|  |
| --- |
| Högskolan Dalarna |
| Laboration 2 |
| GMI24H |

|  |
| --- |
| Mikael Olsson  2022-05-03 |

# Muntlig redovisning

# Syfte

Syftet med denna laboration är att utveckla några algoritmer och implementera en stack.

# Diskussion

Uppgift1

Vid mätning av exekveringstiden uppstod det problem med användandet av rekursion i quicksort algoritmen, när den skulle köras med olika mätdata så uppstod det problem med ”Stackoverflow exception”. Därför kommer indata se olika ut beroende på vilken graf som visas. Trots det går det att dra några slutsatser utifrån mätningarna.

Först visas graferna där alla element är slumpade

Stora ordo för quicksort är vilket kan ses i grafen ovan.

Stora ordo för insertionsort är vilket kan ses i grafen ovan.

Stora ordo för selectionsort är vilket kan ses i grafen ovan.

En sak som kan konstateras om man kollar på graferna är att exekveringstiden varierar mycket. Insertion och selection- sort liknar varandra när det kommer till exekveringstdera. När går det att se att insertion och selection- sort har en exekveringstid runt 6000ms eller 6 sekunder. Quicksort har bara en exekveringstid runt 50ms. Det intressanta kan dock ses nedanför då indata är en sorterad array.

När en sorterad array inmatades då vart exekveringstiden konstant. Detta är ett underligt svar men det gick att återskapa ett flertal gånger. De andra algoritmerna liknade deras osorterade grafer.

Uppgift 2

Vid optimering, där Insertion och quick- sort kombineras, syntes en stor förbättring. Däremot verkar valet av pivot inte vara intressant när det kommer till prestanda.

Ovan går det att se hur quicksort presterar, liknar tidigare graf.

Det intressanta är att ökningen verkar skarpare och att exekveringstiden dubbleras mellan de två olika implementationerna.

De val som gjordes var att implementera insertionsort när det är under 10 element men att valet av pivot-värde verkade inte påverka resultatet därav valdes det att behålla det originella sättet där det sista elementet blir pivot-värdet.

Uppgift 3

En stack implementerades med all funktionalitet, stacken liknar uppgiften i laboration 2 med linked lists. En stack kan t.ex. användas i en ”undo” lista i ett program. När användaren ändrar något läggs detta på stacken och det går då också att gå tillbaks.

# Kommentarer

Intressant labb.