Algoritmuselmélet

Keresőfák, piros-fekete fák

Katona Gyula Y.

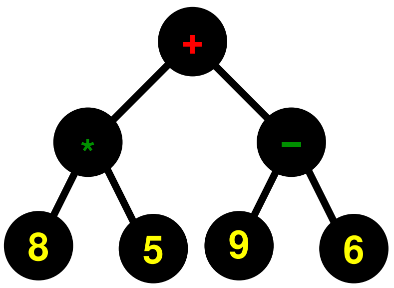
Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

Keresőfák

Tároljuk az U rendezett halmaz elemeit, hogy BESZÚR, TÖRÖL, KERES, MIN, (MAX, TÓLIG) hatékonyak legyenek.

Bináris fa bejárása

teljes fa (új def.): az alsó szint is tele van $\Longrightarrow$ l szintű, teljes fának $2^l - 1$ csúcsa van.
Fa csúcsai $\rightarrow$ $elem(x), bal(x), jobb(x)$ esetleg $apa(x)$ és $reszfa(x)$



Ha x a gyökér, y pedig a 9-es csúcs, akkor

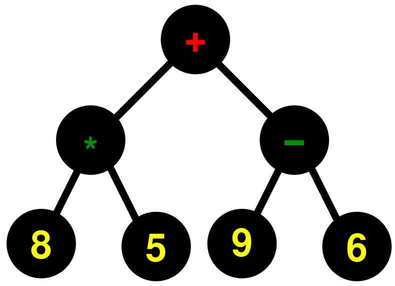
$$\begin{aligned} bal(jobb(x)) &= y, \\ apa(apa(y)) &= x, \\ elem(bal(x)) &= *, \\ reszfa(x) &= 7. \end{aligned}$$

PREORDER, INORDER, POSTORDER

pre(x)
begin
látogat(x);
pre(*bal*(x));
pre(*jobb*(x))
end

in(x)
begin
in(*bal*(x));
látogat(x);
in(*jobb*(x))
end

post(x)
begin
post(*bal*(x));
post(*jobb*(x));
látogat(x)
end



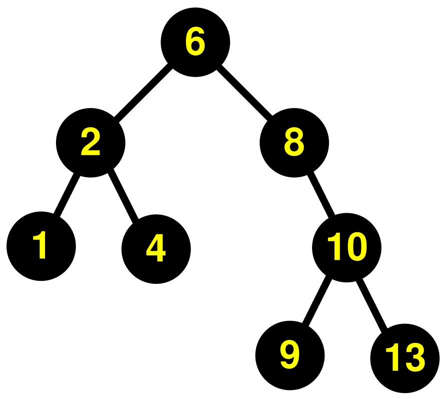
PREORDER: $+ * 8\ 5 - 9\ 6$
INORDER: $8 * 5 + 9 - 6$
POSTORDER: $8\ 5 * 9\ 6 - +$

Lépésszám: $O(n)$

Bináris keresőfa

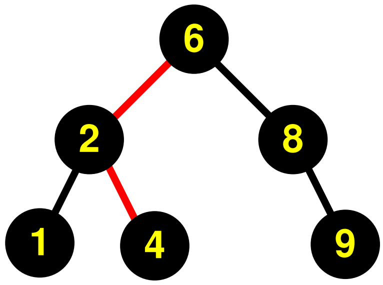
Definíció (Keresőfa-tulajdonság)

Tetszőleges x csúcsra és az x baloldali részfájában levő y csúcsra igaz, hogy elem(y) $\leq$ elem(x). Hasonlóan, ha z egy csúcs az x jobb részfájából, akkor elem(x) $\leq$ elem(z).



Házi feladat: Igazoljuk, hogy egy bináris keresőfa elemeit a fa inorder bejárása *nemcsökkenő sorrendben* látogatja meg.
Egy kényelmes megállapodás: a továbbiakban feltesszük, hogy nincsenek ismétlődő elemek a keresőfában.

