(Beknopte) antwoorden opgaven werkcollege 7

Opgave 39

a. Insertion sort is stabiel. Een element wordt bij het invoegen geplaatst voor het eerste element dat groter is; het element passeert geen waarden van gelijke grootte. Een element van dezelfde grootte links van dit element in het oorspronkelijke array blijft er dus links van staan. Dit zou trouwens anders zijn als de test $A[j] \geq x$ was.

Mergesort is stabiel, mits juist geïmplementeerd; het linkerrijtje moet voorrang krijgen als je twee gelijke waarden tegenkomt bij het samenvoegen. Als we de test $A[i] < A[j]$ veranderen in $A[i] \leq A[j]$ blijft de onderlinge volgorde van gelijke elementen behouden.
De implementatie in het dictaat (en college) is niet stabiel; het rijtje $1_a, 2_a, 1_b, 2_b$ (dus twee keer de waarde 1 en twee keer de waarde 2; subscript om gelijke waarden te onderscheiden) wordt bijvoorbeeld gesorteerd tot $1_b, 1_a, 2_b, 2_a$.

b. Bubblesort (in onze implementatie) is stabiel; een element bubbelt niet verder naar achteren dan een ander (gelijk) element dat erachter staat.

Slowsort is niet stabiel. Neem bijvoorbeeld het rijtje $3, 2_a, 2_b, 1$. Dit wordt door Slowsort gesorteerd tot $1, 2_b, 2_a, 3$. Ga dit zelf na.

Extra: Shellsort is niet stabiel. Elke ronde gaat via Insertion sort, dus lijkt stabiel te zijn, maar de deelrijtjes kunnen vreemd interfereren. Bijvoorbeeld: stapgroottes $2, 1$ en rijtje $1_a, 2_a, 2_b, 1_b, 3$. Dit wordt na 2-sorteren $1_a, 1_b, 2_b, 2_a, 3$ en blijft zo na 1-sorteren.

Opgave 40

a. Ronde 1: de twee buitenste elementen zijn niet kleiner dan en niet groter dan de pivot (ze zijn immers gelijk). Vervolgens worden ze verwisseld. In de volgende iteraties stoppen zowel j als i wederom meteen en worden de betreffende twee elementen verwisseld. Per iteratie worden er steeds 2 vergelijkingen gedaan; we doen 5 iteraties voordat i en j elkaar passeren, dus dat komt neer op 10 vergelijkingen. In alle iteraties behalve de laatste wisselen we, dus dat zijn 4 verwisselingen. Er wordt geeindigd met $j = 4$ en $i = 5$. Hetzelfde doen we vervolgens twee keer voor een array van 4 elementen, dus 6 vergelijkingen en 2 verwisselingen per deel, voor een totaal van 12 vergelijkingen en 4 verwisselingen. Daarna nog op vier deelarrays van 2 elementen, dus op elk deelarray 4 vergelijkingen en 1 verwisseling, met een totaal van 16 vergelijkingen en 4 verwisselingen. Dit alles bij elkaar optellende komen we uit op 38 vergelijkingen en 12 verwisselingen.

b. Merk op: dit is een generalisatie van het vorige onderdeel van deze opgave. We hebben in dit geval voor Partitie van m getallen $m + 2$ vergelijkingen nodig en $\frac{m}{2}$ verwisselingen. Het array wordt telkens in twee gelijke delen gesplitst tot we op rijtjes van één element uitkomen. Als we beginnen met een array ter grootte $n = 2^k$ moeten we dus $k = \lg n$ rondes doen: in ronde 1 Partitie op

n dezelfde elementen, vervolgens in ronde 2 twee keer Partitie op $\frac{n}{2}$ dezelfde elementen, ronde 3 vier keer Partitie op $\frac{n}{2^2}$ dezelfde elementen, etcetera.

We doen zo $n + 2^i$ vergelijkingen in ronde i (ga dit zelf na). In totaal $k = \lg n$ rondes geeft:

$$\sum_{i=1}^{k}(n + 2^i) = kn + 2 \cdot \sum_{i=0}^{k-1} 2^i = n \lg n + 2n - 2$$

vergelijkingen

c. Quicksort is niet stabiel. Neem bijvoorbeeld het rijtje $5_a, 5_b, 5_c$. De twee buitenste elementen worden gewisseld, waardoor ze in verschillende delen terechtkomen, en dus niet meer in de originele volgorde komen te staan. In het voorbeeld: $5_a, 5_b, 5_c \longrightarrow 5_b, 5_c, 5_a$.

d. Deze is voor de echte liefhebbers.
- aflopend gesorteerd: $\frac{n}{2}(n + 4) - 2$, $\frac{n}{2}$ verwisselingen
- oplopend gesorteerd: 0 verwisselingen. Het aantal vergelijkingen wordt niet gevraagd, maar is $\frac{n}{2}(n + 3) - 2$ (zie ook college).

Opgave 41

a. Het rijtje wordt precies in twee gelijke helften gepartitioneerd ($1, 2, \ldots, \frac{n}{2}$ en $\frac{n}{2} + 1, \ldots, n$). Dit in één verwisseling (of twee als je de aanvankelijke wissel met $A[p]$ ook meetelt) en $n + 2$ vergelijkingen. Vervolgens recursief verder op soortgelijke rijtjes.

b. Dit resulteert in $q = i - 1$.

c. Dit doet in totaal $11 + 10 + 11 + 9 + 9 = 50$ vergelijkingen, tegenover 63 voor het gesorteerde rijtje (een slecht geval) en 43 in de best case. Dit zit dus dichter bij het beste geval.

d. Je doet er telkens twee rondes over met (ongeveer) $2m+1$ vergelijkingen om een splitsing te bereiken die minstens even goed is als we in één ronde ongeveer bereiken bij de beste splitsing. Voorbeeld: $25 \longrightarrow 1, 12, 12$ versus $25 \longrightarrow 12, 13$. In dat (beste) geval doen we $m + 1$ vergelijkingen. We doen dus dubbel zo veel werk om net iets minder werk over te hebben. Netto zal het goed/slechtpartionerende algoritme maximaal dubbel zo veel werk doen, dus hooguit een constante factor 2 meer aan vergelijkingen, dus dezelfde orde van grootte.

Opgave 42

(i) $4 + 9 + 13 + 18 = 44$ vergelijkingen
(ii) $5 + 14 + 23 = 42$ vergelijkingen
(iii) $9 + 21 = 30$ vergelijkingen

Opgave 43

Zie uitwerking website