

DM507 – Opgaver uge 9

Eksaminatorier I

1. Eksamen juni 2008, opgave 2 (eksamensopgaver kan findes på kursets hjemmeside).
2. Eksamen juni 2011, opgave 2. Opgaven bruger notationen $f(n) \in O(g(n))$, hvilket betyder det samme som $f(n) = O(g(n))$. Se også diskussion side 45 i bogen.
3. Eksamen januar 2007, opgave 3.
4. Implementer Mergesort i Java til ud fra bogens pseudokode (side 31 og 34). Lad input være et array af `int`'s. Som værdien ∞ (se pseudokode side 31), brug `Integer.MAX_VALUE`, og sørg for at denne værdi (som er $2^{31} - 1 = 2.147.483.647$) ikke optræder i input. Alternativt, brug til MERGE din pseudokode fra øvelse 2.3-2 (fra seddel med opgaver til uge 6), som ikke anvender ∞ .

Test at din kode fungerer ved at generere små arrays med forskelligt indhold (f.eks. stigende, faldende, random), sortere dem og checke resultatet. 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. Divider 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 tidtagning 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.

5. (*) Cormen et al. opgave 2-4 (side 41), spørgsmål **d**.

Eksaminatorier II

1. Cormen et al. øvelse 7.1-1 (side 173).
2. Cormen et al. øvelse 7.1-2 (side 174). Her er q værdien som `PARTITION` returnerer, 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`.
3. 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). Hint: i øvelse 7.1-2 (side 174), som er løst ovenfor, er der undersøgt i detaljer hvad `PARTITION` fra side 171 gør på det pågældende input. Argumenter for køretiden, når dette sker gentagne gange i Quicksort.
4. Cormen et al. øvelse 7.2-3 (side 178). Vis at det faktisk gælder både når input er stigende sorteret og når det er aftagende sorteret. Hint: undersøg i detaljer hvad `PARTITION` fra side 171 gør på de pågældende input. Argumenter så for køretiden, når dette sker gentagne gange i Quicksort.
5. Cormen et al. problem 7-2, spørgsmål b (side 186). Svar derefter igen på de to sidste øvelser [dvs. Cormen et al. øvelse 7.2-2 (side 178) og øvelse 7.2-3 (side 178)], 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 $A[q]$).

6. Eksamen juni 2008, opgave 1 b. Man må gerne referere til oplysninger i bogen, når man giver begrundelser. Quicksort antages at bruge PARTITION fra side 171.
7. Cormen et al. øvelse 6.1-5 (side 154).
8. Cormen et al. øvelse 6.1-6 (side 154).
9. Eksamen juni 2008, opgave 4 a. Hob er en fordanskning af ordet heap.
10. Cormen et al. øvelse 6.1-4 (side 154).
11. Eksamen januar 2008, opgave 1 b (sidehenvielsen skal være til side 164 i vores udgave (tredie) af Cormen et al.).

Studiegrupper

Forslag til fokus for arbejde i studiegrupper (hvis man er i en sådan): Forbered dele af opgaverne til eksaminatorietimer, f.eks. på nedenstående måde.

- Lav eksamensopgaverne individuelt på forhånd, og ret hinandens i gruppen.
- I opgave I.4, lav programmeringen og programkørsel (gerne i par) før gruppemøde. På mødet, sammenlign køretider.
- Forsøg at løse opgaverne I.5 og II.2–5 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.