Algoritmusok és adatszerkezetek II.

Kupacok

Szegedi Tudományegyetem

Fapacok (Treaps)

Emlékeztető

n kulcsból álló **véletlen** építésű bináris keresőfa h magasságának **várható értéke** $\log n$

- Adverzaliális műveleti sorrend mellett azonban n magas is lehet

Ötlet

A keresőfa,-és kupactulajdonságot egyidejűleg követeljük meg

1. Keresőfa tulajdonság biztosítja a kulcsok $O(h)$ kereshetőségét
2. Kupactulajdonság miatt h **várható értékben** $\log n$

Fapacok (Treaps)

Emlékeztető

n kulcsból álló **véletlen** építésű bináris keresőfa h magasságának **várható értéke** $\log n$

- Adverzaliális műveleti sorrend mellett azonban n magas is lehet

Ötlet

A keresőfa,-és kupactulajdonságot egyidejűleg követeljük meg

1. Keresőfa tulajdonság biztosítja a kulcsok $O(h)$ kereshetőségét
2. Kupactulajdonság miatt h **várható értékben** $\log n$
   - A kupactulajdonság ne az eltárolt kulcsokra, hanem egy **véletlenszerűen** generált kiegészítőinformációra teljesüljön!

Kupacok

Felhasználásuk

1. Prioritási sor megvalósításánál fontos, hogy a minimális/maximális kulcsot hatékonyan tudjuk visszaadni
2. Szintén fontos művelet egy adott kulcs értékének módosítása

Kupacok

Felhasználásuk

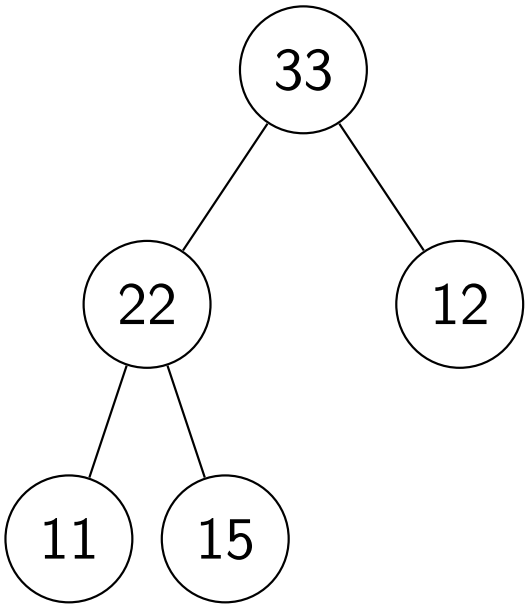
1. Prioritási sor megvalósításánál fontos, hogy a minimális/maximális kulcsot hatékonyan tudjuk visszaadni
2. Szintén fontos művelet egy adott kulcs értékének módosítása

Kupactulajdonság

Azt mondjuk, hogy egy fa rendelkezik a minimum (maximum) kupactulajdonsággal, ha minden p csúcsának minden q fiára

- $q = Nil$ vagy
- $p.kulcs < q.kulcs$ ($p.kulcs > q.kulcs$)

Példa maximum bináris kupacra



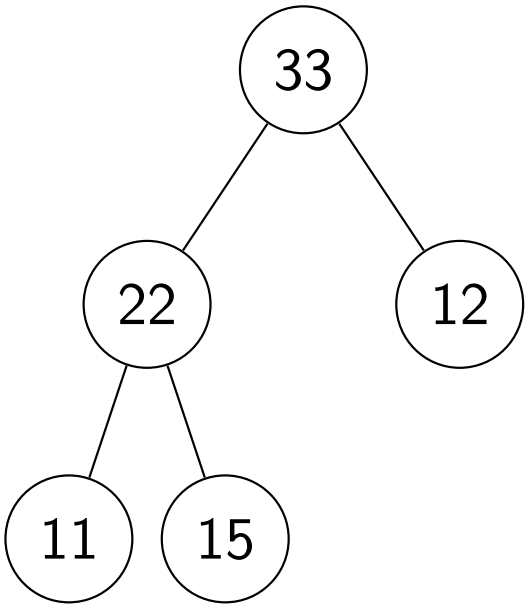
Beszúr(40)

