1e deeltentamen Datastructuren 2014/2015

Je hebt twee uur voor dit deeltentamen.

Schrijf op ieder vel je naam. Schrijf op het eerste vel je collegekaartnummer, en het aantal ingeleverde bladen.

Bij verschillende opgaven wordt om een algoritme gevraagd. Probeer hierbij duidelijke antwoorden te geven. Pseudocode met een toelichting in tekst is daarbij een goede optie. Als om een tijdgrens wordt gevraagd, geef dan een 'zo goed mogelijke grens', waarbij je steeds naar keuze van de O-notatie of de Θ-notatie gebruik mag maken.

Schrijf leesbaar en netjes. Zet je mobiel uit voor aanvang van het tentamen. Je mag geen rekenmachine gebruiken, maar wel vier vellen A4 met zelfgemaakte aantekeningen. Als je deze aantekeningen met een computer hebt gemaakt, dan moet je ze bij je werk inleveren. Handgeschreven aantekeningen mag je mee naar huis nemen na afloop van het tentamen.

Als om een algoritme met een bepaalde tijdgrens gevraagd wordt, en het lukt je niet om die te halen, dan kan je mogelijk een (evt. kleine) deelscore halen met een langzamer algoritme. Het is het handigst om zo'n beantwoording pas op te schrijven nadat je de andere vragen van het tentamen gemaakt hebt.

Veel succes!

1. O, Ω, en Θ (1.5 punt)

Zeg van elk van de volgende beweringen of ze waar zijn of onwaar. Je hoeft je antwoord niet toe te lichten. Normering: -0.5 voor elk fout antwoord, -0.25 voor elke niet beantwoorde vraag. (Je kan geen negatieve score halen)

1. $3n^2 + 4n \log n + 5n = O(n^2)$: WAAR of ONWAAR
2. $3n^2 + 4n \log n + 5n = \Omega(n^2)$: WAAR of ONWAAR
3. $3n^2 + 4n \log n + 5n = \Theta(n^2)$: WAAR of ONWAAR
4. $6\sqrt{n} \cdot n = O(n)$: WAAR of ONWAAR
5. $6\sqrt{n} \cdot n = \Omega(n)$: WAAR of ONWAAR
6. $10^{10} + 1/n = \Theta(1)$: WAAR of ONWAAR

Bij deze vraag hoef je dus alleen steeds WAAR of ONWAAR op te schrijven voor elk van de onderdelen.

2. Functies en Θ-notatie (0.5 punt)

Schrijf de volgende functie zo compact mogelijk in Θ-notatie. Voorbeeld: $3n^{\sqrt{4}} + 2n^2 = \ldots = \Theta(n^2)$

$$3 \cdot \sqrt{n} \cdot 2^{\log n} \cdot \log(n^3) + 4 \cdot n^{2/3} \cdot \log(n^4) = \Theta(\cdots)$$

3. Analyse van algoritmen I: 1 punt

Maak een zo goed mogelijke tijd-analyse (worst-case) van methode DUURLOOP, in termen van grote O van een functie in n. Je beantwoording moet ook uitleg bevatten hoe je aan het antwoord komt.

methode DUURLOOP (int n, array A)
Input: integer n; integer array $A[1 \cdots n]$
Uitvoer: integer k

```
k = 1;
r = ⌊n/3⌋;
for i = 1 to r
do k++;
   z = n;
   while z > 1
   do k = k + A[z] ;
      z = z/3 ;
   enddo ;
enddo;
return k;
```

4. Analyse van algoritmen II: 1 punt

Maak een zo goed mogelijke tijd-analyse (worst-case) van methode SPRINTJE, in termen van grote O van een functie in n. Je beantwoording moet ook uitleg bevatten hoe je aan het antwoord komt.

methode SPRINT (int n, array A)
Input: integer n; integer array $A[1 \cdots n]$
Uitvoer: integer k

```
a = 1;
for i = 1 to n
do for j = 1 to n
   do for k = 1 to ⌊n/j⌋
      do a = a + 1;
      enddo;
   enddo;
enddo;
return a;
```

5. Dutch National Flag (2 punten)

Het **Dutch National Flag probleem** is het volgende probleem. Gegeven is een array $B[1 \cdots n]$ van n elementen. Ieder element heeft een *kleur* en een *waarde.* De kleur kan zijn: rood (R), wit (W) of blauw (B); de waarde is een positieve integer. We moeten de elementen in de array herrangschikken, zodat eerst alle rode elementen komen, dan alle witte elementen, en dan alle blauwe elementen. De onderlinge volgorde voor de elementen met dezelfde kleur maakt niet uit.

