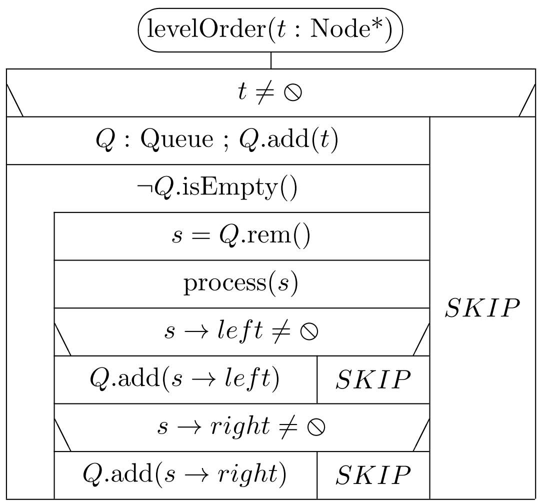
Algoritmusok és adatszerkezetek I, 8. gyakorlat

Téma:

Bináris fa szintfolytontos bejárása, bináris keresőfa és alapműveletei

Szintfolytonos bejárás

Tulajdonképpen egy szélességi bejárásról van szó (ezt a hallgatók még nem tanulták, szóval inkább ne hivatkozzunk rá). A bináris fa csúcsait a mélységük szerinti sorrendben látogatjuk meg, egy szinten belül balról jobbra. Ehhez egy sort használunk, ez egy nagyon jó példa a Queue alkalmazására. Az algoritmus megtalálható a jegyzet 70. oldalán:



Egy apró megjegyzés a struktogramhoz: a jegyzetben a t paraméter absztrakt BinTree típusú, nálunk az alábbiakban is mindig ennek a konkrét láncolt megvalósítása lesz, amiben Node objektumok a bináris fa csúcsai.

Mekkora a műveletigény, és miért? $\Theta(n)$, ahol n a csúcsok száma, hiszen minden csúcs egyszer kerül bele a sorba, és minden iterációban kiveszünk egyet, amivel konstans műveletet végzünk.

Feladatok a szintfolytonos bejárás alkalmazására

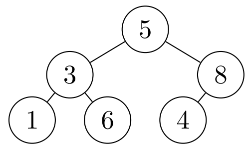
Elsőként egy gyakorló feladat:

1. Levelek száma: számoljuk meg egy bináris fa leveleit szintfolytonos bejárással! Ez egy gyakorló feladat, amit megoldottunk már rekurzívan is. Párhuzam vonható a megszámolás programozási tétellel.

Bináris keresőfa

A keresőfa tulajdonság:

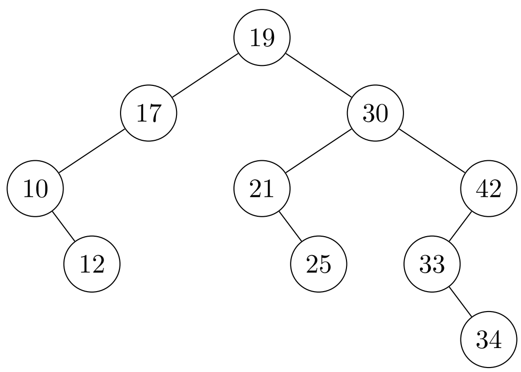
- Egy hibás definíció: minden csúcs balgyerekének kulcsa kisebb, jobbgyerekének kulcsa nagyobb. Miért rossz ez? A bal részfában egy szinttel mélyebben lehetne nagyobb kulcsú elem. Például:



- A helyes definíció: egy csúcs kulcsánál a bal részfájában minden csúcs kulcsa kisebb, a jobb részfájában minden csúcs kulcsa nagyobb.
  Jegyezzük meg, hogy ez nem engedi meg az egyenlőséget, tehát egy bináris keresőfában csupa különböző elemek vannak. A **rendezőfa** elnevezést használjuk, ha megengedjük az egyenlőséget.