0	1	2	3	4	5
–	33	22	12	11	15

Bináris kupac

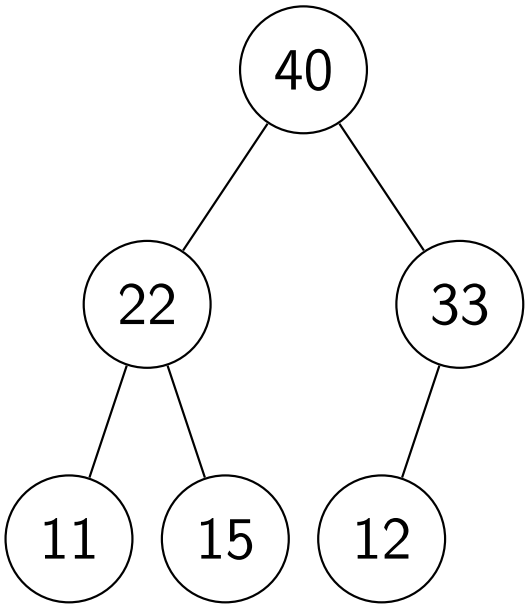
Teljes bináris fa, melyre teljesül a kupactulajdonság.
$\Rightarrow$ mivel legfeljebb egy belső pontnak lehet 2-nél kevesebb fia, így egyszerűen egy tömbbel implementálhatjuk

Példa maximum bináris kupacra



0	1	2	3	4	5
–	33	22	12	11	15

Beszúr(40)

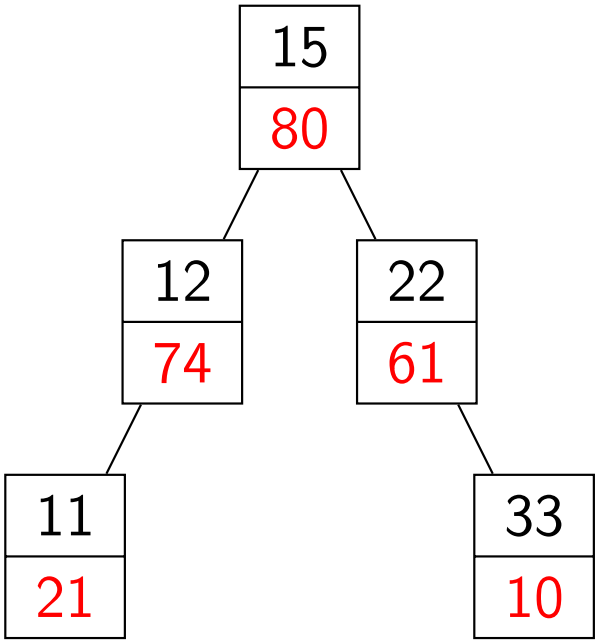


0	1	2	3	4	5	6
–	40	22	33	11	15	12

Bináris kupac

Teljes bináris fa, melyre teljesül a kupactulajdonság.
$\Rightarrow$ mivel legfeljebb egy belső pontnak lehet 2-nél kevesebb fia, így egyszerűen egy tömbbel implementálhatjuk

Fapac példa



- A kulcsok keresőfa tulajdonság szerint helyezkednek el
- A véletlen felépítést az extra adattag eredményezi
- A kiegyensúlyozott fáknál megszokott módon állítjuk helyre a megkövetelt tulajdonságokat (pl. (Beszúr(27, 100)))

Vissza a kupacokhoz

n elemű kupacban hogy keresnénk meg a maximális elemet?
És egy adott kulcs rákövetkezőjét?
Egy n és egy m kulcsból álló kupacot miként egyesítenénk?

Vissza a kupacokhoz

n elemű kupacban hogy keresnénk meg a maximális elemet?	O(1)
És egy adott kulcs rákövetkezőjét?	O(m+n)
Egy n és egy m kulcsból álló kupacot miként egyesítenénk?	O(m+n)

Kérdés

Lehetne hatékonyabban is?