Veckoövning V6: Sortering

Under veckans föreläsning har vi tittat närmare på begreppet rekursion: metoder som anropar sig själva (eller andra metoder) om och om igen tills ett visst mål uppnåtts. I den här övningen får du lära dig att använda rekursion till någonting (mer eller mindre) praktiskt: implementera *merge sort*, en rekursiv algoritm för att skapa en sorterad kopia av en array.

Merge sort är relativt enkel att beskriva. För en array `arr`:

1. Om `arr` har 0 eller 1 element så är `arr` sorterad per definition, så returnera `arr`.
2. Om `arr` har fler än ett element:
    1. Sortera den första halvan av `arr` rekursivt, och kalla resultatet för `a`.
    2. Sortera den andra halvan av `arr` rekursivt, och kalla resultatet för `b`.
    3. Returnera `merge(a, b)`, där `merge` är en metod som slår ihop två *sorterade* arrayer så att även resultatet är sorterat. Dvs, `merge({1,3,6}, {2,4,5,9}) == {1,2,3,4,5,6,9}`.

1. Förberedelser

Gå igenom veckans föreläsning och läsanvisningar. Försök implementera kodexemplen själv om du är osäker på hur rekursion fungerar.

2. Skriv `merge`-metoden

Implementera metoden `public static int[] merge(int[] a, int[] b)`. Denna metod skrivs enklast iterativt (dvs. med loopar). Tänk på att `a` och `b` kan ha olika längd.

3. Skriv resten av sorteringsalgoritmen

Använd din implementation av `merge` för att implementera merge sort så som den beskrivs ovan. Testa din algoritm med några olika arrayer (av olika längd!) för att kontrollera att den verkligen gör det den ska.

4. En krångligare sorteringsalgoritm

Merge sort är relativt enkel att förstå, och kan lätt anpassas efter datamängder som är för stora för att få plats i datorns minne på samma gång. Dock är det inte den snabbaste algoritmen för mindre datamängder. Den äran går till quicksort, som dock är lite svårare att förstå sig på.

Skumma igenom Wikipediasidan för Quicksort, och försök skriva en egen metod `public static void quickSort(int[] arr, int start, int end)` som implementerar algoritmen.

5. Test med större datamängder

Det kan ofta vara användbart att testa sin kod med slumpmässig data: kanske kan vi slumpa fram ett testfall som belyser en bugg vi inte hade hittat om vi själva valt våra testfall för hand?

1. Skriv en metod som genererar med slumpmässigt innehåll. Försök att även göra storleken på arrayen slumpmässig, inom något lämpligt intervall.
2. Skriv en metod som kontrollerar att en array faktiskt är sorterad. Det vill säga, givet en array `arr`, så ska varje element `arr[i]` vara mindre än eller lika med `arr[i+1]`.
3. Testa din merge sort och din quicksort med dina slumpade arrayer och metoden som kontrollerar att om en array är sorterad. Fungerar sorteringen fortfarande korrekt?