Verslag – threaded – merge sort

In deze verslag leg ik uit hoe ik de threaded merge sort heb gerealiseert.

Ceyhun Çakir | 1784480

2022

Inhoud

[3B.1 2](#_Toc97713569)

[Merge sort met 1 thread 2](#_Toc97713570)

[Merge sort met 2 thread’s 3](#_Toc97713571)

[Merge sort met 4 thread’s 4](#_Toc97713572)

[Merge sort met 8 thread’s 5](#_Toc97713573)

[3B.2 6](#_Toc97713574)

[Merge sort complexiteit met 1 thread’s 6](#_Toc97713575)

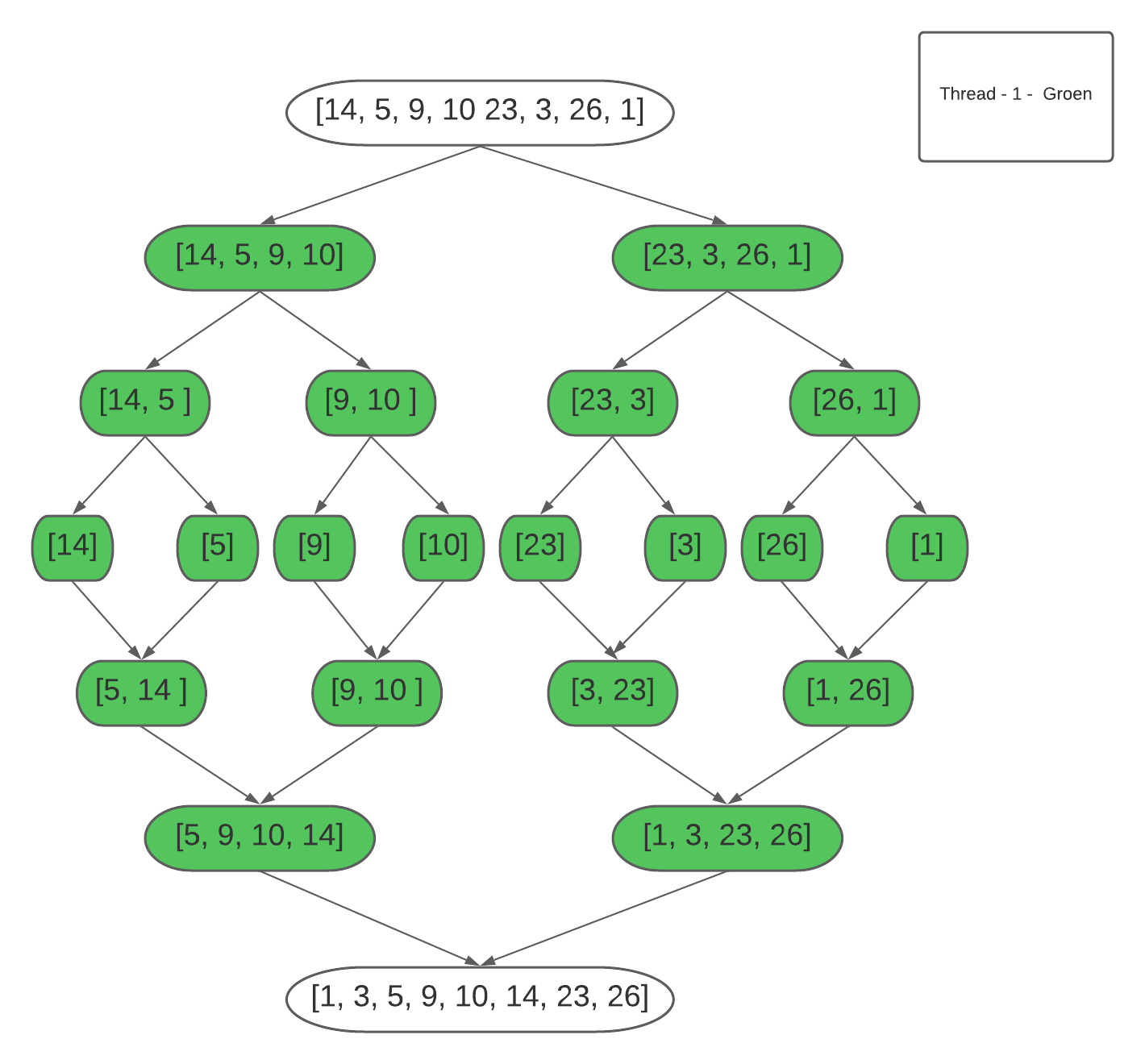
[Merge sort complexiteit met 2 thread’s 6](#_Toc97713576)

