Tarea Independiente 22/09/2025

David Núñez Franco

September 24, 2025

Inventario de Conceptos Claves

- FP en java como un "parche"
- Modelaje de tipos de lambdas como objetos: Function, BiFunction, Predicate, Supplier, Consumer.
- Noción de Stream¡T¿ como "generador/iterador" de objetos de tipo T. Es lazy consumible una sola vez. Hay subclases para tipos concretos tal como lo IntStream.
- Principales "combinadores" con ejemplos
  - map, filter, reduce
  - Interfaz funcional (y métodos default). También se le dice SAM (acrónimo de single abstract method interface)

Ejercicio 0

```
Logre tipos adecuados de forma que foo reciba un tipo correcto. Ud. debe sustituir ?_0
?_0 foo = (?_1 a, ?_2 f) -> a.map(x -> x**2 - 2*x + 1 == 0).filter(r -> f.checkItOut(r
```

Solución

```java
import java.util.function.BiFunction;
import java.util.stream.Stream;

public class Ejercicio0 {
    @FunctionalInterface
    interface Checker {
        boolean checkItOut(Boolean b);
    }

    public static void main(String[] args) {
```

```
        // definimos foo
        BiFunction<Stream<Integer>, Checker, Stream<Boolean>> foo =
        (Stream<Integer> a, Checker f) ->
        a.map(x -> x * x - 2 * x + 1 == 0)
        .filter(r -> f.checkItOut(r));

        // ejemplo: lista de enteros
        Stream<Integer> numbers = Stream.of(1, 2, 3, 4, 5);

        // CHECKER que solo deja pasar true
        Checker cheker = r -> r;

        // aplicamos foo
        Stream<Boolean> result = foo.apply(numbers, cheker);

        // consumimos el stream
        result.forEach(System.out::println);
    }
}
```

Listing 1: Soluc. en JAVA

Ejercicio 1

Implemente un factorial funcional en Java-FP.

Solución

```
// Implementa el factorial de un numero entero positivo (utilizando Streams + reduce).
import java.util.stream.IntStream;

public class Ejercicio1 {

    public static long factorial(int n) {
        if (n < 0) {
            throw new IllegalArgumentException("n must be >= 0");
        }

        // Genera el rango [1, n] inclusive como stream de enteros, y reduce multiplicando cada elemento. Valor inicial
        // de 1
        return IntStream.rangeClosed(1, n)
```

```
14                  .reduce(1, (a, b) -> a * b);
15              }
16
17              public static void main(String[] args) {
18                  System.out.println("0! = " + factorial(0));
19                  System.out.println("1! = " + factorial(1));
20
21                  System.out.println("5! = " + factorial(5));
22                  System.out.println("10! = " + factorial(10));
23              }
24          }
```

Listing 2: Soluc. en Java