Mutassunk egy példát arra, hogy sorozatos beszúrásokkal hogyan épül fel egy bináris keresőfa:

$$19, 17, 30, 10, 21, 12, 42, 25, 33, 34$$



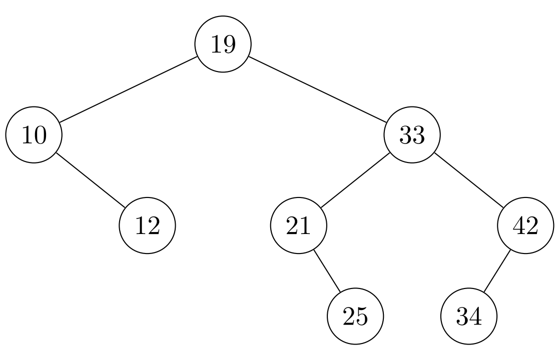
Nézzük meg azt is egy konkrét példán, hogy mi történik egy kulcs keresésekor, merre megyünk, amikor a 25-öt keressük? És ha a 23-at?

Mondják meg azt is, hogy milyen kulcsok kerülhetnek:

(a) 12 jobb részfájába (Válasz: 13..16)

(b) 33 bal részfájába (Válasz: 31..32)

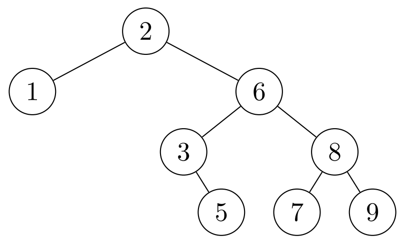
Majd: hogyan kell törölni egy elemet? Mi történjen, ha a 17-et töröljük? És ha a 30-at?



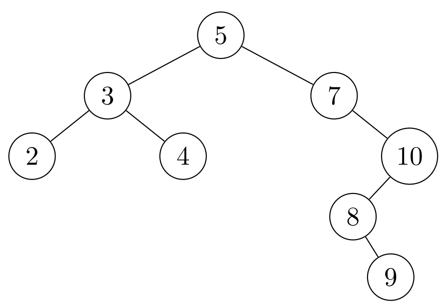
Feladat

Megadjuk egy bináris keresőfa pre/post order bejárását. Ez alapján határozzuk meg a keresőfát.

1. Egy bináris keresőfa preorder bejárásában a kulcsok sorrendje: 2, 1, 6, 3, 5, 8, 7, 9. Rajzold fel a keresőfát!
Megoldás: Az első elem a gyökér, ezt követi a balgyereke, míg a jobbgyereke a sorrendben az első nála nagyobb elem. Ezt kell rekurzívan alkalmazni a részfákra is.



2. Egy bináris keresőfa postorder bejárásában a kulcsok sorrendje: 2, 4, 3, 9, 8, 10, 7, 5. Rajzold fel a keresőfát!
Megoldás: Hátulról érdemes kezdeni. Az utolsó elem a gyökér, ezt előzi meg a jobbgyereke, míg a balgyereke a sorrendben hátulról haladva az első nála kisebb elem. Ezt kell rekurzívan alkalmazni a részfákra is.



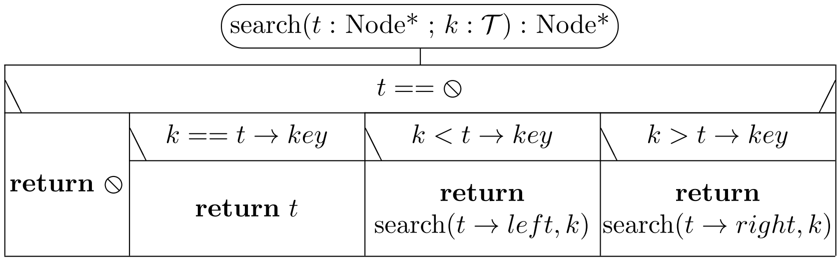
A bináris keresőfa alapműveletei

A bináris keresőfák műveletei:

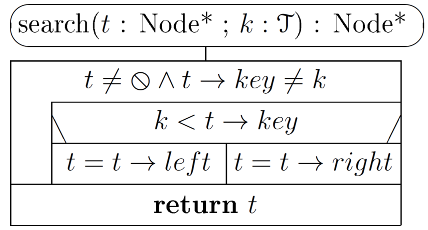
- $search(t, k)$: A t fában megkeresi a k kulcsú elemet, NULL-t ad vissza, ha nincs benne.
- $insert(t, k)$: Beszúrja a k kulcsú elemet, ha még nincs a fában.
- $min(t)$: A minimális kulcsú csúcs címét adja vissza.
- $remMin(t, minp)$: A fából kiveszi a minimális kulcsú csúcsot, és a címét a minp pointerben ada vissza.
- $del(t, k)$: Törli a k csúcs elemet, amennyiben az megtalálható a fában.

Mindegyik művelet $O(h)$, ahol h a fa magassága. Az előadásjegyzetben egyes műveletek rekurzívan, mások iteratívan vannak megvalósítva. Gyakorlaton vegyük őket a másik fajta módszerrel.

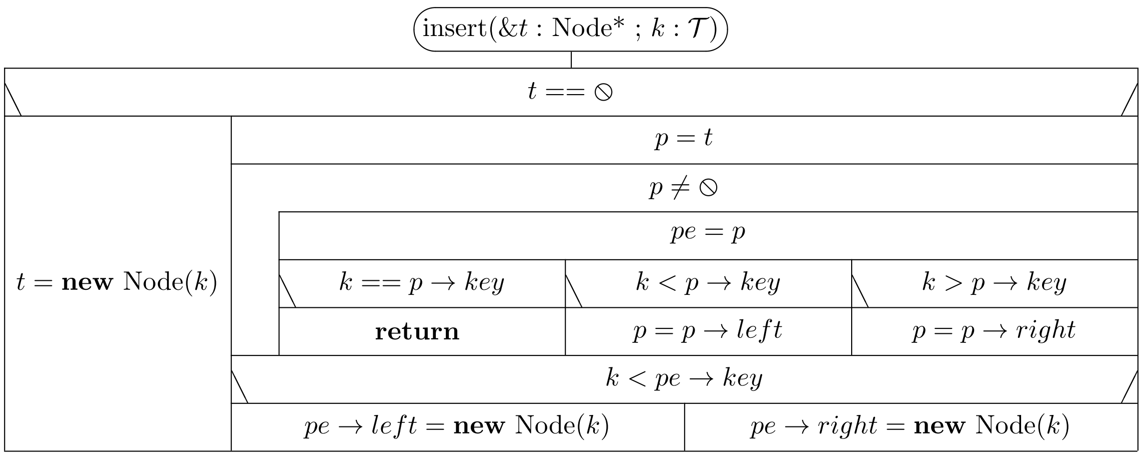
1. A keresés rekurzívan.



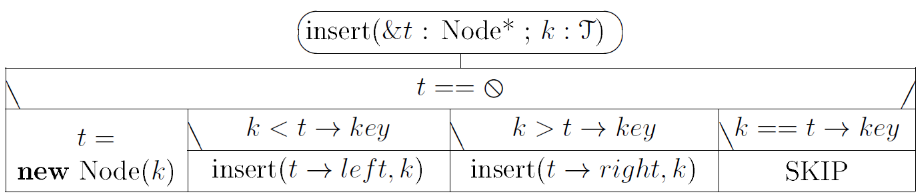
(A jegyzetbeli iteratív verzió:)