[Merge sort complexiteit met 4 thread’s 6](#_Toc97713577)

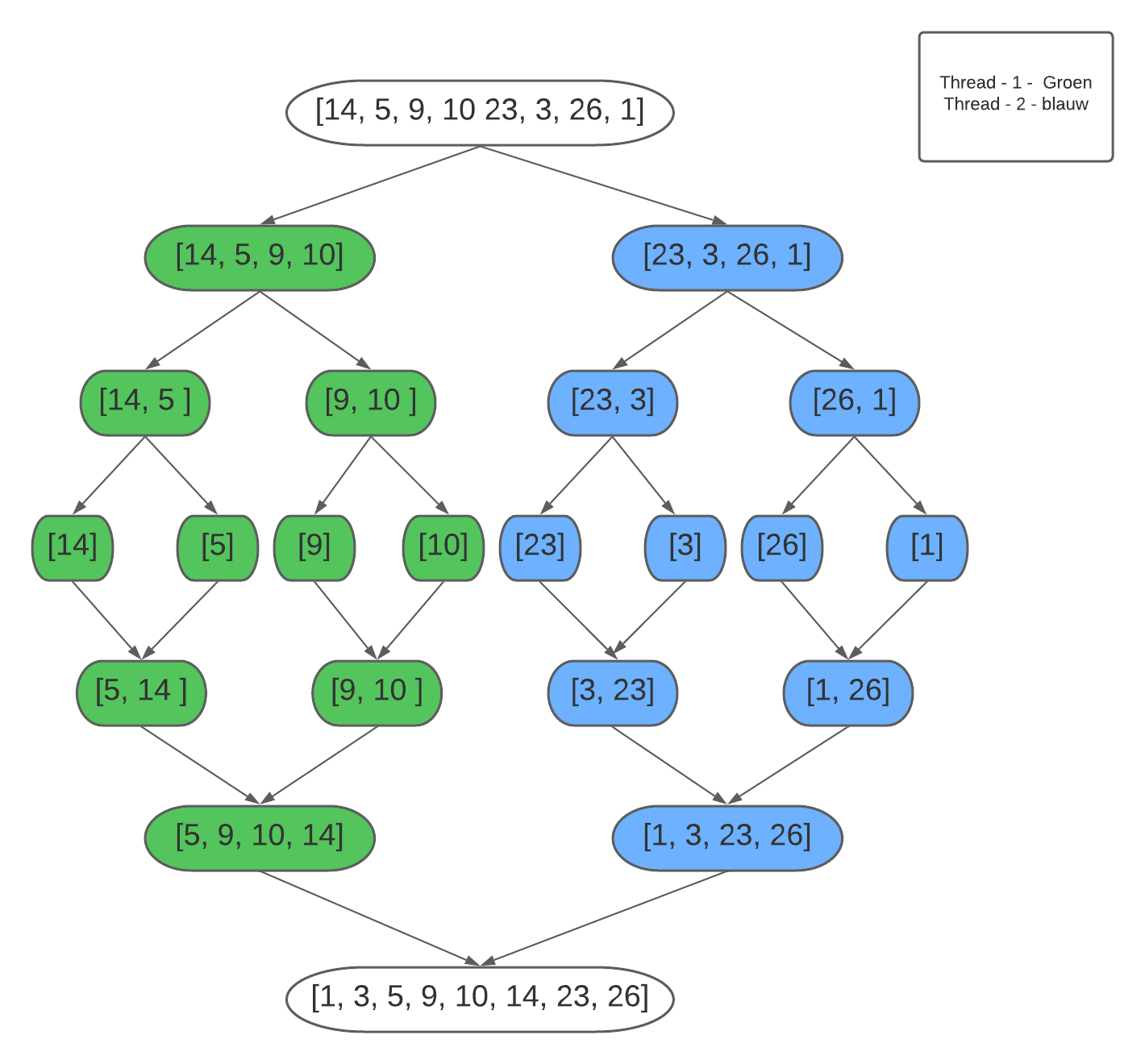
[Merge sort complexiteit met 8 thread’s 6](#_Toc97713578)