Naiv algoritmusok



KERES$(4, S)$

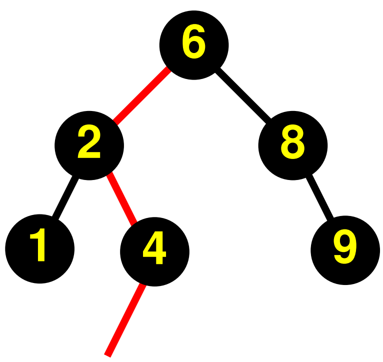
KERES(s,S): Összehasonlítjuk s-et S gyökerében tárolt s' elemmel.

- Ha $s = s'$, akkor megtaláltuk.
- Ha $s < s'$, akkor balra megyünk tovább.
- Ha $s > s'$, akkor jobbra megyünk.

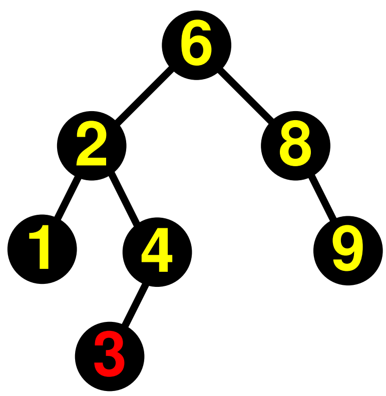
Ugyanezt az utat járjuk be a KERES$(5, S)$ kapcsán, de azt nem találjuk meg.
Lépésszám: $O(l)$, ahol l a fa mélysége
MIN: mindig balra lépünk, amíg lehet
MAX: mindig jobbra lépünk, amíg lehet
Lépésszám: $O(l)$
TÓLIG(a, b, S): KERES$(a, S) \longrightarrow$ INORDER a-tól b-ig
Lépésszám: $O(l + k)$, ahol k az a és b között levő elemek száma

Naiv BESZÚR

BESZÚR(s, S): KERES(s, S)-sel megkeressük, hova kerülne, és új levelet adunk hozzá, pl. BESZÚR$(3, S)$:



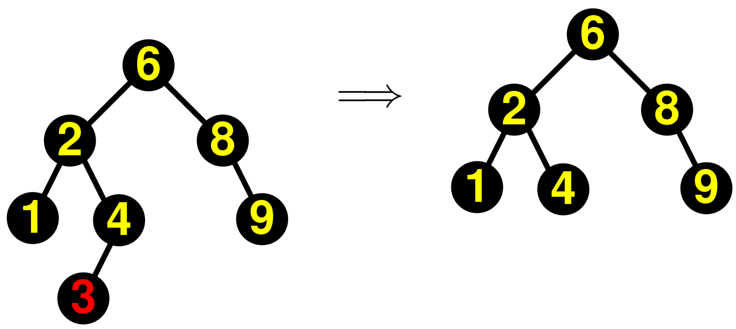
$\Longrightarrow$



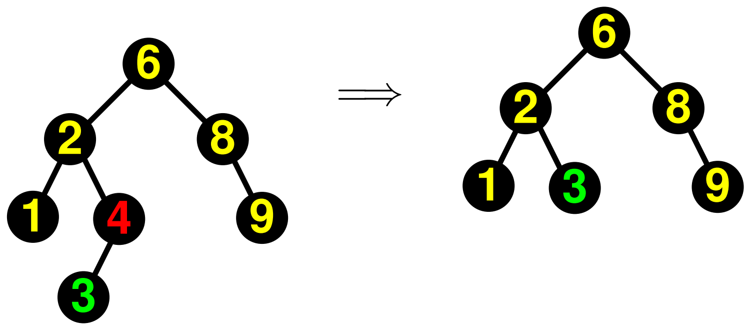
Lépésszám: $O(l)$

Naiv TÖRÖL

- TÖRÖL(s, S): Ha s levél, akkor triviális, pl. TÖRÖL$(3, S)$:

