86 - Excepciones propias

Listado completo de tutoriales

La librería o API de Java proporciona gran cantidad de clases que manejan casi cualquier tipo de excepción, pero no estamos obligados a utilizar solo esas clases sino que podemos crear nuestras propias excepciones.

Si queremos crear una excepción no verificada debemos heredar de RuntimeException y si queremos que sea verificada deberemos heredar de Exception o de alguna de sus subclases.

Veamos con un ejemplo como crear una clase que herede de Exception, luego en otra clase lanzar una excepción de la nueva clase creada y finalmente llamar al método que lanza la excepción.

Problema:

Confeccionar una clase que administre una lista tipo pila (se debe poder insertar, extraer e imprimir los datos de la pila).

En el método extraer lanzar una excepción si la pila se encuentra vacía, crear una excepción propia que herede de Exception.

Clase: PilaVaciaException

```
public class PilaVaciaException extends Exception {

    public PilaVaciaException(String mensaje) {
        super("Problema:" + mensaje);
    }

}
```

La clase 'PilaVaciaException' hereda de la clase 'Exception', esto significa que quien lance una excepción de este tipo luego deberá ser capturada en forma obligatoria.

Clase: Pila

```
public class Pila {

    class Nodo {
        int info;
        Nodo sig;
    }
```

```
    private Nodo raiz;

    public Pila() {
        raiz = null;
    }

    public void insertar(int x) {
        Nodo nuevo;
        nuevo = new Nodo();
        nuevo.info = x;
        if (raiz == null) {
            nuevo.sig = null;
            raiz = nuevo;
        } else {
            nuevo.sig = raiz;
            raiz = nuevo;
        }
    }

    public int extraer() throws PilaVaciaException {
```

El método 'extraer' lanza una excepción de tipo 'PilaVaciaException' en caso que la pila se encuentre vacía:

```
    public int extraer() throws PilaVaciaException {
        if (raiz != null) {
            int informacion = raiz.info;
            raiz = raiz.sig;
            return informacion;
        } else {
            throw new PilaVaciaException("No hay mas elementos en la p
        }
    }
```

En la main donde creamos un objeto de la clase 'Pila' insertamos 3 enteros:

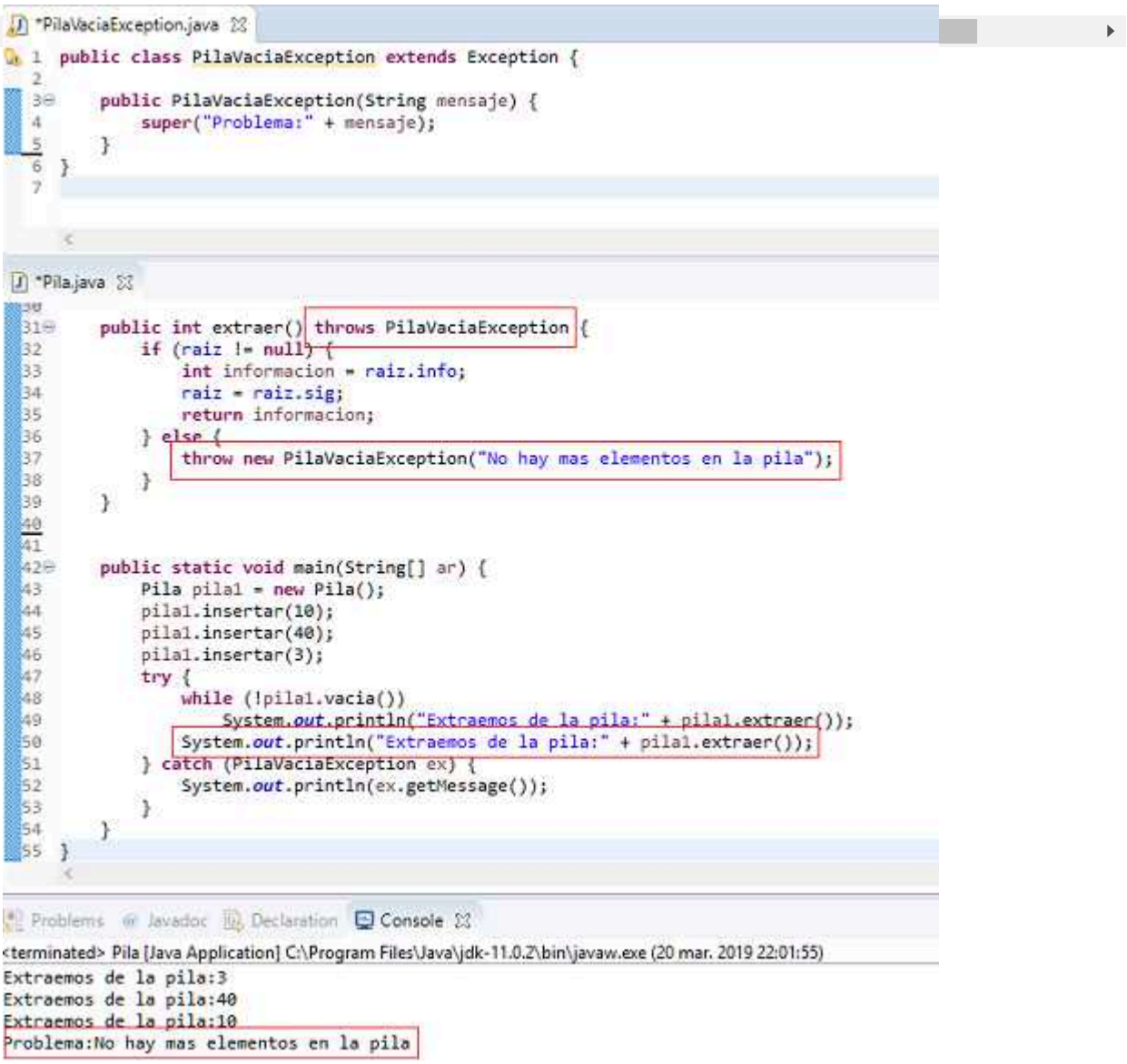
```
        Pila pila1 = new Pila();
        pila1.insertar(10);
        pila1.insertar(40);
        pila1.insertar(3);
```

Seguidamente dentro de un bloque try/catch atrapamos si ocurre una excepción de tipo 'PilaVaciaException', esto ocurre cuando finaliza el while y volvemos a llamar al método extraer, siendo que la pila se encuentra vacía:

```
        try {
            while (!pila1.vacia())
                System.out.println("Extraemos de la pila:" + pila1.ext
            System.out.println("Extraemos de la pila:" + pila1.extraer
```

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

Tenemos una salida similar a esta:



Problema propuesto

1. Plantear una clase llamada 'Termometro' que defina un atributo llamado temperatura. Lanzar una excepción propia llamada 'TemperaturaFueraRangoException' si se intenta fijar una temperatura con un valor menor a -192 o superior a 100.

Solución

Retornar