# 3B.1

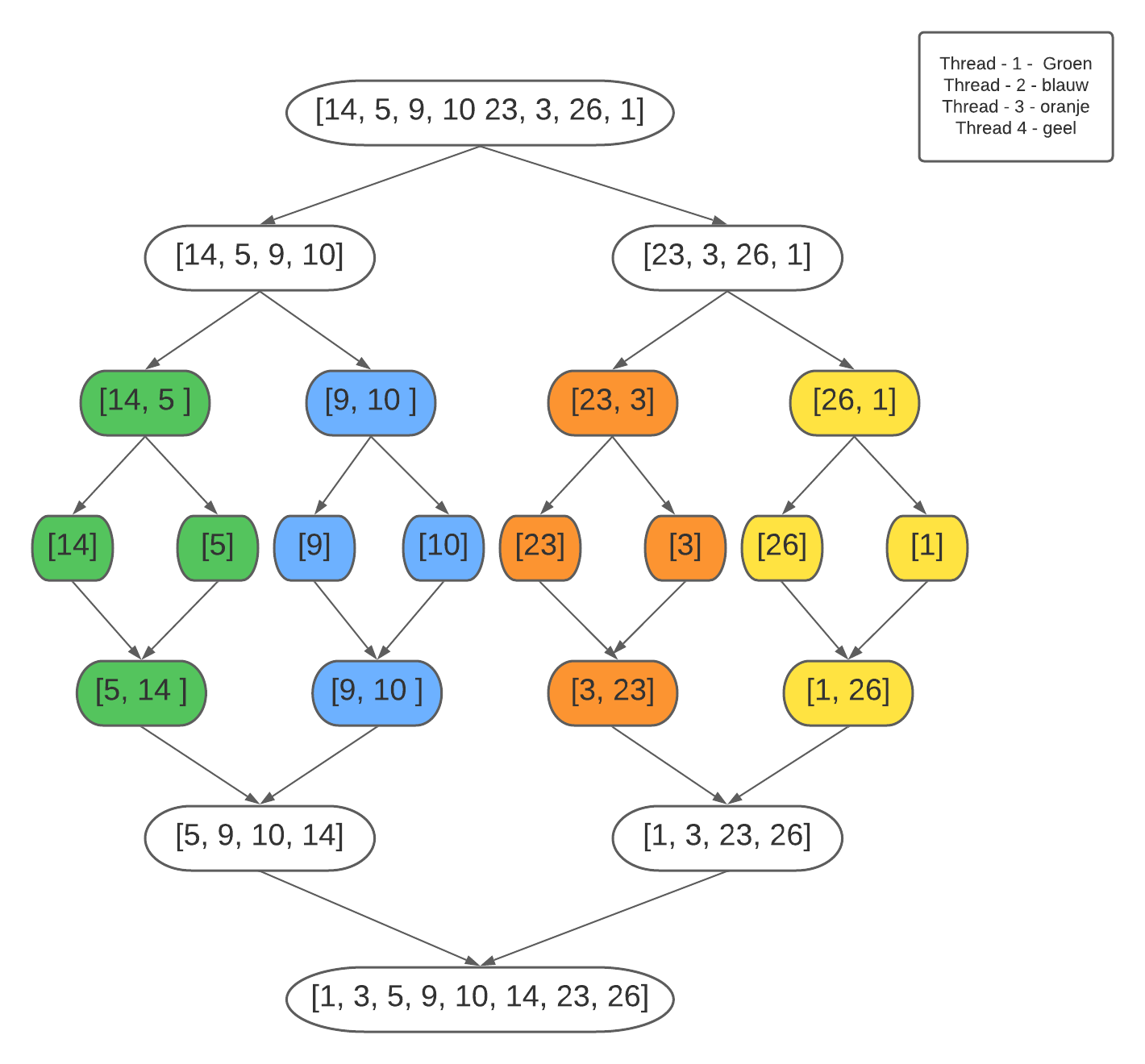
## Merge sort met 1 thread



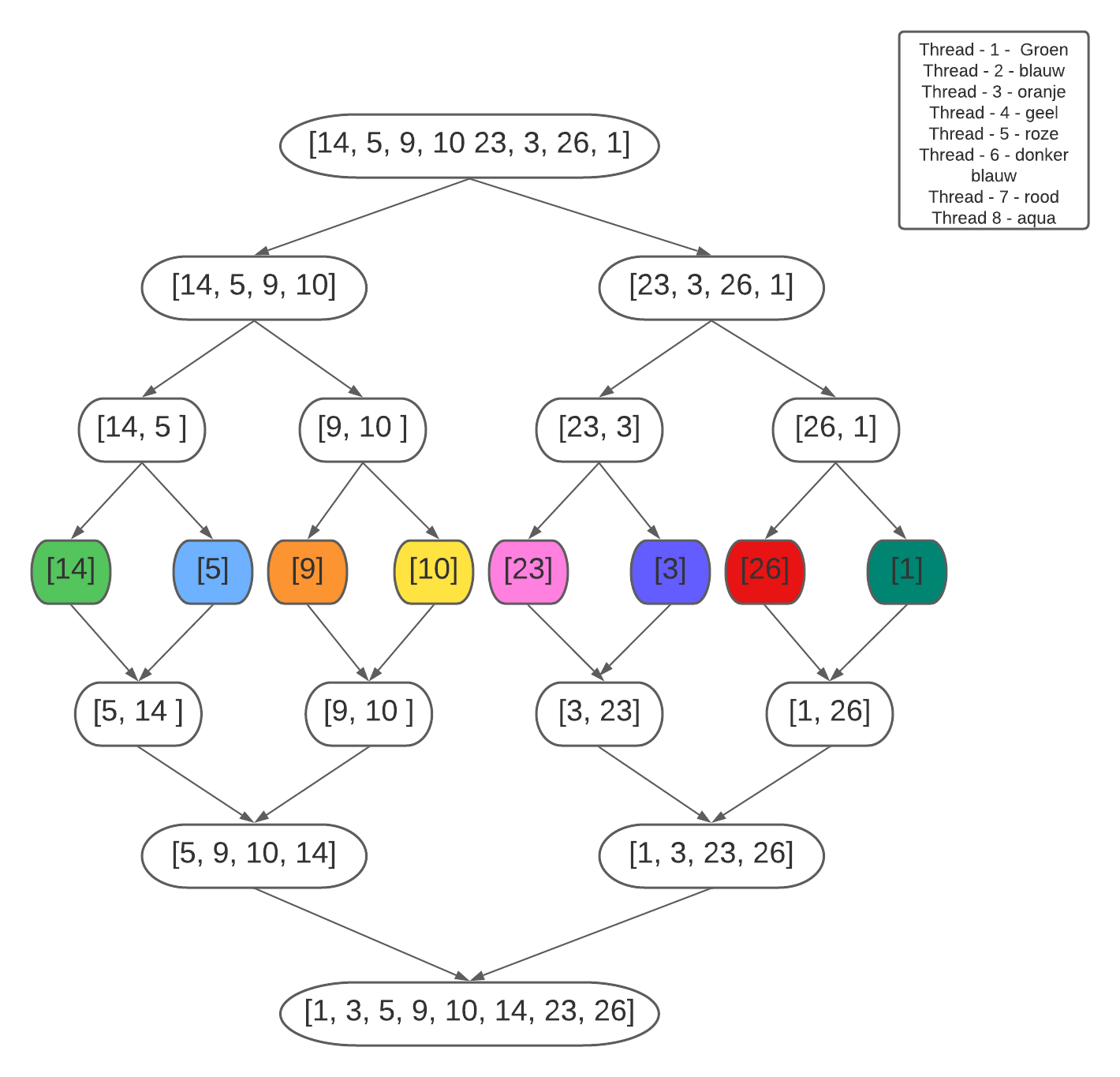
## Merge sort met 2 thread’s



## Merge sort met 4 thread’s



## Merge sort met 8 thread’s



# 3B.2

## Merge sort complexiteit (default)

We kunnen aan de hand van analyse van de code en ook research te doen op het internet vastleggen dat de merge sort een tijds complexiteit heeft van een O(n log n).

## Merge sort complexiteit met 1 thread’s

Wanneer de we merge sort algoritme realiseren met gebruik van threads verrandert de tijdscomplexiteit ook. De tijdscomplexiteit kunnen we dan vastleggen als O((n log n) / 1). Dit doen we aangezien we het moeten delen door de gebruikte threads voor de sortering.

## Merge sort complexiteit met 2 thread’s

Wanneer de we merge sort algoritme realiseren met gebruik van 2 threads verrandert de tijdscomplexiteit. De tijdscomplexiteit kunnen we dan vastleggen als O((n log n) / 2). Dit doen we aangezien we het moeten delen door de gebruikte threads voor de sortering.

## Merge sort complexiteit met 4 thread’s

Ook wanneer de 4 threads gebruiken verrandert de tijdscomplexiteit naar O((n log n) / 4)

## Merge sort complexiteit met 8 thread’s

Ook wanneer de 4 threads gebruiken verrandert de tijdscomplexiteit naar O((n log n) / 8)