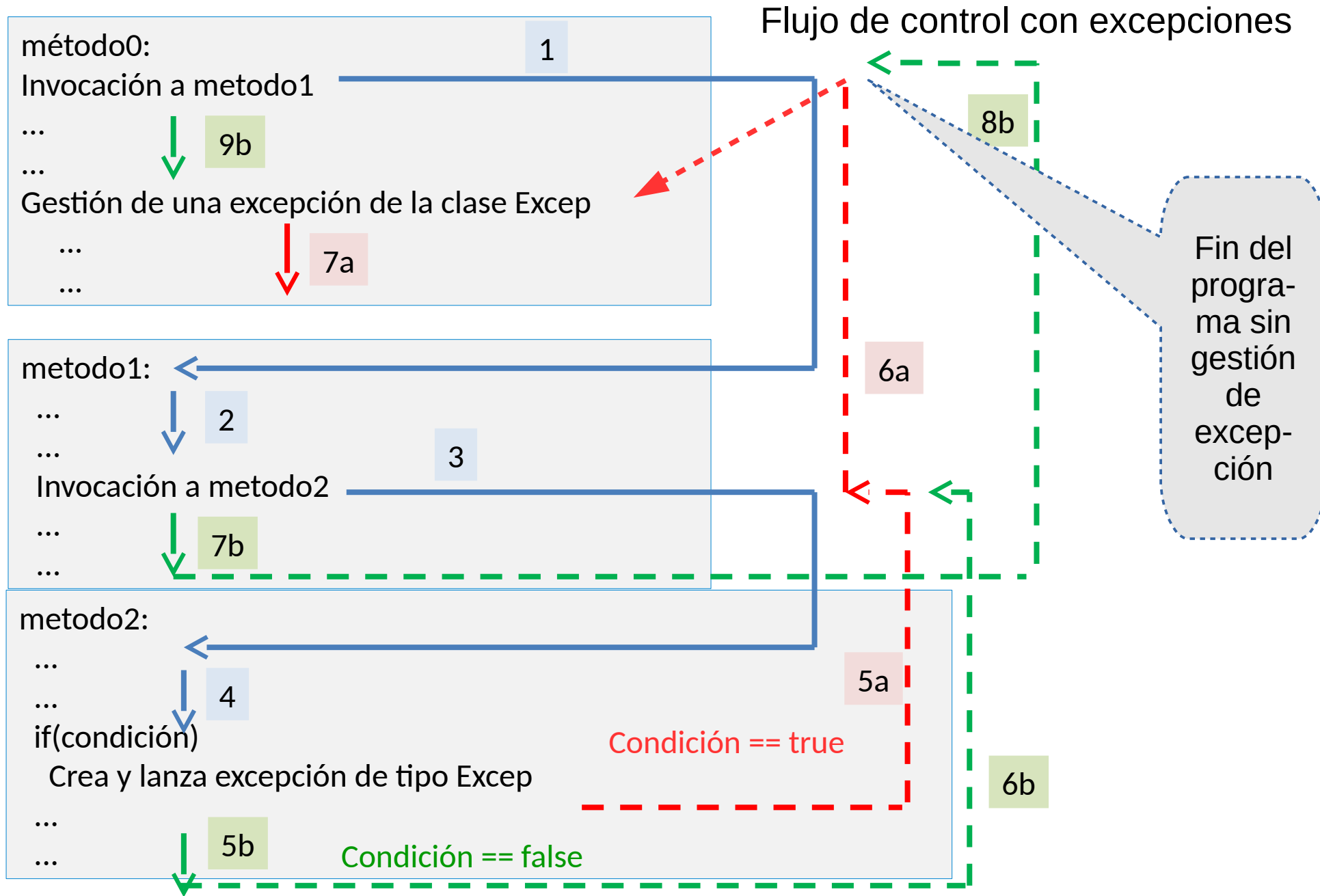
- Se muestra de forma comprensible un mensaje de error.
- Se cambia el valor de la variable o el estado del objeto.

