What is Map-Reduce?

Map-Reduce is a functional programming model. It serves our 2 purposes.

Map 🡪 Transforming data

Reduce 🡪 Aggregating data

Example 🡪 Stream: [2,4,9,4,8,1,3] Sum of numbers presents in Streams?

Map () 🡪 Transform stream<Object> to stream of int

Reduce () 🡪 Combine Stream of int and produce the sum result.

Integer sumResult = Stream.of(2,3,4,5,6,7,8).reduce(0,(a,b) -> a+b);

Here 0 = it’s identity with initial value.

Accumulator = (a,b) -> a+b function.

Map() & flatMap()

Java 8 stream API provides map() and flatMap() method. Both these methods are intermediate methods and returns another stream as part of the output.

Map() method used for transformation

flatMap() method used for transformation and flattering

flatMap() 🡪 map() + flattering

map() method 🡪 Data transformation

map() takes Stream<T> as a input and return Stream<R>

Stream<R> map(Stream<T> input){}

It’s a mapper function produce single value for each input. Hence it is also called One-to-One Mapping.

flatMap() method 🡪 map() + flattering

flatMap() takes Stream<Stream<T>> as a input and return Stream<R>

Stream<R> flatMap(Stream<Stream<T>> input){}

It’s mapper function produces multiple value for each input value. Hence it is also called One-to-Many mapping.

Data Transformation :- Stream.of(“a”,”b”,”c”,”d”) 🡪 [A,B,C,D] (Transform date from lowercase to uppercase)

Data flattering :- [ [1,2], [3,4], [5,6], [7,8] ] 🡪 [1,2,3,4,5,6] (convert Stream of Stream into a single Stream)

What is Java Parallel Streams?

Java Parallel Streams is a feature of Java 8, It meant for utilizing multiple cores of the processor.

Normally any java code has one stream of processing, where it is executed sequentially. Whereas by using parallel streams, we can divide the code into multiple streams that are executed in parallel on separate cores and the final result is the combination of the individual outcomes.

The order of execution is not our control.

What is Consumer?

Consumer<T> is an in-built functional interface introduced in Java 8. Consumer can be used in all contexts where an object needs to be consumed i.e., taken as input and some operation needs to be performed on the object without returning any result.

Name of Abstract method 🡪 void accept(T t);

What is Predicate?

This functional interface used for conditional check, where you think we can use these true/false returning functions in day-to-day programming we should choose predicate.

Name of Abstract method 🡪 Boolean test(T t);

What is Supplier?

Supplier can be used in all contexts where there is no input but an output is expected.

Name of the Abstract method 🡪 T get();