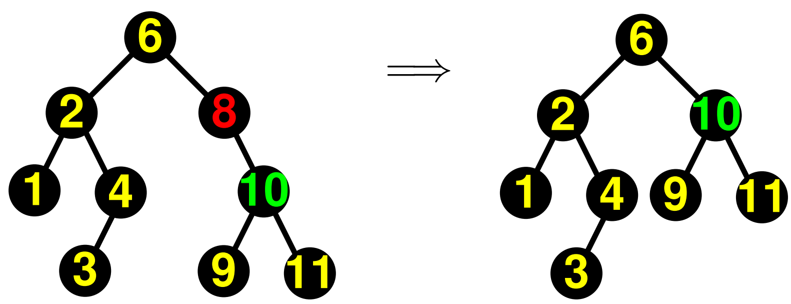


- TÖRÖL(s, S): Ha s-nek egy fia van, akkor: $s \leftarrow$ fiú(s), pl. TÖRÖL$(4, S)$:

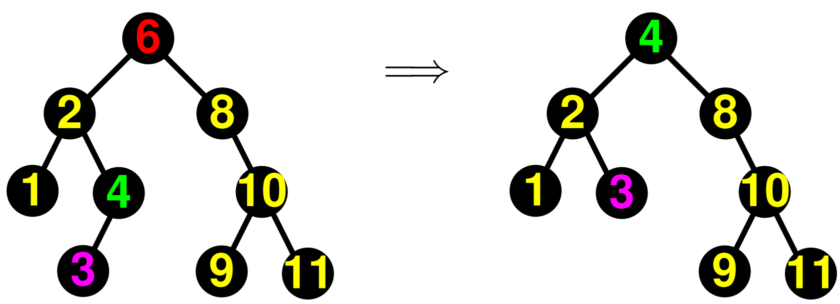


Naiv TÖRÖL

- Vagy pl. TÖRÖL$(8, S')$:



- TÖRÖL(s, S): Ha s-nek két fia van, akkor visszavezetjük az előző esetre. s helyére tegyük $y := \mathrm{MAX}(bal(s))$-t és töröljük y-t. Pl. TÖRÖL$(6, S')$:



Naiv TÖRÖL

Állítás

$y := MAX(bal(s))$ csúcsnak nem lehet két fia.

Bizonyítás.

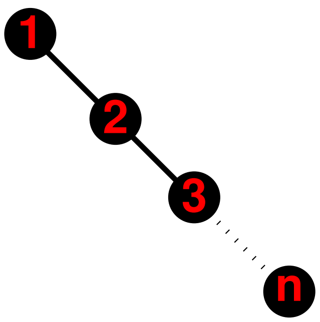
Ha lenne két fia, akkor lenne egy y' jobb fia is. De ekkor $y' > y$.

↯ $\square$

Lépésszám: $O(l)$

Faépítés naiv beszúrásokkal

Ha pl. az $1, 2, \ldots, n$ sorozatból építünk fát így, akkor ezt kapjuk:



Az építés költsége: $2 + 3 + \ldots + (n-1) = O(n^2)$

Tétel

Ha egy véletlen sorozatból építünk fát naiv beszúrásokkal, akkor az építés költsége átlagosan $O(n \log_2 n)$. A kapott fa mélysége átlagosan $O(\log_2 n)$.

