Beknopte antwoorden opgaven werkcollege 1

Opgave 1

a. Basisoperatie: vermenigvuldigingen (van reële getallen). Het meest voor de hand liggende algoritme zal herhaald met x vermenigvuldigen, beginnend met 1 (het geval $n = 0$). Aantal keren: n.
Indien gegeven zou zijn dat $n > 0$, kun je beginnen met x en doe je $n - 1$ vermenigvuldigingen.

b. $\lfloor \lg n \rfloor$ vermenigvuldigingen voor het vinden van alle tweemachten. Hooguit $\lfloor \lg n \rfloor$ vermenigvuldigingen voor het combineren van hooguit $\lfloor \lg n \rfloor + 1$ tweemachten. Totaal dus hooguit $2\lfloor \lg n \rfloor$.

c. Het algoritme uit **b.** gebruikt zes vermenigvuldigingen voor x^{15}. Het kan echter in vijf: $x^2 = x \times x$, $x^3 = x^2 \times x$, $x^5 = x^3 \times x^2$, $x^{15} = x^5 \times x^5 \times x^5$. Nee dus.

Opgave 2

a. $\begin{pmatrix} 26 & 10 \\ 82 & 34 \end{pmatrix}$ 8 vermenigvuldigingen en 4 optellingen.

b. n^3 vermenigvuldigingen en $n^2(n-1)$ optellingen.

c. De invoer bestaat uit twee $n \times n$-matrices, dus in totaal $k = 2 \cdot n^2$ elementen. Dat betekent dat $n = \sqrt{\frac{k}{2}}$, en als we de complexiteit uitdrukken in k, krijgen we: $\frac{k}{2} \cdot \sqrt{\frac{k}{2}}$ vermenigvuldigingen en $\frac{k}{2} \cdot (\sqrt{\frac{k}{2}} - 1)$ optellingen.

d. Je kan een vermenigvuldiging van twee gehele getallen ook uitrollen tot een (lange) reeks optellingen, dus het kan dan technisch gezien ook zonder vermenigvuldigingen. Je hebt in dat geval 0 vermenigvuldigingen, maar dit is nu geen goede maat meer voor de hoeveelheid werk. Om de complexiteit te bepalen moet je naar optellingen en vermenigvuldigingen kijken.

e. Je zult in ieder geval elk van de n^2 elementen in de uitvoer moeten berekenen. Dit kost toch zeker minstens één operatie per element, waaruit men kan concluderen dat de *complexiteit* ten minste $\Omega(n^2)$ is.

f. Allebei mogelijk, maar je kan geen van beide uitspraken concluderen uit deze constatering. Misschien bestaat er wel degelijk een algoritme met complexiteit $O(n^2)$, misschien kunnen we een scherpere ondergrens bewijzen. Of er bestaat een algoritme met complexiteit tussen de ondergrens uit **d.** en de complexiteit uit **b.** in. In dat geval kan het kennelijk beter en is de ondergrens (wellicht ook) niet scherp.
Merk op: dit laatste blijkt het geval (algoritme van Strassen en verbeteringen daarop). Het is echter nog steeds niet duidelijk of de $\Theta(n^2)$ gehaald kan worden.

Opgave 3

Zie ook de uitgebreidere uitwerking op de website.

a. Maatgevend voor de complexiteit, en een fundamentele operatie voor het oplossen van het probleem.

b. 1, als $A[1] = X$.

c. n, als $X \notin \{A[1], \ldots, A[n-1]\}$ (d.w.z. als X niet voorkomt in A of als $X = A[n]$ maar X is ongelijk aan alle voorgaande elementen)

d. Als $X \in A$: $\frac{1}{n}\sum_{i=1}^{n} i = \frac{1}{n} \times \frac{1}{2}n(n+1) = \frac{1}{2}(n+1)$. Als $X \notin A$: n.

Laat $0 \leq q \leq 1$ de kans zijn dat $X \in A$. Dan is de average case complexity $q \times \frac{1}{2}(n+1) + (1-q)n \in \Theta(n)$ (los na te gaan voor $q = 0$, $q = 1$ en $0 < q < 1$).

e. Zie uitwerking website.

Opgave 4

a. 1, als $A[1] \geq X$.

b. n, als $A[n-1] < X$. Dit is dezelfde worst case-complexiteit, maar er zijn minder worst case-rijtjes.

c. Als $X \in A$: $\frac{1}{n}\sum_{i=1}^{n} i = \frac{1}{2}(n+1)$ (zie 3d).

Als $X \notin A$: $\frac{1}{n+1}(\sum_{i=1}^{n} i + n) = \frac{1}{2}n + \frac{n}{n+1}$ (i vergelijkingen als $A[i-1] < X < A[i]$ of (voor $i = 1$) als $X < A[1]$, en los nog eens n vergelijkingen als $X > A[n]$)

Laat q de kans zijn dat $X \in A$. Dan:

$$
\begin{aligned}
& q \times \frac{1}{2}(n+1) + (1-q)(\frac{1}{2}n + \frac{n}{n+1}) \\
= & \frac{1}{2}n + \frac{n}{n+1} + \frac{1}{2}qn + \frac{1}{2}q - \frac{1}{2}qn - q \times \frac{n}{n+1} \\
= & \frac{1}{2}n + \frac{n}{n+1} + q(\frac{1}{2} - \frac{n}{n+1}) \\
\in & \Theta(\frac{n}{2})
\end{aligned}
$$

Opgave 6

Zie uitwerking website.