Tentamen Data Structures and Algorithms

19 Oktober 2009

Dit tentamen bestaat uit 8 opdrachten.

Opdracht 1.

(a) Gegeven is het volgende algoritme:

```
Loop(n):
    s = 0
    for i = 1 to n do
        for j = 1 to 6 do
            s = s + 1
        done
    done
```

Wat is de tijdscomplexiteit van dit algoritme in termen van O?

(5 punten)

(b) Vervang elk vraagteken door een zo simpel mogelijke expressie:

(i) $n + n^3 + 3 \cdot n^2 \quad \in O(?)$

(ii) $n + 2 \cdot \log_2 n \quad \in O(?)$

(iii) $4^n + 5 \cdot n^5 \quad \in O(?)$

(6 punten)

(c) Los de volgende recurrente betrekking op:

$$T(n) = \begin{cases} 1, & \text{als } n \leq 1 \\ 2 \cdot T(\frac{n}{2}), & \text{als } n > 1 \end{cases}$$

(5 punten)

Opdracht 2.

(a) Geef de pseudo-code voor het implementeren van de operaties push en pop van het ADT voor de stack, gebruikmakend van twee queues Q_1 and Q_2.

(5 punten)

(b) Geef de worst-case tijdscomplexiteit van de methoden push en pop uit je antwoord op vraag (a) in termen van O. (We gaan uit van queue operaties in $O(1)$.) Het antwoord volstaat.

(4 punten)

Opdracht 3.

(a) We implementeren een priority queue met een AVL boom. Wat is de tijdscomplexiteit van enqueue en dequeue in termen van O? Motiveer je antwoord.

(5 punten)

(b) Beschouw weer een priority queue geïmplementeerd met een AVL boom. Wat is de tijdscomplexiteit voor het sorteren van een (ongesorteerde) lijst gebruikmakend van deze priority queue? Motiveer je antwoord.

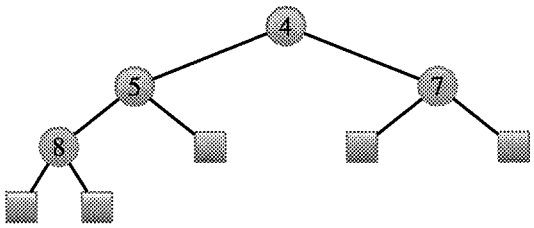
(5 punten)

(c) Beschrijf (hoeft niet in pseudo-code) hoe insertion-sort werkt. Wat is de tijdscomplexiteit van insertion-sort? Wanneer is het goed om insertion-sort te gebruiken? Motiveer je antwoord.

(6 punten)

Opdracht 4.

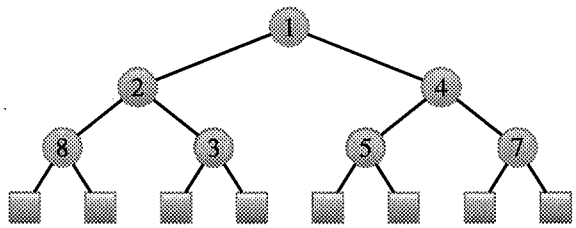
(a) Voeg 3 toe aan de volgende heap (geef alleen het eindresultaat):



Voeg vervolgens (aan het resultaat) 2 toe (geef weer alleen het eindresultaat).

(6 punten)

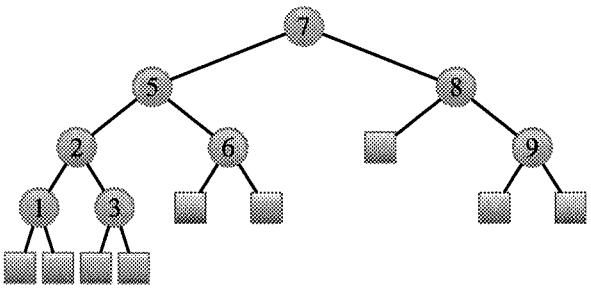
(b) Verwijder de key 1 van de volgende heap (geef alleen het eindresultaat):



(5 punten)

Opdracht 5.

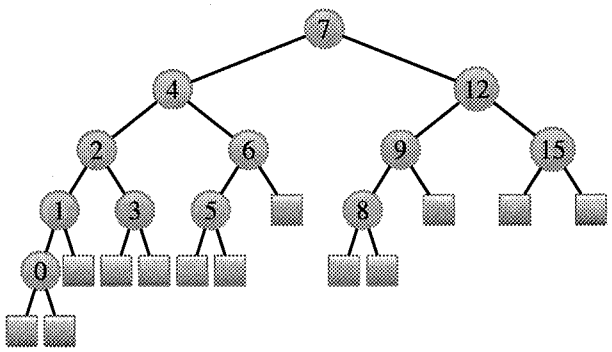
(a) Gegeven is de volgende AVL boom:



Geef stap voor stap aan hoe een knoop met key 0 wordt toegevoegd.

(5 punten)

(b) Gegeven is de volgende AVL boom:



Geef stap voor stap aan hoe de knoop met key 15 wordt verwijderd.

(7 punten)

Opdracht 6.

(a) Maak een hash tabel van grootte 7. Gebruik de hash functie $h(k) = k \mod 7$. Pas linear probing toe op de initieel lege hash tabel om de volgende getallen (in deze volgorde) toe te voegen:

$$9,\ 3,\ 2,\ 15,\ 1$$

(5 punten)

(b) Wat is de:

(i) average (expected), en

(ii) worst-case

tijdscomplexiteit (in termen van O) voor het zoeken van een key k in een hash tabel (gebruikmakend van chaining)?

(4 punten)

Opdracht 7.

(a) Gegeven is de volgende text/string T:

a	b	a	c	a	a	b	c	a	d	a	b	a	c	a	b	a	a	b	b
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

samen met het volgende patroon P:

a	b	a	c	a	b
0	1	2	3	4	5

Pas het Boyer-Moore pattern matching algoritme toe om P in T te zoeken.

(5 punten)

(b) Gegeven is de volgende text/string T:

a	b	a	a	b	c	a	b	a	b	a	a	b	a	a	b	a	a	b	b
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

samen met het volgende patroon P:

a	b	a	a	b	a
0	1	2	3	4	5

Pas het Knuth-Morris-Pratt pattern matching algoritme toe om P in T te zoeken.

(5 punten)

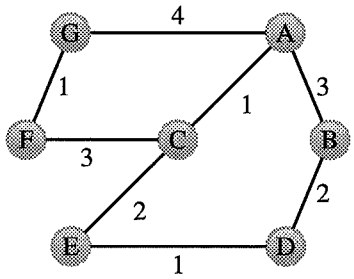
Notatie voor Opdracht 7:
Geef per stap aan welke letter uit P vergeleken wordt met welke letter van T, en geef ook aan hoe P opgeschoven wordt. Een voorbeeld van deze notatie bij toepassen van het brute-force algoritme op (a):

1. $P[0] = T[0]$
2. $P[1] = T[1]$
3. $P[2] = T[2]$
4. $P[3] = T[3]$
5. $P[4] = T[4]$
6. $P[5] \neq T[5]$
7. patroon wordt met 1 positie opgeschoven (nieuwe positie is 1)
8. $P[0] = T[1]$

...

Opdracht 8.

Gegeven is de volgende graaf:



Geef stap voor stap aan hoe de minimal spanning tree (MST) wordt berekend gebruikmakend van het Prim-Jarnik algoritme. Begin met knoop G en geef de volgorde aan waarin de knopen worden bekeken.

(7 punten)

The grade is the (total amount of points plus 10) divided by 10.