Java animáció: Bejárások, bináris keresőfa

Piros-fekete fák

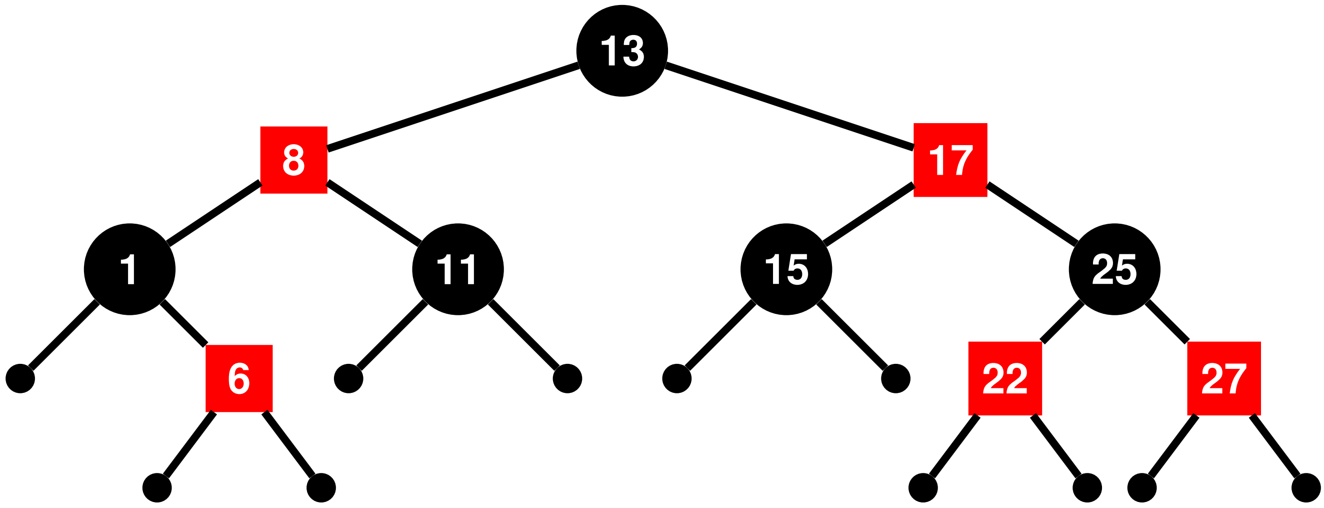
Olyan bináris keresőfa, melynek mélysége nem lehet nagy.
BESZÚR, TÖRÖL, KERES, MIN, (MAX, TÓLIG) hatékonyak.

Definíció

*A **piros-fekete fa** egy bináris keresőfa, melyre teljesülnek a következők:*

1. *Minden nem levél csúcsnak 2 fia van.*
2. *Elemeket belső csúcsokban tárolunk.*
3. *Teljesül a keresőfa tulajdonság.*
4. *A fa minden csúcsa piros vagy fekete.*
5. *A gyökér fekete.*
6. *A levelek feketék.*
7. *Minden piros csúcs mindkét gyereke fekete.*
8. *Minden v csúcsra igaz, hogy az összes v-ből levélbe vezető úton ugyanannyi fekete csúcs van.*

Példa



Megj.: A szokásos bináris fát kiegészítjük üres levelekkel.

Piros-fekete fák

Jelölések

- F_v: v gyökerű részfa
- $m(v)$: v magassága, a leghosszabb v-ből levélbe vezető út éleinek száma
- $fm(v)$: v fekete-magassága, a v-ből levélbe vezető összes úton a fekete csúcsok száma, v-t nem számolva.
  (Ez minden úton egyforma a 8. tulajdonság miatt.)

Tulajdonságok

Állítás

Egy piros-fekete fa minden v csúcsára teljesül

$$\frac{m(v)}{2} \leq fm(v) \leq m(v).$$

Bizonyítás.

A leghosszab levélbe vezető úton a feketék száma nem lehet több az élek számánál $\Longrightarrow fm(v) \leq m(v)$. $\surd$

7. pont miatt a leghosszabb úton a pontoknak legalább a fele fekete $\Longrightarrow \frac{m(v)}{2} \leq fm(v)$. $\surd$ $\square$

Tulajdonságok

Állítás

F_v belső csúcsainak száma $b_v \geq 2^{fm(v)} - 1$.

Bizonyítás.

Indukcióval $m(v)$-re: $m(v) = 0 \Longrightarrow fm(v) = 0$, $b_v \geq 2^0 - 1$ $\surd$

Ha $m(v) > 0$, akkor legyen x, y a két fia.

