Deeltentamen DataStructuren 25 juni 2003
docent Marinus Veldhorst

Schrijf niet met rood en niet met potlood; zet op elk in te leveren vel:

- je naam (met initialen),
- collegekaartnummer,
- je werkcollegeleider; liefst de werkcollegeleider die je tentamen van 11 juni j.l. heeft nagekeken. Deze werkcollegeleider zal je werk nakijken en heeft (als het goed is) met de werkcollegegroep afgesproken hoe en wanneer het werk teruggegeven wordt.
- Zet op het eerste vel het totaal aantal vellen papier dat je inlevert.

Je hebt 1 uur en 3 kwartier de tijd voor dit deeltentamen.

9. **Hashing (1.5 punt)**
We willen een hashtabel van 13 posities vullen met een nog nader te bepalen verzameling positieve gehele getallen. Als hashfunctie gebruiken we de functie h gegeven door

$$h(i) = (2i + 1) \bmod 13$$

Collision handling doen we met behulp van quadratic probing.

Voeg de volgende getallen in in de hashtabel in de aangegeven volgorde

15 41 8 54 22 80 28 67 4

10. **Zoekbomen met elementen in de bladen**
Het boek behandelt zoekbomen als een manier om in een geordende verzameling S te kunnen zoeken, invoegen en verwijderen. Daarbij worden elementen van S opgeslagen in de interne knopen. Zoeken in S wordt veelal gedaan op basis van keys waarbij elk element een eigen unieke key heeft.

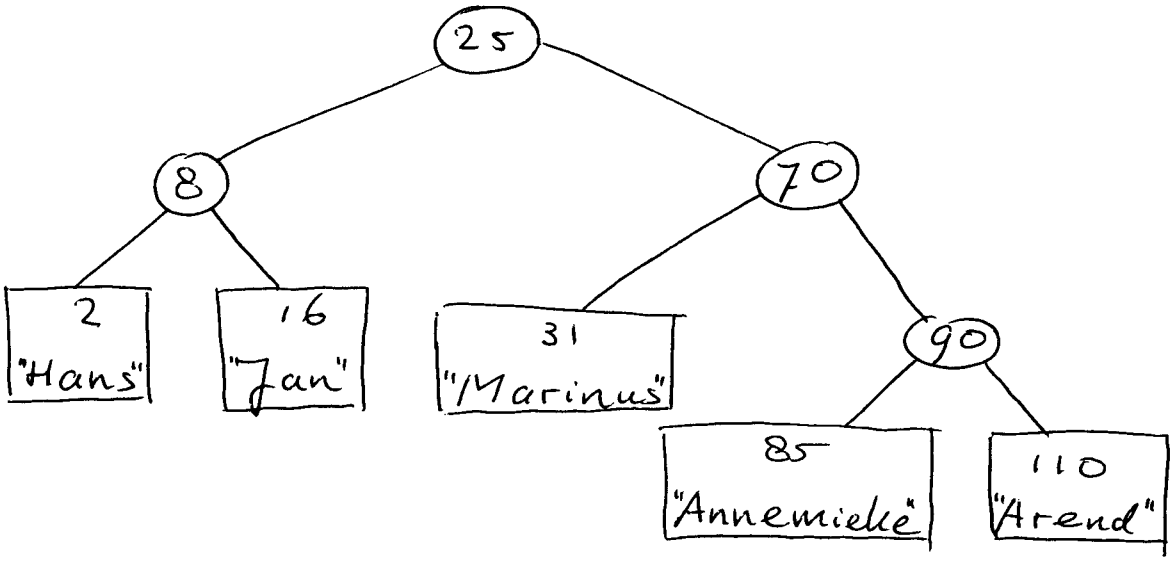
Z.O.Z.

1

Een andere soort zoekboom voor S is een zoekboom waarin elementen van S alleen maar in de bladen (externe knopen) zitten. De interne knopen bevatten keys (niet noodzakelijkerwijs voorkomend in S) die de weg moeten wijzen naar de gezochte elementen (of bij Insert: de plaats waar het nieuwe element moet komen). De elementen van S zitten in stijgende volgorde van hun keys in de externe knopen van de boom. Om precieser te zijn: bij een inorder traversal kom je de keys in interne en externe knopen van T tegen in stijgende (of in ieder geval niet-dalende) volgorde.
Beschouw als voorbeeld een verzameling S van elementen die elk bestaan uit een geheel getal (de key) en een string van characters:

$$S = \{ \ (31, "Marinus"),\ (16, "Jan"),\ (2, "Hans"),\ (110, "Arend"),\ (85, "Annemieke") \ \}$$

De zoekboom voor S zou er dan als volgt uit kunnen zien:



Zij nu in het algemeen T een dergelijke boom waarin een verzameling S in de bladen is opgeslagen. In interne knopen is slechts opgeslagen: een key, een pointer naar het linker kind en een pointer naar het rechter kind, maar **geen** parent pointer.
Neem aan dat S uit n elementen bestaat, en dat T hoogte h heeft. We gaan ervan uit dat twee elementen van S hetzelfde zijn als hun keys gelijk zijn, en dat S nooit dubbele elementen mag bevatten.

