

## DM507 – Opgaver uge 7

### Eksaminatorier

1. Eksamen juni 2011, opgave 2 (eksamensopgaver kan findes på kursets hjemmeside).
2. Eksamen juni 2008, opgave 2.
3. Eksamen januar 2007, opgave 3.
4. Cormen et al. øvelse 7.1-1 (side 173).
5. Cormen et al. øvelse 7.1-2 (side 174). Her er  $q$  værdien som returneres, dvs.  $(i+1)$  i linie 8 i pseudo-koden i Cormen et al. side 171. Hint: Start med at lave princippet for det mørkegrå område på figur 7.2 (side 173) om fra " $> x$ " til " $\geq x$ " (overvej hvorfor dette ikke vil ændre på korrektheden af selve Quicksort), og fyld så det lysegrå og det mørkegrå område så balanceret som muligt under udførslen af PARTITION.
6. Cormen et al. øvelse 7.2-2 (side 178). Antag her at det er PARTITION fra side 171, der bruges (ikke PARTITION varianten lavet i øvelse 7.1-2 ovenfor).
7. Cormen et al. øvelse 7.2-3 (side 178). Vis at det gælder, både når input er stigende sorteret, og når det er aftagende sorteret.
8. Cormen et al. problem 7-2, spørgsmål b (side 186). Svar derefter på de to ovenstående øvelser igen, under antagelse af at Quicksort bruger den nye PARTITION procedure (og laver to rekursive kald på de to dele af array  $A$  hvor PARTITION har placeret elementer forskellige fra  $A[q]$ ).
9. Implementer Mergesort i Java ud fra bogens pseudokode (side 31 og 34). Test at din kode fungerer ved at generere arrays med forskelligt indhold og sortere dem. Tilføj tidtagning af din kode (kun selve sorteringen, ikke den del af programmet som genererer array'ets indhold).

Kør derefter din kode med input, som er random `int`'s. Gør dette for mindst 5 forskellige værdier af  $n$  (antal elementer at sortere), vælg værdier som får programmet til at bruge fra ca. 100 til ca. 5000 millisekunder. Gentag hver enkelt kørsel tre gange og find gennemsnittet af antal millisekunder brugt ved de tre kørsler. Dividér de fremkomne tal med  $n \log_2 n$  og check derved hvor godt analysen passer med praksis – de resulterende tal burde ifølge analysen være konstante.

Se seddel med opgaver til uge 6 for informationer om bogens array-indeksering i forhold til Java's, samt om metoder i Java til tidstagnung og til generering af random `int`'s. For at beregne logaritmer i Java, brug metoden `java.lang.Math.log()`, som beregner  $\log_e(x)$ , dvs. logaritmen med grundtal  $e$ . For at beregne  $\log_2(x)$  kan man bruge at  $\log_2(x) = \log_e(x) / \log_e(2)$  (jvf. formel 3.15 i bogen side 56). Metoden skal have en `double` som input, så hvis dit  $n$  meget naturligt er en `int`, skal du bruge type casting (typekonvertering), dvs. `(double) n`. Se evt. dokumentationen for `java.lang.Math` klassen og denne vejledning om type casting.

## Studiegrupper

Forslag til fokus for arbejde i studiegrupper: Forbered dele af opgaverne til eksaminatorietimer, f.eks. på nedenstående måde.

- Lav eksamensopgaverne (opgave 1–4) individuelt på forhånd, og ret hinandens i gruppen.
- Forsøg at løse opgave 5–8 i fællesskab. Arbejd både med at få ideer på skitseplanet til de ønskede algoritmer og argumenter, og med at få dem formuleret præcist til sidst. I kan evt. dele disse opgaver op imellem delgrupper, som senere forsøger at formidle de fundne løsninger til hinanden så klart og præcist som muligt.
- I opgave 9, lav programmeringen og programkørsel (gerne i par) før gruppemøde. På mødet, sammenlign køretider.