P2W6 Concurrency - Opdracht ParallelStreams (3 delen)

Deel 1:

Maak een **main** methode met daarin de volgende code (creatie van een lijst **getallen** met de getallen van 1 tot en met 10 via een **Stream**).

```
List<Integer> getallen =
Stream.iterate(1, n -> n + 1)
.limit(10)
.collect(Collectors.toList());
```

Test nu achtereenvolgens het volgende uit

1. Maak op basis van getallen een **parallelStream** en gebruik daarna de **forEach** methode om de getallen af te drukken. Wordt de originele volgorde bewaard of niet?

2. Maak een nieuwe **List<Integer> collected** op basis van getallen. Maak daarvoor gebruik van een **parallelStream** methode gevolgd door een **collect**. Druk de inhoud van collected vervolgens af. Wat geeft dit voor de volgorde?

3. Maak een **String** met de naam **string**. Zet de getallen van de List getallen via een **parallelStream**, een **map** en een **collect** om naar één string waarbij de getallen van elkaar gescheiden worden door een komma gevolgd door een spatie. Wat geeft dit voor de volgorde?

Deel 2:

Maak een **main** methode met daarin de volgende code (creatie van een lijst getallen van 1 tot en met 10000 via een **IntStream**).

```
List<Integer> getallen =
IntStream.rangeClosed(1, 10000)
.boxed()
.collect(Collectors.toList());
```

1. Bepaal de som van alle getallen in de lijst **getallen** en stop het resultaat in de **Integer** variabele **totaal**. Werk met **parallelStream** en **reduce**. Druk daarna de waarde van **totaal** af (zou 50005000 moeten zijn).

2. Bepaal het aantal getallen in de lijst **getallen** dat deelbaar is door 3 en een veelvoud van 5 en stop het resultaat ervan in de **long** variabele

aantal. Maak gebruik van een **parallelStream** en **count**. Druk de waarde van **aantal** af (zou 666 moeten zijn).