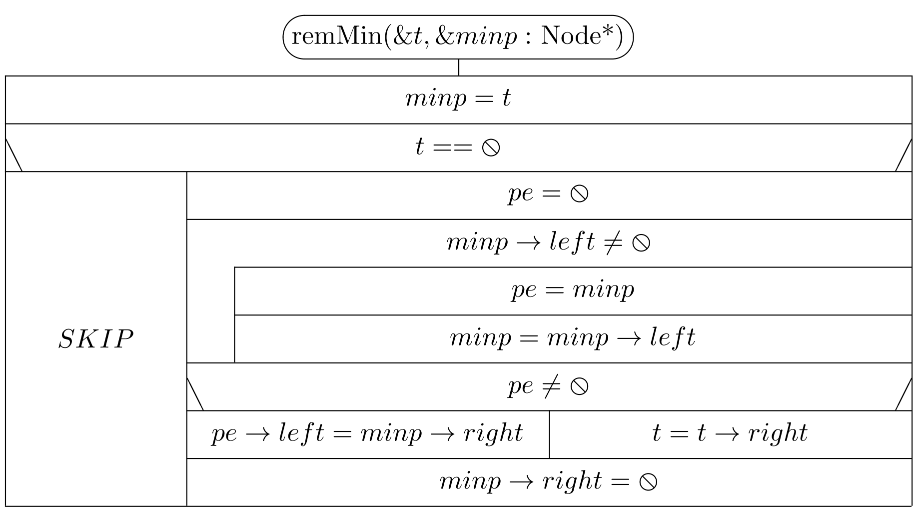
2. A beszúrás iteratívan.



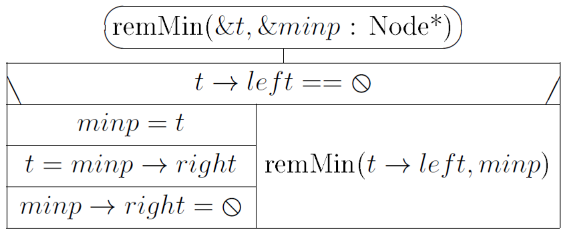
(A jegyzetbeli rekurzív verzió:)



3. (Lehet házi) Minimum törlés iteratívan.

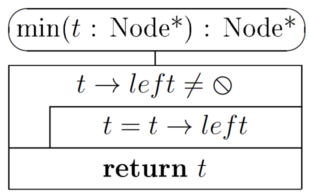


(A jegyzetbeli rekurzív verzió:)



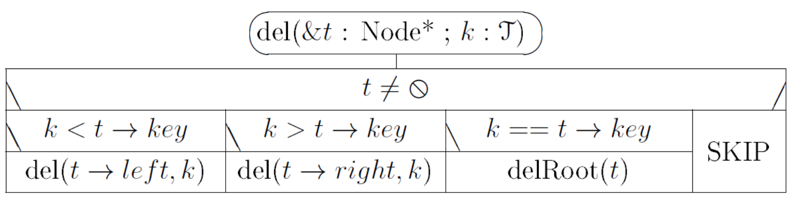
4. A $min(t)$ rekurzívan triviális, ezért ezt kihagyjuk.

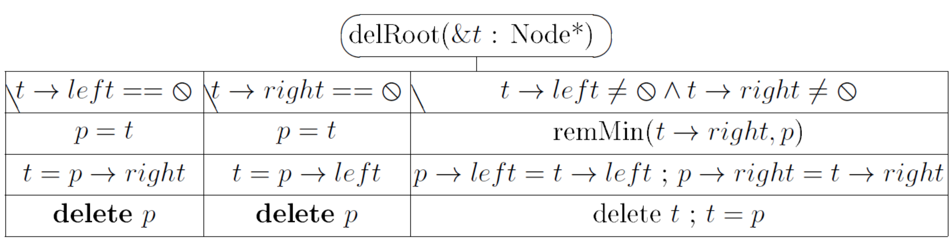
(A jegyzetbeli iteratív verzió:)



5. A törlés iteratív megvalósítását szintén nem tárgyaljuk.

(A jegyzetből a rekurzív verzió:)





Házi feladat javaslatok

1. A szintfolytonos bejárást alkalmazva határozzuk meg egy bináris fában azon levelek számát, amelyek mélysége legalább k.

Megoldás: Mivel volt az órán a szintszámlálós trükk, ez könnyű, csak azt a struktogramot kell kiegészíteni.