(a) **(2.5 punt)**
Geef $O(h)$ tijd algoritmen voor *Findelement*(*key* k), *Insert*(*key* k, *element* e), en *Remove*(*key* k). Geef je algoritme voor *Insert* (ook) in pseudocode.
Geef uitleg betreffende correctheid en behaalde tijdgrens voor je algoritmen voor de drie operaties.

(b) **(1 punt)** We willen een extra operatie op S uitvoeren, nl. een operatie *count* die, gegeven twee keys p en q, telt het aantal elementen $s \in S$ met $p < s.key < q$. Geef een algoritme voor *count* zodanig dat een aanroep ervan met uitkomst k in $O(h + k)$ tijd afgehandeld wordt.
Leg uit dat je algoritme correct is, en aan de gestelde tijdgrens voldoet.

2

11. **(2.5 punt)**
Zij T een binaire zoekboom waarbij de elementen en hun keys opgeslagen zijn in de interne knopen. De keys zijn integers. We gaan ervan uit dat per interne knoop van T opgeslagen zijn: element en key, pointer naar linker kind, en pointer naar rechterkind. Met name zijn de parent-pointers **niet** opgeslagen.

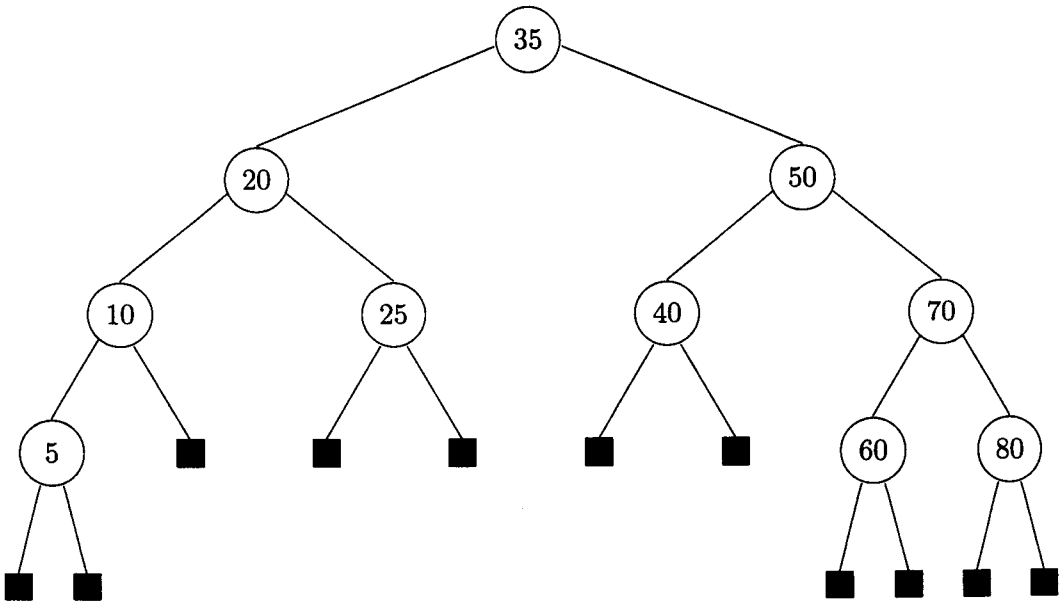
Q We willen voor een gegeven key k (k komt niet noodzakelijkerwijs voor in T), een key x in T bepalen die het dichtst bij k ligt (oftewel een key x in T die $|x-k|$ minimaliseert; als k in T voorkomt, dan is het antwoord k).

(a) Geef een $O(h)$ tijd algoritme voor bovenstaand probleem **Q**, waarbij h de hoogte van de binaire zoekboom is. Het is wenselijk dat in je algoritme je maar één keer een zoekpad vanaf de wortel afloopt.
Beargumenteer dat je algoritme correct is, en toon aan dat je algoritme voldoet aan de gestelde tijdgrens.

12. **AVL bomen**

(a) **(0.5 punt)** Geef de recursieve definitie van een AVL-boom.

(b) **(2 punten)** Gegeven de volgende AVL-boom T_0 met integers als keys.



Geef aan hoe de AVL-bomen T_1, T_2, T_3, T_4, T_5 er uit (kunnen) zien (zonodig met relevante tussenstadia), waarbij

T_1 ontstaat uit T_0 door invoeging van een item met key 100
T_2 ontstaat uit T_1 door verwijdering van het item met key 60
T_3 ontstaat uit T_2 door invoeging van een item met key 95
T_4 ontstaat uit T_3 door invoeging van een item met key 105
T_5 ontstaat uit T_4 door verwijdering van het item met key 40

einde deeltentamen