Øving 1 Rekursjon – Rapport

Alle tester er gjort med x = 1,001 og forskjellig n, som vist i tabellen under

	n = 10	n = 100	n = 500	n = 1000	n = 2000	n = 5000
Program 1	1,40 ms	1,45 ms	1,57 ms	1,62 ms	1,7 ms	1,85 ms
Program 2	0,025 ms	0,027 ms	0,03 ms	0,03 ms	0,032 ms	0,035 ms
Math.pow()	0,05 ms	0,05 ms	0,05 ms	0,055 ms	0,055 ms	0,055 ms

Jeg hadde en teori om at programmene ville ha ulik tid, ut ifra kompleksiteten til hver av programmene. Tidene mellom programmene var forskjellige, og det var ingen store likheter. Program 2 og Math.pow() hadde nesten lik tid. Jeg ble overrasket når jeg så at endringen i tid, når n ble høyere, ikke var stor. Det er nesten ingen endring i tid. Jeg har noen tanker om hvorfor det er liten forskjell, selv om n øker.

Min første tanke var at programmet var feil, og at jeg fikk feil svar. Dette skjedde ikke. Jeg har dobbeltsjekket svarene, og alle svar jeg får er korrekte. Jeg har også fulgt alle regler for rekursjon i Java.

Min andre tanke er at det kan ha noe med hvor god ytelse en PC har for at det skal være noe forskjell i tid. Jeg gjennomførte testene på min stasjonære PC, hvor jeg nylig har oppgradert prosessor, hovedkort og RAM. Før denne oppgradering, brukte programmene lengre tid. Da kunne man også se større tidsforskjeller i program 1 og 2. Etter oppgraderingen virker det som om PC-en ikke blir påvirket av høyere n-verdi. Derfor mener jeg at ytelse kan påvirke hvor raskt programmet kjører.

Teoretisk sett skal program 2 være mye raskere enn program 1, da program 2 looper log(n) ganger, kontra program 1, som looper n ganger. Logisk sett skulle program 1 og program 2 ha en tidsendring på hhv. n og log(n).

Alt i alt kan vi se at program 2 bruker kortere tid enn program 1 og Math.pow(). Likevel ligner tidene til program 2 og Math.pow() veldig mye, mens program 1 skiller seg ut.