

Nachdenkzettel: Streams

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/0bcfeb6a-745d-4807-90be-d6e063809774/streamsn8s2.pdf>

Aufgabe 1

```
final List<String> names = Arrays.asList("John", "Karl", "Steve", "Ashley", "Kate");
names.stream().filter(s -> s.startsWith("K")).forEach(System.out::println);
```

Aufgabe 2

- map()
- filter()
- println()
- stream()

The access method is named “stream.of”

Aufgabe 3

The values or functions passed to `forEach()` or `peek()` shouldn't be stateless. Because the functions can cause side effects because of their state, `forEach()` and `peek()` operates through side effects.

Aufgabe 4

The `sort()` function always accesses the whole collection. The whole collection is being “collected”, sorted, and passed further on in the stream. Other functions distribute the

data. When using multiple streams, `sort()` shouldn't be used, because it slows down the application.

Aufgabe 5

a) parallel HashSet map causes errors because it's a parallel stream with "side effects". The parallel stream operates on the list which can lead to unwanted results

b) wrong use of strings. A result can be added, without other threads knowing about it.

Aufgabe 6

- Won't work (`.pos()` doesn't exist). But will work when replaced with: `.maptoInt(x → Integer.parseInt(x.substring(0,1)))`
- Good for the performance, performance doesn't get wasted

Aufgabe 7

used to accumulate because of parallel stream → needed for the addition of values

Aufgabe 8

Performance boost, but you'll lose the order of the collection

Aufgabe 9

a) 3rd line: Stream already used (can only be used once) → causes `IllegalStateException`

b) produces infinite numbers → needs a limit

c) Causes infinite numbers (iteration only generates 0 and 1)

d) Causes Modification side effects because of the `peek()`