hacer lectura superficial de todo esto

¿QUÉ ES PROGRAMACIÓN FUNCIONAL?.

Un lenguaje de programación sigue uno o varios paradigmas de programación

¿Qué es un paradigma de programación?
Un paradigma de programación representa un enfoque particular o filosofía para diseñar soluciones.
https://es.wikipedia.org/wiki/Paradigma_de_programaci%C3%B3n

¿Qué paradigmas de programación existen?
Hay dos grandes grupos: programación imperativa y declarativa. Estos a su vez pueden tener variantes

- **Programación imperativa** o por procedimientos: Es el más usado. Se basa en escribir un conjunto de instrucciones que indican al ordenador "como" resolver el problema. La programación imperativa es la más usada y la más antigua, el ejemplo principal es el Lenguaje de máquina. Hay varios subparadigmas de programación imperativa pero los dos más importantes son:
  - Programación estructurada: es el paradigma que nos habla de cómo usar las 3 estructuras básicas de control de flujo (secuencial, condicional y repetitiva) para escribir algoritmos. Ejemplos de lenguajes puros de este paradigma serían el C, BASIC o Pascal.
  - Programación orientada a objetos: es el paradigma que usa objetos para escribir algoritmos. La POO a su vez utiliza la programación estructurada fundamentalmente para escribir los métodos(funciones). Ejemplos de lenguajes que siguen este paradigma son C++ o Java, pero de forma totalmente puro sería Smalltalk.
- **Programación declarativa:** Está basado en describir el problema declarando propiedades y reglas que deben cumplirse, en lugar de instrucciones. Dentro de la programación declarativa hay varios subparadigmas, los dos más sonoros son:
  - Programación funcional: basada en la definición de los predicados y es de corte más matemático. Haskell es un lenguaje de programación funcional puro.
  - Programación lógica: basado en la definición de relaciones lógicas, está representado por Prolog.
  - sin llegar a pertenecer a ningún subparadigma siendo simplemente "declarativos" tenemos: SQL y otros lenguajes de acceso a BD, expresiones regulares y otros.

A menudo se describe la diferencia entre imperativa y declarativa de la siguiente manera:

La programación imperativa indica "cómo" (how) se hacen las cosas para obtener un resultado y la programación declarativa se dedica a decir "que" (what) que queremos sin decir cómo tenemos que hacer para obtener un resultado.

Para que entiendas la diferencia entre imperativo y declarativo, piensa en SQL. SQL No es un lenguaje de programación funcional pero es eminentemente declarativo ya que expresamos los datos que queremos recuperar sin tener que indicar una serie de pasos para recuperarlos.

Ejemplo:
En sql: dame los nombres de todas las personas

```
select nombre
from personas
```

En imperativo(pseudocódigo):escribo un algoritmo, una serie de pasos

```
abrir fichero de personas
Mientras no fin de fichero personas
        leer una persona
        nombrePersona=acceder al nombre de persona
        imprimir nombrePersona
cerrar el fichero
```

¡qué fácil SQL!Su inconveniente es que es 100% específico, sólo sirve para recuperar datos de una BD relacional. Los lenguajes de programación funcional son de propósito general y por tanto mucho más potentes y flexibles, pero también son más difíciles.

Programación multiparadigma: consiste en el uso de dos o más paradigmas dentro de un programa. Python y Java (entre otros) son multiparadigma, ya que aunque son principalmente lenguajes O.O. tienen también algunos recursos que permiten programación funcional. También estos dos siguen el paradigma de programación orientada a eventos ya que permiten la creación de programas que interaccionan gráficamente con el usuario y por supuesto incluyen los principios de programación estructurada

PRINCIPIOS DE PROGRAMACIÓN FUNCIONAL

Aunque no vamos a entender en profundidad lo que es programación funcional y sus principios hasta que escribamos programas según este paradigma a modo introductorio veamos sus principios

- **Uso de funciones:** El concepto de "divide y vencerás" utilizado por Julio Cesar y Napoleón también se usa en programación para resolver problemas. En programación funcional todo lo podemos ver dividido en funciones y cada función resuelve un subproblema.
- **Se trabaja con funciones de primera clase y de orden superior:** Esto lo analizaremos mejor cuando veamos programación funcional en python. De forma resumida las funciones del lenguaje de programación deben de tener una serie de características que permitan hacer los mismos razonamientos que con las funciones matemáticas.
- **Funciones puras:** Totalmente predictivo, los mismos datos de entrada producirán los mismos datos de salida.
- **Recursividad:** Las funciones se pueden llamar a sí mismas
- **Inmutabilidad:** No hay variables, sólo constantes. Ya vimos que en POO existen gran cantidad de objetos "inmutables" y analizamos que la inmutabilidad proporciona seguridad e incluso eficiencia.
- **Evaluación perezosa:** En la programación funcional podemos trabajar con expresiones que no han sido evaluadas, o dicho de otra manera, podemos disponer de variables con operaciones cuyo resultado aún no se conoce. A esto se le denomina evaluación no estricta.

Un ejemplo de programación funcional con java

Dada una lista de números decimales queremos descontar un 20% a todos los que sean mayores de 25 y sumarlos:

Con programación imperativa

```
import java.util.ArrayList;

class DescuentoLista {
   public static void main(String[] args) {
      // creamos una lista de números decimales
      ArrayList<Double> lista = new ArrayList<Double>();
      lista.add(10.5);
      lista.add(30.0);
      lista.add(15.25);
      lista.add(40.75);

      // inicializamos la variable para almacenar la suma
      double suma = 0.0;
```

```
        // iteramos sobre la lista
        for (Double num : lista) {
            // verificamos si el número es mayor a 25
            if (num > 25) {
                // si el número es mayor a 25, aplicamos un 20% de descuento
                double descuento = num * 0.2;
                double nuevo_valor = num - descuento;
                suma += nuevo_valor;
            }
        }

        // imprimimos la suma de los valores descontados
        System.out.println("La suma de los valores descontados es: " + suma);
    }
}
```

Con programación funcional

Observa como desaparece el bucle y el if y todo se resuelve llamando a funciones

```
import java.util.ArrayList;

class DescuentoLista {
    public static void main(String[] args) {
        // creamos una lista de números decimales
        ArrayList<Double> lista = new ArrayList<Double>();
        lista.add(10.5);
        lista.add(30.0);
        lista.add(15.25);
        lista.add(40.75);

        // aplicamos el descuento y sumamos los valores descontados con programación funcional
        double suma = lista.stream()
                .filter(num -> num > 25)  // filtramos los números mayores a 25
                .map(num -> num * 0.8)    // aplicamos el descuento del 20%
                .mapToDouble(Double::doubleValue)  // convertimos los Double a double primitivos
                .sum();  // sumamos los valores descontados

        // imprimimos la suma de los valores descontados
        System.out.println("La suma de los valores descontados es: " + suma);
    }
}
```