Maskin brukt for testing: MacBook Pro
Hukommelse: 8 GB 1867 MHz DDR3
Prosessor: 2,7 GHz Intel Core i5

	Sekvensiell-Erastothenes	**Parallell-Erastothenes**	**Speedup**
N = 2000000	24.583136ms.	45.951679ms.	0,534
N = 20000000	107.584113ms.	152.444079ms.	0,706
N = 200000000	1423.108404ms.	1818.38998ms.	0,782
N = 2000000000	17965.546457ms.	???	???

	Sekvensiell-Faktorisering	**Parallell-Faktorisering**	**Speedup**
N = 2000000	32.036119ms.	486.379274ms.	0,065
N = 20000000	119.117615ms.	3607.520931ms.	0,003
N = 200000000	1441.123221ms.	33932.427484ms.	0,033
N = 2000000000	???	???	???

Ved 10-dobling av problemet, ser vi at erastotenes-sekvensielt gir en en mer eller mindre 10-dobling av tiden, mens ved parallell, øker den mer.
Ved faktoriseringen øker tiden mindre enn 10 ganger ved sekvensiell, mens den øker mye mer ved parallell

Jeg løste faktoriseringen ved at jeg forsøkte å dele tallet på 2. Fungerte dette la jeg 2 i et array, som ville inneholde alle faktoriserings-verdiene til tallet, delte tallet på faktoriserings-verdien og forsøkte med 2 igjen. Fungerte ikke 2, ville jeg øke til 3, 4, osv. Helt til jeg kom til tallets egne verdi. Da ville jeg vite at tallet bare var delelig med seg selv og 1, og faktoriseringen var dermed ferdig.

Jeg parallelliserte dette ved å la alle trådene få jobbe på samme primtall for så å prøve å finne faktoriserings-verdier hver for seg.