85 - Excepciones - lanzar una excepción mediante comando throw

Listado completo de tutoriales

Hemos visto hasta ahora cual es la sintaxis para capturar excepciones que lanzan métodos. Ahora veremos como podemos lanzar una excepción para que sea eventualmente capturada posteriormente.

Hay que definir con juicio que métodos de una clase deben lanzar excepciones cuando el dato es incorrecto debido que luego se hace más difícil consumir dicho método.

Aquellas partes críticas de nuestras clases pueden lanzar excepciones para que sean más robustas. Por ejemplo si tenemos una clase 'ClienteBanco' y queremos controlar que nunca pueda sacar más dinero del que tiene depositado, el método extraer puede lanzar una excepción evitando que quede en negativo el monto del cliente.

Problema:

Declarar una clase llamada 'PersonaAdulta' con los atributos nombre y edad. Lanzar una excepción de tipo IOException en caso que llegue en el constructor una edad menor a 18 años.

En este problema hemos tomado la decisión de validar y lanzar una excepción en caso que se intente la creación de un objeto de la clase PersonaAdulta con una edad inferior a 18. Esto es porque a juicio del programador considera que es muy importante que nunca haya un objeto de tipo PersonaAdulta que sea menor de edad.

Programa:

```
public class PersonaAdulta {
    private String nombre;
    private int edad;

    public PersonaAdulta(String nombre, int edad) throws Exception {
        this.nombre = nombre;
        if (edad < 18)
            throw new Exception("No es adulta la persona " + nombre
        this.edad = edad;
    }

    public void fijarEdad(int edad) throws Exception {
        if (edad < 18)
```

```
            throw new Exception("No es adulta la persona " + nombre
        this.edad = edad;
    }

    public void imprimir() {
        System.out.println(nombre + " - " + edad);
    }

    public static void main(String[] ar) {
        try {
            PersonaAdulta persona1 = new PersonaAdulta("Ana", 50);
            persona1.imprimir();
            PersonaAdulta persona2 = new PersonaAdulta("Juan", 13);
            persona2.imprimir();
```

Para lanzar una excepción debemos utilizar la palabra clave 'throw' y seguidamente la referencia de un objeto de la clase 'Exception' o de una subclase de 'Exception':

```
        if (edad < 18)
            throw new Exception("No es adulta la persona " + nombre +
                                " porque tiene " + edad + " años.");
```

En el método que utilizamos el comando 'throw' debemos enumerar en su declaración luego de la palabra clave 'throws' los nombres de excepciones que puede lanzar dicho método (pueden ser más de uno):

```
    public PersonaAdulta(String nombre, int edad) throws Exception {
```

De forma similar el método 'fijarEdad' verifica si la edad que llega es menor a 18 para lanzar la excepción:

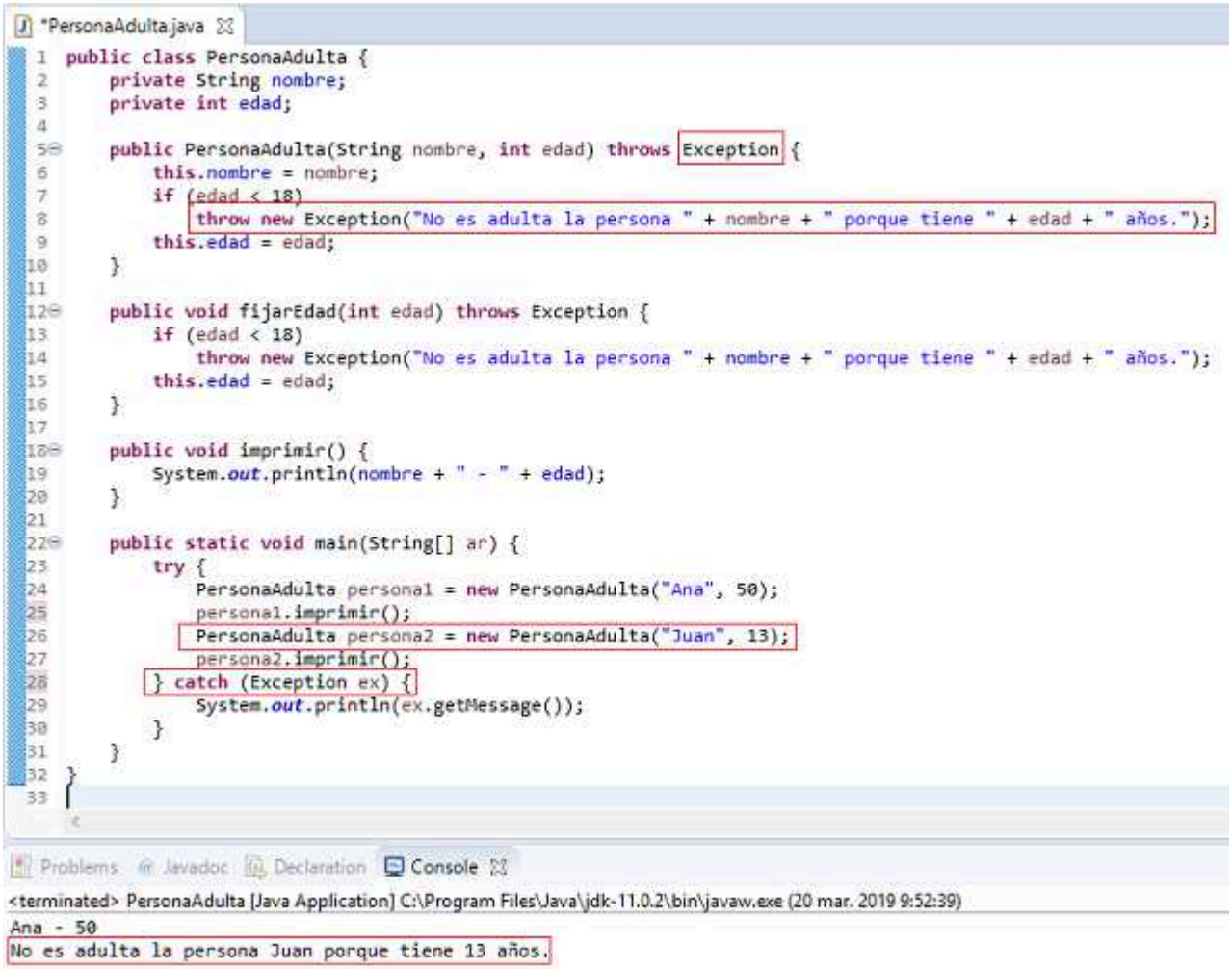
```
    public void fijarEdad(int edad) throws Exception {
        if (edad < 18)
            throw new Exception("No es adulta la persona " +
                                nombre + " porque tiene " + edad + " a
        this.edad = edad;
    }
```

