



UNIVERSIDAD DE EL SALVADOR  
FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE  
DEPARTAMENTO DE INGENIERIA.  
Análisis Numérico ciclo I/2011

Docente: Ing. Carlos Arturo Ruano

Instructor: Br. Oscar G. Rodríguez

## MANEJO DE EXCEPCIONES

Ya tenemos creadas la clase Fraccion y Matriz pero no hemos tomado en cuenta errores que pueden surgir, por ejemplo al querer crear una fracción con denominador igual a cero o querer sumar dos matrices de dimensiones distintas, es aquí donde crearemos excepciones para nuestras clases.

Las excepciones en Java están destinadas, al igual que en el resto de los lenguajes que las soportan, son para la detección y corrección de errores. Si hay un error, la aplicación no debería morir. Se debería lanzar (*throw*) una excepción que nosotros deberíamos capturar (*catch*) y resolver la situación de error.

Lo haremos utilizando **throw** y para la implementación el **try** y el **catch**.

Lo primero que debemos hacer es crear una clase (dentro del paquete donde está la clase donde puede generarse la excepción) con el nombre que le daremos a nuestra excepción dicha clase heredará de la clase Exception, en el constructor recibirá una cadena con el error y esa misma cadena la mandará a su clase padre (Exception) a través del **super**.

La clase quedará de la siguiente forma:

```
6  package Frac;
7
8  /**
9   *
10  * @author mipe
11  */
12 public class FraccionFormatException extends Exception {
13     public FraccionFormatException(String ex){
14         super(ex);
15     }
16 }
17
```



Ahora que ya tenemos nuestra clase **FraccionFormatException** vamos a hacer uso de ella donde lo necesitemos en este caso será cuando el formato creado de una fracción sea incorrecto, el caso en el que una fracción puede ser incorrecta es que su denominador sea igual a cero. Así que la aplicaremos en el `parseFraccion`.

Para llamar la excepción si se cumple la condición incorrecta allí la llamaremos a travez del `throw` y le mandaremos en una cadena el error.

```
throw new FraccionFormatException("Denominador igual a cero");
```

De igual forma lo llamaremos en dado caso q al convertir la él numerador o el denominador a Long de error, o si al ingresar el String tiene una forma diferente a una fracción.

Nuestra función `parseFracción` quedara de la siguiente forma:

```
107 public static Fraccion parseFraccion(String a) throws FraccionFormatException {
108     String f[] = a.split("/");
109     try {
110         Fraccion resultado;
111         if (f.length == 2) {
112             if (!f[1].equals("0")) {
113                 resultado = new Fraccion(Long.parseLong(f[0]), Long.parseLong(f[1]));
114                 return resultado;
115             } else {
116                 throw new FraccionFormatException("Denominador igual a cero");
117             }
118         } else if (f.length == 1) {
119             resultado = new Fraccion(Long.parseLong(f[0]), 1);
120             return resultado;
121         } else {
122             throw new FraccionFormatException("formato fraccion incorrecto");
123         }
124     } catch (NumberFormatException ex) {
125         throw new FraccionFormatException("formato fraccion incorrecto");
126     }
127 }
128
129 }
```



A la firma de la función le colocamos **throws FraccionFormatException** que indica que en esa función se puede generar la excepción **FraccionFormatException**.

Ahora cuando hagamos uso del `parseFraccion` lo haremos encerrando el código en un **try** y lo que hara si se genera la excepción en un **catch**, ejemplo:

```
20 public class Main {
21
22     /**
23      * @param args the command line arguments
24      */
25     public static void main(String[] args) {
26         try {
27             String a = JOptionPane.showInputDialog("ingrese una fraccion de la forma num/den");
28             JOptionPane.showMessageDialog(null, "la fraccion ingresada es: \n" + Fraccion.parseFraccion(a));
29         } catch (FraccionFormatException ex) {
30             JOptionPane.showMessageDialog(null, ex.toString());
31         }
32     }
33
34
35 }
36
```

Cree la porción de código anterior para terminar de comprender el funcionamiento. Y pruebe ingresando: 5/2, 5/s, 4/0, 7/8/8.

Ahora proceda a manejar los siguientes errores:

- En el constructor de la clase `Fraccion` no pueda ingresar el denominador igual a cero.
- Que no pueda sumar dos matrices de diferente dimensión.
- Que para multiplicar dos matrices filas de `a` sea igual a columnas de `b` y columnas de `a` sea igual a filas de `b`.
- Que para invertir una matriz debe de ser cuadrada.
- Y todas las excepciones que considere necesarias en matriz y fracción.