## 2. Envío de mensajes entre objetos

Flujo de control que se produce durante el envío de mensaje




# 3. Esquema del tratamiento de excepciones



## 4. Mecanismos en los lenguajes OO

- Los lenguajes de programación OO proveen de mecanismos para:
  - La definición e instanciación de clases/tipos de Excepciones.
  - El lanzamiento de excepciones, cuando se producen.
  - La captura y tratamiento de excepciones.
  - En algunos casos: avisar en la cabecera que un método propaga determinado tipo de excepciones.



Ejemplo de excepciones en los proyectos:  
ExcepcionesJava y excepciones\_ruby

# 5. Alternativas en Java

```
try{ ...  
    a.metodo();  
    ...  
} catch (UnTipoDeExcepcion e) {  
    // Código a ejecutar cuando se produzca una excepción de tipo  
    // UnTipoDeExcepcion  
    dentro del código try  
} catch (OtroTipoDeExcepcion o){  
    // Código a ejecutar cuando se produzca una excepción de tipo  
    // OtroTipoDeExcepcion dentro del código try  
} finally {  
    // Código que se ejecuta independientemente de que se lance o no una  
    // excepción dentro del código try  
}
```



```
void metodo() throws UnTipoDeExcepcion, OtroTipoDeExcepcion {  
    ...  
    if (algopasa)  
        throw new UnTipoDeExcepcion("mensaje_error1");  
    ...  
    if (otracosapasa)  
        throw new OtroTipoDeExcepcion("mensaje_error2");  
    ...  
}
```

# 6. Alternativas en Ruby

```
begin
  ...
  a.metodo
  ...
rescue UnTipoDeExcepcion => e
  # Código a ejecutar cuando se produzca una excepción de tipo
  UnTipoDeExcepcion dentro del begin anterior
rescue OtroTipoDeExcepcion => o
  # Código a ejecutar cuando se produzca una excepción de tipo
  OtroTipoDeExcepcion dentro del begin anterior
else
  # Código que se ejecuta si no hay errores dentro del begin
ensure
  # Código que se ejecuta siempre
end
```



```
def metodo
  ...
  if (algopasa)
    raise UnTipoDeExcepcion, 'mensaje_error1'
  end
  if (otracosapasa)
    raise OtroTipoDeExcepcion, 'mensaje_error2'
  end
end
```



# 7. Tipos

Además, los lenguajes de programación proveen clases correspondientes a los tipos de excepciones más comunes.

