Overview

Let's recap what our stream chain does and look at the results of each step:

Input	Method/Operation	Output
The ArrayList someBingoNumbers	stream()	A Stream that contains all the items in the someBingoNumbers list, in the same order
Stream containing all bingo numbers	map(String::toUpperCase)	A Stream that contains all the bingo numbers uppercased
Uppercased stream	filter(s→s.startsWith("G")	A Stream containing all items beginning with "G" ("G53", "G49", "G60", "G50", "G64")
"G" items stream	sorted()	A stream containing the sorted items ("G49", "G50", "G53", "G60", "G64")
Sorted "G" items stream	forEach(System.out::println)	Each "G" item is printed to the console. Void result. The chain ends.

When a chain is evaluated, a stream pipeline is created. The stream pipeline consists of a source, zero or more intermediate operations, and a terminal operation. In our example, we used a collection as the source, but we could also be an array or an I/O channel, and we can build streams from scratch.

of each step so firstly when the chain is evaluated a stream pipeline is

Overview

Input	Method/Operation	Output
The ArrayList someBingoNumbers	stream()	A Stream that contains all the items in the someBingoNumbers list, in the same order
Stream containing all bingo numbers	map(String::toUpperCase)	A Stream that contains all the bingo numbers uppercased
Uppercased stream	filter(s→s.startsWith("G")	A Stream containing all items beginning with "G" ("G53", "G49", "G60", "G50", "G64")
"G" items stream	sorted()	A stream containing the sorted items ("G49", "G50", "G53", "G60", "G64")
Sorted "G" items stream	forEach(System.out::println)	Each "G" item is printed to the console. Void result. The chain ends.

The items in the source enter the pipeline, and the chain result emerges at the other end of the pipe. As we've seen, elements may be removed from the stream as a result of an operation, so the set of elements that comes out at the other end of the pipe doesn't have to match the number that entered the pipe.

the items in the source enter the pipeline and the chain result emerges as

Overview

Input	Method/Operation	Output
The ArrayList someBingoNumbers	stream()	A Stream that contains all the items in the someBingoNumbers list, in the same order
Stream containing all bingo numbers	map(String::toUpperCase)	A Stream that contains all the bingo numbers uppercased
Uppercased stream	filter(s→s.startsWith("G")	A Stream containing all items beginning with "G" ("G53", "G49", "G60", "G50", "G64")
"G" items stream	sorted()	A stream containing the sorted items ("G49", "G50", "G53", "G60", "G64")
Sorted "G" items stream	forEach(System.out::println)	Each "G" item is printed to the console. Void result. The chain ends.

We noticed that when we weren't using streams, our "g64" number was printed with a lower-cased 'g'. But when we used streams, it was printed with an upper-cased 'G'. In the non-stream case, we didn't use the result of the toUpperCase() call. When an item passed the test, we assigned the original string with the lower-cased 'g' to the new list.

also notice that when we weren't using streams our g64 number is printed with a

Overview

Input	Method/Operation	Output
The ArrayList someBingoNumbers	stream()	A Stream that contains all the items in the someBingoNumbers list, in the same order
Stream containing all bingo numbers	map(String::toUpperCase)	A Stream that contains all the bingo numbers uppercased
Uppercased stream	filter(s→s.startsWith("G")	A Stream containing all items beginning with "G" ("G53", "G49", "G60", "G50", "G64")
"G" items stream	sorted()	A stream containing the sorted items ("G49", "G50", "G53", "G60", "G64")
Sorted "G" items stream	forEach(System.out::println)	Each "G" item is printed to the console. Void result. The chain ends.

In the stream case, the map() method maps each source string to the Function result, therefore the upper-cased string is added to the resulting stream and passed to the next step in the chain. That's why the non-stream case prints a lower-cased 'g', and the stream case prints an upper-cased 'G'.

but in the stream case the map method maps each source string to the function