Sorteringsexperiment

Quicksort (Varför metoden “IsSorted” hittar flaws)

I teori tror vi att anledningen varför vår Quicksort får en del flaws(10+) är pågrund av att vår pivot i quicksort inte hinner sortera en så stor mängd information på så kort tid. Ett vanligt problem med quicksort är att pivot missar vissa tal och sätter in dem i en tom lista då den inte hinner förstå var de hör till. Talen i denna listan kommer då ut som flaws då quicksort inte hunnit sortera dem korrekt. Vi studerade vanliga problem med quicksort och det verkade stämma in på vår kod. En stor anledningen varför de flesta föredrar mergesort även om det tar längre tid är för mer precision i sortering angående stora mängder information. I slutändan verkar det som att quicksort är en mycket snabbare typ av sorting för en mindre mängd information, 0.01 sekunder, medans mergesort definitivt tar mer tid, upp till 1 minut, men har en mer säker utskrivning sorterad.

Mergesort (Varför uträkning tar så mycket längre tid än quicksort)

Som nämnt ovan tar mergesort ungefär 1 minut att gå igenom alla tal samt sortera dem. Vi tror att anledningen till detta ligger i att vi använt oss av en for-loop istället för en while-loop då for-loop har gett oss lite långsammare resultat. Vi testade detta och det visade sig att det tar nästan lika lång tid, det verkade t.om. som att en while-loop var långsammare. Vi spekulerar om att kanske tömma onödig data som redan är sorterad för att spara tid samt minne. Hur vi skulle lägga in det i programmeringen har vi dock svårt att lista ut. Vi försöker hitta områden i mergesort metoden som kanske är onödiga eller kan optimeras genom att rensa innehåll som är onödigt att ha kvar.