$\Longrightarrow m(x) < m(v)$ és $m(y) < m(v)$

$fm(v) - 1 \leq fm(x) \leq fm(v)$ és $fm(v) - 1 \leq fm(y) \leq fm(v)$

$b_v = b_x + b_y + 1 \Longrightarrow$

$b_v \geq (2^{fm(x)} - 1) + (2^{fm(y)} - 1) + 1 \geq 2 \cdot (2^{fm(v)-1} - 1) + 1 = 2^{fm(v)} - 1.$ $\square$

Tulajdonságok

Állítás

Ha egy piros-fekete fában n elemet tárolunk, akkor a fa magassága $\leq 2\log(n+1)$.

Bizonyítás.

Ha r a gyökér $\Longrightarrow b_r = n$.

$n = b_r \geq 2^{fm(r)} - 1 \Longrightarrow \log(n+1) \geq fm(r) \geq \frac{m(r)}{2}$ $\surd$ □

Tétel

KERES, MAX, MIN lépésszáma piros-fekete fában $O(\log n)$.

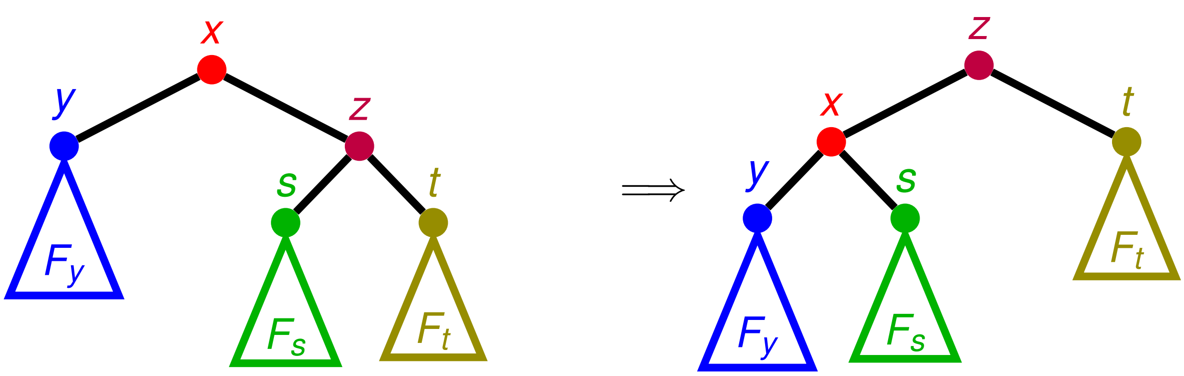
Bizonyítás.

Általában minden keresőfában a lépésszám a fa magasságával arányos $\Longrightarrow O(l) = O(\log n)$. □

BESZÚR lépésszáma

Ha a keresőfáknál használatos beszúrást használnánk, akkor megsérülhetne a piros-fekete tulajdonság.

Forgatás

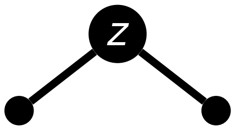


Megj.: Ez a művelet megtartja a keresőfa tulajdonságot.

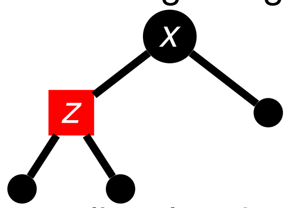
BESZÚR

Szúrjuk be az új elemet a keresőfáknál megismert módon. $\Longrightarrow$
Új belső csúcs keletkezik (gyerekei csak üres fekete levelek): z

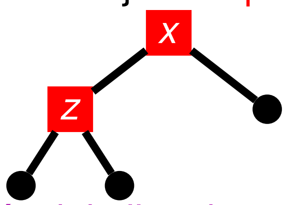
- Ha z a gyökér, akkor legyen fekete $\Longrightarrow$



- Ha z nem gyökér, akkor legyen az apja x, $\Longrightarrow$
  z legyen piros.
  (1) Ha x fekete $\Longrightarrow$ fekete-magasságok sehol nem változnak √ $\Longrightarrow$



  (2) Ha x piros $\Longrightarrow$ nem teljesül a piros-fekete tulajdonság $\Longrightarrow$