Ahora cuando creamos objetos de la clase 'PersonaAdulta' debemos capturar obligatoriamente la excepción mediante un bloque try/catch:

```
    public static void main(String[] ar) {
        try {
            PersonaAdulta persona1 = new PersonaAdulta("Ana", 50);
            persona1.imprimir();
            PersonaAdulta persona2 = new PersonaAdulta("Juan", 13);
            persona2.imprimir();
```

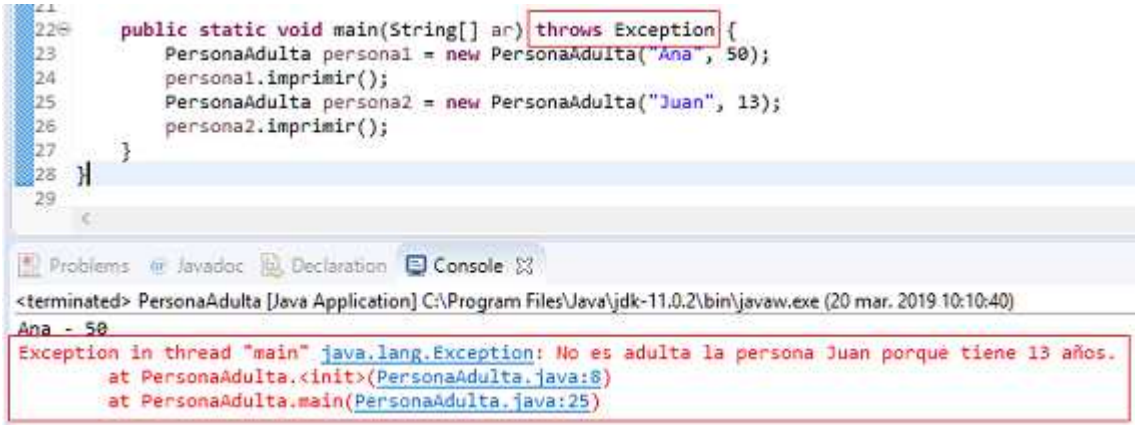
```
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }
```

Cuando creamos el objeto 'persona2' se captura la excepción debido que "Juan" tiene 13 años:



Cuando se ejecuta el comando throw no se continúan ejecutando las instrucciones del método, sino que se aborta su ejecución y vuelve al método que lo llamó.

Si cuando creamos un objeto de la clase 'PersonaAdulta' el método main omite atraparlas, luego la máquina virtual de Java detiene el programa informando la excepción atrapada:



Problemas propuestos

1. Implementar la clase Operaciones. Debe tener dos atributos enteros. Desarrollar los métodos para calcular su suma, resta, multiplicación y división, imprimir dichos resultados. Lanzar una excepción en el método que calcula la división si el segundo valor es cero.

2. Declarar una clase 'Cliente' que represente un cliente de un banco. Definir los siguientes atributos y métodos:

```
Cliente
    atributos
        nombre
        monto
    métodos
        constructor
        depositar
        extraer
        imprimir
```

Generar una excepción si se intenta extraer más dinero del que tiene el atributo 'monto'.

Solución

Retornar