Explained functional interface in Java 8 with example using Runnable.

Created a custom functional interface with one abstract method and showed example implementation.

Explained @FunctionalInterface annotation, its purpose, and that it’s not mandatory but helps enforce single abstract method.

Used a functional interface with lambda to perform flexible operations like add or multiply.

Used Predicate<String> to filter strings starting with "J" in a list.

Explained difference between Function<T, R> (one input) and BiFunction<T, U, R> (two inputs).

Used Predicate and removeIf to remove even numbers from a list.

Used Consumer<List<String>> to print all list items in uppercase.

Explained a functional interface can have any number of default methods but exactly one abstract method.

Used Comparator to sort a list of strings based on their length.

Created two interfaces with same default method run(), implemented both in a class, and resolved conflict by overriding and using InterfaceName.super.run().

Demonstrated calling an interface’s default method from an implementing class when it is overridden using InterfaceName.super.methodName().

Provided an example explicitly calling a default method using InterfaceName.super.methodName().

Created interface with static method square(int) and called it from main method.

Created interface StringUtils with static methods isNullOrEmpty and capitalize, and demonstrated calling them.

Defined functional interface MathOperation and implemented add, subtract, multiply operations using lambdas.

Sorted a list of strings by their length using lambda in sort().

Used String::toUpperCase as a method reference assigned to Function<String, String>.

Filtered a list of strings to find elements starting with "A" using stream and filter.

Used Predicate.and() and Predicate.or() to filter numbers divisible by both 2 and 3 (and either).

Used Supplier<Double> to generate and print 5 random numbers with Math.random().

Used Consumer<String> to print each element in a list.

Used Consumer<Integer> to double each integer in a list and print the result.