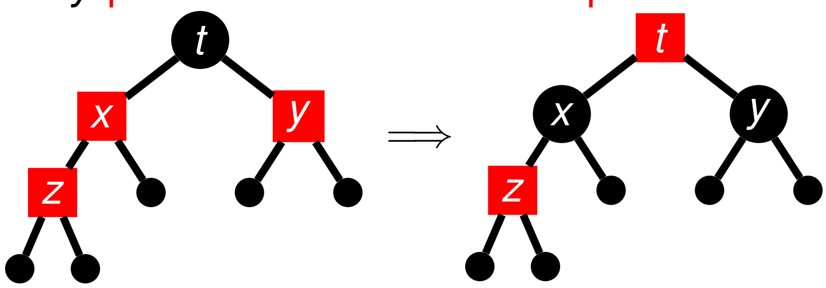


$\Longrightarrow$ további lépések kellenek.

BESZÚR

(2) Mivel x piros, nem gyökér $\Longrightarrow$
legyen x apja t (fekete), x testvére y.

(2.1) Ha y piros $\Longrightarrow$ átszínezzük t-t pirosra

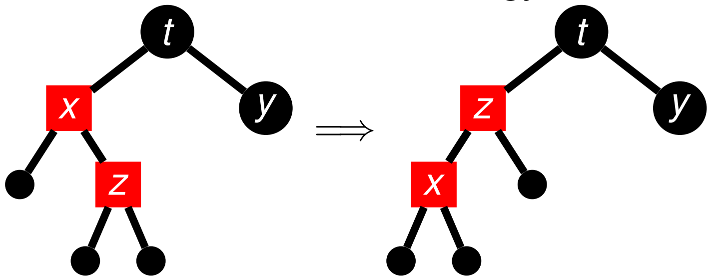


Evvel a problémát két szinttel feljebb toltuk, ott folytatjuk a fa rendbetételét.
Kivéve, ha t a gyökér $\Longrightarrow$ t marad fekete $\Longrightarrow$ $fm(t)$ eggyel nagyobb lesz.

BESZÚR

(2.2) Ha *y* fekete:

(2.2.1) Ha *z* és *x* nem azonos oldali gyerek $\Longrightarrow$ forgatunk *x* körül.

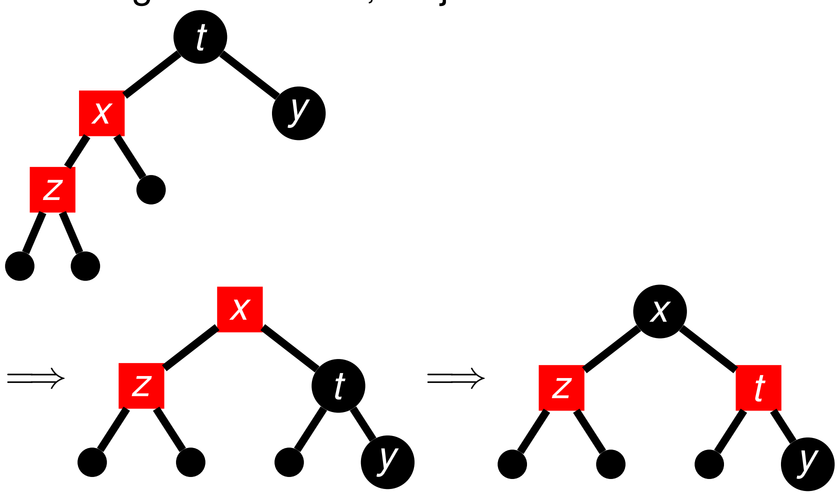


Evvel a következő esetre vezettük a problémát.

BESZÚR

(2.2) Ha *y* fekete:

(2.2.2) Ha *z* és *x* azonos oldali gyerek
$\Longrightarrow$ forgatunk *t* körül, majd átszínezünk.