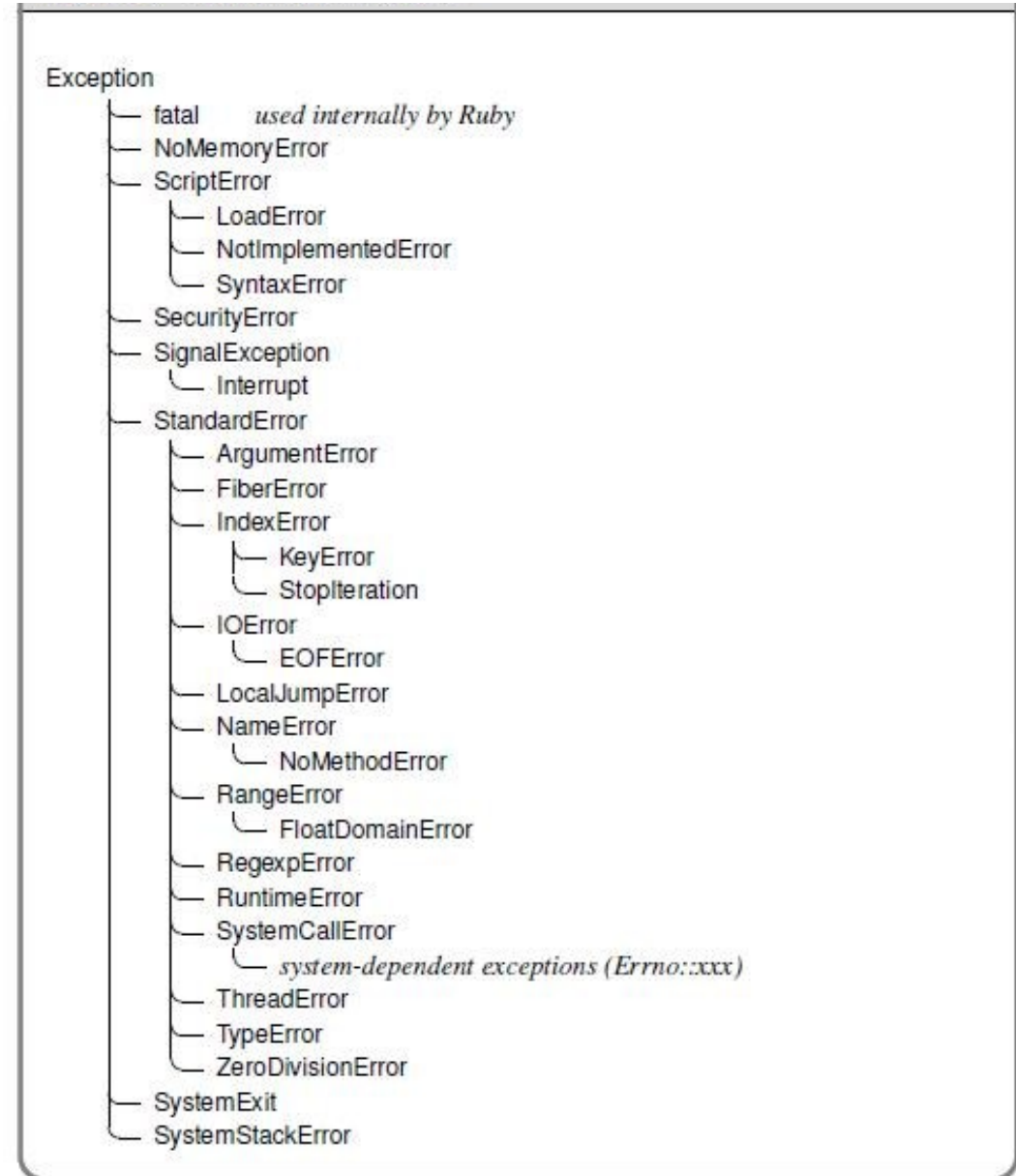
- Ejemplos de algunas clases de excepciones de Java:
  - `ArrayStoreException`
  - `NullPointerException`
  - `ArrayIndexOutOfBoundsException`
  - `NumberFormatException`
  - `ClassNotFoundException`

**Jerarquía de  
excepciones en Java**

# 8. Tipos

Ejemplos de algunas clases de excepciones de Ruby:

- RuntimeError
- NoMethodError
- ArgumentError
- IOError



# 9. Ejemplos Java



Código con excepción **sin** tratamiento definido:

```
int divisor = 0;
int resultado = 100/divisor;
System.out.println("Resultado: " + resultado);
/* Salida: Exception in thread "main"
    java.lang.ArithmeticException: / by zero */
```

Se detecta el error y se termina la ejecución con un mensaje por defecto

Código con excepción **con** tratamiento definido:

```
try {
    int divisor = 0;
    int resultado = 100/divisor;
    System.out.println("Resultado: " + resultado);
} catch (ArithmeticException ex) {
    System.out.println("Error, dividiendo por cero");
    // Salida: Error, dividiendo por cero
    System.out.print(ex.getMessage());
    // Salida: / by zero
}
```

Se detecta el error y se pasa a su tratamiento

# 10. Ejemplos en Ruby

## Código con error **sin** tratamiento definido:

```
divisor = 0
resultado = 100/divisor
puts "Resultado: " + resultado
# Salida: ZeroDivisionError: divided by 0
```

Se produce el error y se termina la ejecución con un mensaje de error por defecto

## Código con error **con** tratamiento definido:

```
begin
  divisor = 0
  resultado = 100/divisor
  puts "Resultado:" + resultado
rescue ZeroDivisionError => ex
  ERR.puts "Error, dividiendo por cero"
end

# Salida: Error, dividiendo por cero

puts ex.message
# Salida: divided by 0
```

Se detecta el error y se pasa a su tratamiento, al estar éste definido



# Pruebas



Para afianzar y comprender mejor los conceptos aprendidos en esta lección haz pruebas con los siguientes ejemplos en Java y Ruby:

- Ejemplos de comparadores: ComparacionJava y comparacion\_ruby
- Ejemplos de excepciones: ExcepcionesJava y excepciones\_ruby