Een voorbeeld: stel de invoer is R1, W4, R7, B10, B8, W11, W13. Een correcte uitvoer is: R1, R7, W4, W11, W13, B10, B8. Een ander voorbeeld van een correcte uitvoer is: R7, R1, W4, W13, W11, B8, B10.

1. **Geef een zo efficient mogelijk algoritme voor het Dutch National Flag** probleem. (We meten de efficiente door de looptijd van je algoritme als functie van n te bekijken, met O-notatie of Θ-notatie.) **Je algoritme mag hooguit een constante hoeveelheid extra geheugenplaatsen gebruiken.** (Je mag dus bijvoorbeeld niet extra arrays van lengte n aanmaken.)

2. Beargumenteer de correctheid van je algoritme. Als de correctheid van je algoritme triviaal is, dan hoeft dit niet.

3. Hoeveel tijd (in O-notatie of Θ-notatie) gebruikt je algoritme? Beargumenteer dit.

6. Analyse van algoritmen II: 2 punten

Beschouw het volgende recursieve algoritme.

method TREK (array A; integer n; integer m)
Input: integer n; integer array $A[1 \cdot n]$; integer m
Uitvoer: integer k

```
if n ≤ 3 then return A[1] + m;
else
    onethird = ⌊n/3⌋;
    create arrays B[1···onethird]; C[1···onethird]; D[1···onethird];
    for i = 1 to onethird
    do B[i] = A[i];
       C[i] = A[i+ onethird];
       D[i] = A[i + 2· onethird];
    enddo;
    x = TREK(B,onethird,1)+ TREK(C, onethird, 1) + TREK(D, onethird, 1);
    y = TREK(B,onethird,2)+ TREK(C, onethird, 2) + TREK(D, onethird, 2);
    z = TREK(B,onethird,3)+ TREK(C, onethird, 3) + TREK(D, onethird, 3);
    return max{x − m, y + m, z};
endif
```

In deze opgave gaan we de tijd van dit algoritme analyseren.

(a) **Stel een recurrente betrekking op** van de tijd van dit algoritme, als functie van n.

(b) **Hoeveel tijd kost dit algoritme**, in Θ-notatie, als functie van n? **Beargumenteer je antwoord, met hulp van de master-theorem, en je antwoord bij deel (a).**

Mocht onderdeel (a) niet lukken, dan kan je deelpunten voor deze vraag halen door de volgende recurrente betrekking op te lossen.

$$T(n) = 3T(n/3) + \sqrt{n}$$

Normering: deel (a) goed en deel (b) fout: 1 punt. Deel (a) fout, maar eigen antwoord in deel (b) goed doorgerekend: 1 punt. Deel (a) fout, maar in de opgave gegeven betrekking goed doorgerekend: 1 punt. Deel (a) goed, en de in de opgave gegeven betrekking, niet de eigen gevonden goed doorgerekend: 1.5 punt. Deel (a) goed en eigen antwoord in deel (b) goed doorgerekend: 2 punten.

7. De k-grootste elementen: 2 punten

Bekijk het volgende probleem. Gegeven is een array $A[1 \cdots n]$ van n integers en een getal k. De getallen in A staan in willekeurige volgorde. Lever op: de k grootste getallen van A in gesorteerde volgorde, van klein naar groot.

Een voorbeeld: Stel A heeft achtereenvolgens de getallen $3, 7, 2, 100, 8, 4$ en $k = 3$. De output is dan $100, 8, 7$.

Je mag aannemen dat alle integers in A verschillend zijn.

1. **Geef een algoritme dat dit probleem oplost in $O(n + (k \log n))$ tijd.** Je mag hierbij gebruik maken van in het college behandelde resultaten. Methoden die behandeld zijn mag je aanroepen zonder dat je de bijbehorende (pseudo)code geeft. Tijdgrenzen die bewezen zijn mag je ook aannemen zonder ze hier zelf te bewijzen.

2. Beargumenteer dan de correctheid van je algoritme. Als de correctheid van je algoritme triviaal is, dan hoeft dit niet.

3. Leg kort uit waarom je algoritme $O(n + k \log n)$ tijd gebruikt.