Evvel a gyökér fekete-magassága nem változik, és teljesül a piros-fekete tulajdonság. $\surd$

BESZÚR

Tétel

A BESZÚR során

(a) *a lépésszám $O(\log n)$,*

(b) *legfeljebb 2 forgatás történik.*

Bizonyítás.

(a) y piros esetben a (2.1) pontban 2 szinttel feljebb kerül a baj $\Longrightarrow$ szintenként konstans lépés $\Longrightarrow O(\log n)$. $\surd$

(b) Forgatás csak a (2.2) esetben történik, de ekkor nincs felgyűrűzés, rögtön kijavítjuk a fát. $\surd$

$\square$

TÖRÖL

Hasonló módszerek, de bonyolultabb.

Tétel

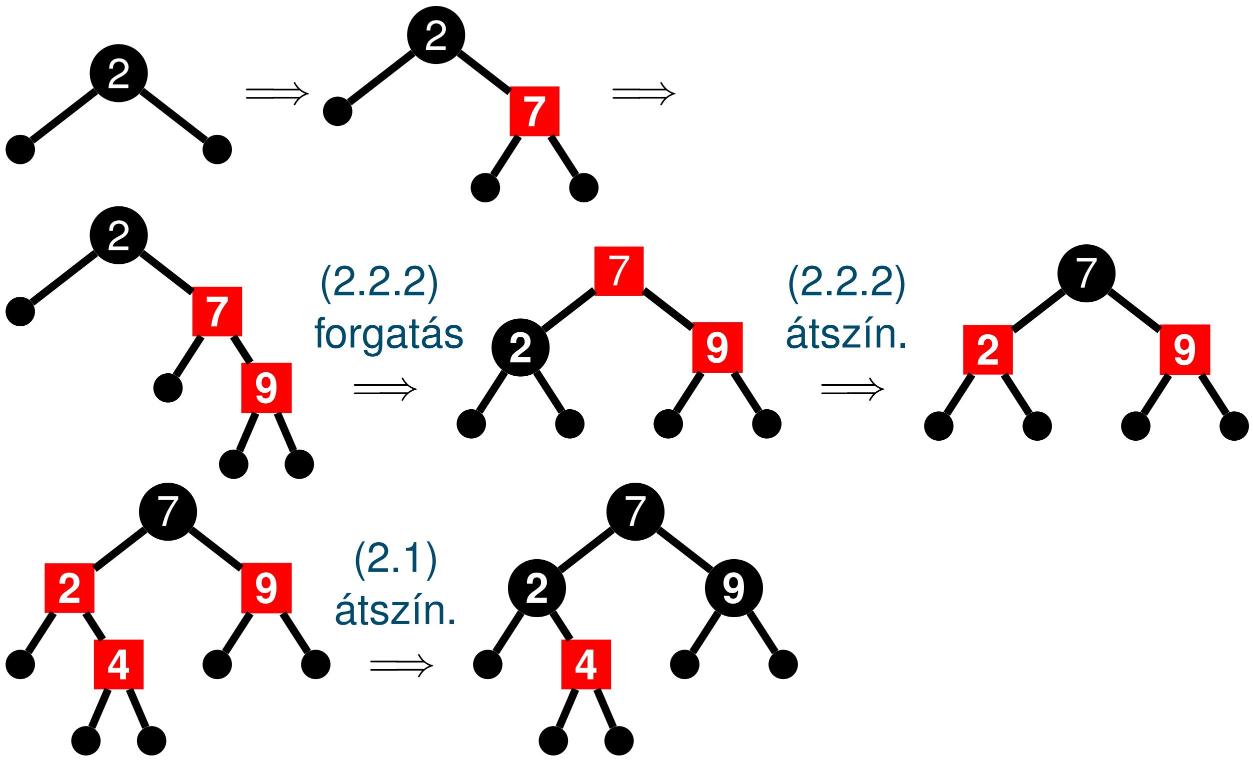
A TÖRÖL során

(a) *a lépésszám $O(\log n)$,*

(b) *legfeljebb 3 forgatás történik.*

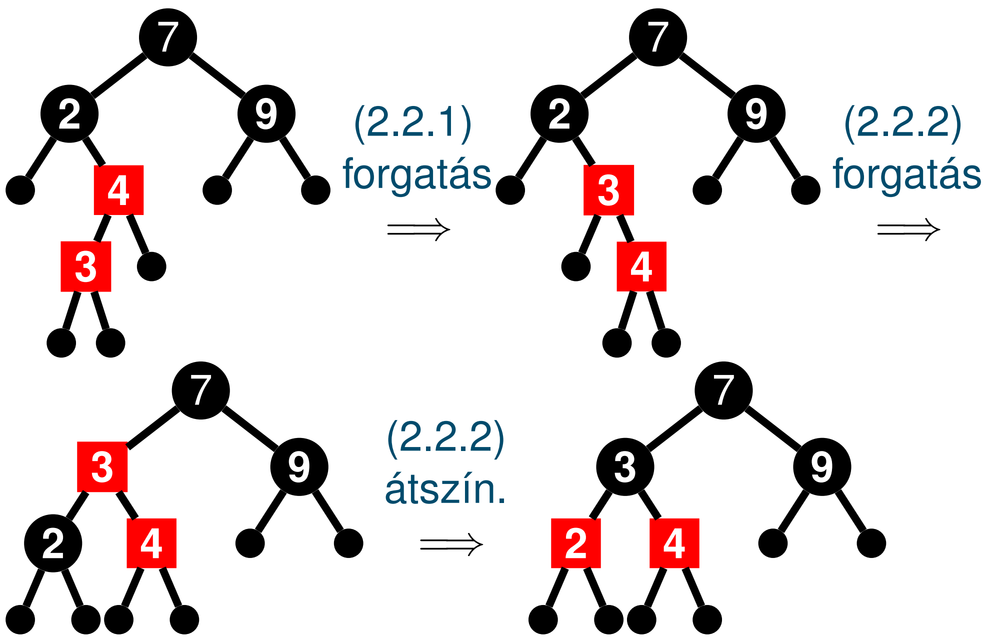
Példa BESZÚRásokra

Szúrjuk be egy üres fába sorban a $2, 7, 9, 4, 3, 1$ elemeket.



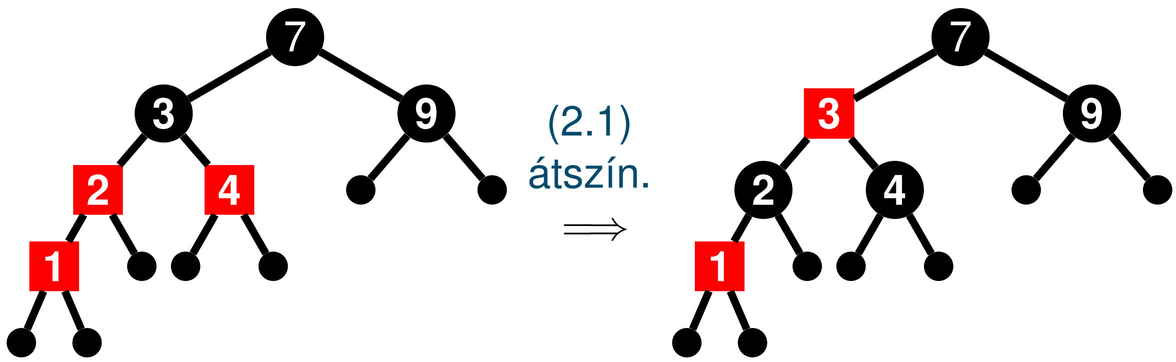
Példa BESZÚRásokra

Szúrjuk be egy üres fába sorban a $2, 7, 9, 4, 3, 1$ elemeket.



Példa BESZÚRásokra

Szúrjuk be egy üres fába sorban a $2, 7, 9, 4, 3, 1$ elemeket.



Java animáció: Piros-fekete fa