Programación funcional:

Ejercicio 46:

Haz un ejemplo usando la expresión lambda Consumer y BiConsumer

Ejercicio 47:

Crea una clase PersonaJava8 donde tenga un nombre y apellidos. Desde el main crea un método que le cambie el nombre a una persona.

Crea un método que te pinte por pantalla un String introducido por parámetro. Llama a este método para mostrar el contenido de un arrayList

Haz un método que devuelva un numero al azar.

Crea una PersonaJava8 con la expresión Supplier y cambiale el nombre.

Ejercicio 48:

Crea una expresión Function que devuelva un valor aleatorio entre 1 y un valor introducido por parámetro.

Crea una expresión Function que reciba una cadena y la devuelva convertida en mayúsculas.

Crea una expresión BiFunction que reciba 2 números, que compare cual es mayor y devuelva un número al azar entre ambos.

Ejercicio 49:

Crea un Predicate que reciba un numero y te diga si es divisible entre 2.

Crea un BiPredicate que te diga si dos cadenas introducidas son iguales ignorando mayúsculas/minúsculas.

Ejercicio 50:

Crea una interfaz funcional que contenga un método “operacion” al que se le pasen dos números (o doubles).

Crea una clase calculadora que tenga un método al que se le pasen dos enteros (o doubles) y un objeto tipo Arithmetic y realice la operación entre ambos valores. Crea otro método que haga lo mismo pero en vez de pasarle un objeto tipo Arithmetic se le pase una BiFuncional.

En el main, crea dos operaciones diferentes (suma y resta, por ejemplo) de tipo Arithmetic y creando un objeto tipo calculadora mira las diferentes posibilidades que hay de llamar a sus métodos.

Ejercicio 51:

Crea un Stream de nombres de personas y muestra sus datos por consola (en 2 líneas de código)

Ejercicio 52:

Crea un Stream de nombre de personas en mayúsculas, conviértelo en nombres con solo la primera en mayúscula (el resto en minúsculas), muéstralo por pantalla y luego mételo dentro de una lista.

Ejercicio 53:

Crea un Stream de nombre de personas en minúscula, conviértelo en nombres con solo la primera en minúscula , muéstralo por pantalla , luego haz a la inversa (primera mayúscula y las siguientes minúsculas), muéstralo por pantalla y luego mételo dentro de una lista.

Ejercicio 54:

Crea un Stream de nombre de personas en mayúsculas, conviértelo en nombres con solo la primera en mayúscula (el resto en minúsculas), y muestra por pantalla, sólo los nombres que empiecen por J.

Ejercicio 55:

Crea un Stream de nombre de personas en mayúsculas, conviértelo en nombres con solo la primera en mayúscula (el resto en minúsculas), y usa las siguientes operaciones terminales (puedes inventarte tu la condición… que empiece por L, que acabe en "o",…):

- anyMatch
- noneMatch
- allMatch

Ejercicio 56:

- Crea un Stream de nombre de personas en minúscula, usa la operación distinct() y muéstralo por pantalla el nombre con la primera en mayúsculas
- Añádele la operación reduce()

Ejercicio 57:

- Crea un Stream de int y añádele una operación que sume sus valores
- Crea un IntStream con números entre el 1 y el 50
- A partir de IntStream anterior, crea un IntStream con número aleatorios entre el valor y el valor + 50.
- Con el IntStream resultado, muestra por pantalla algunas de sus estadísticas (máximo valor, mínimo, sumatorio, contar, media,….)

Ejercicio 58:

- Crea un Stream de nombres y obtén un IntStream con la longitud de cada uno de ellos
- Crea un IntSummaryStatistics para obtener las estadística (máximo valor, mínimo, sumatorio, contar, media,….)

Ejercicio 59:

- Con un Stream de nombres y usando la operación flatMap devuelve el nombre en mayúsculas si el nombre empieza por A y devuelve el nombre en minúsculas si empieza por T (por ejemplo, lo puedes adaptar a tu ejemplo)

Ejercicio 60:

- Usa la función generate() para mostrar números aleatorios por pantalla cada X minisegundos (se duerme el hilo durante ese tiempo), con un límite de 25 números

Ejercicio 61:

- Usa la función generate() para mostrar números aleatorios por pantalla cada X minisegundos (muestra este tiempo también por pantalla), con un límite de 25 números.
- Haz lo mismo que el punto anterior pero usando la función parallel().