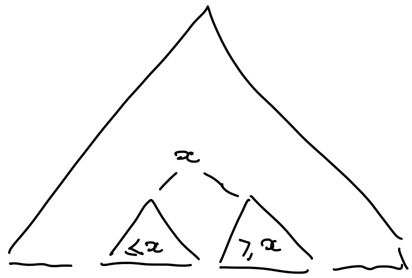
Algoritmi e Strutture Dati (09/11/2021)

* Alberi binari di ricerca

- → alberi binari ordinati
- → nodi x — $x.left$, $x.right$, $x.p$, $x.key$



Operazioni

→ InOrder (x) $\Theta(n)$

Search (x, k)
Min (x)
Max (x)
Succ (x)
Pred (x)
Insert (T, x)

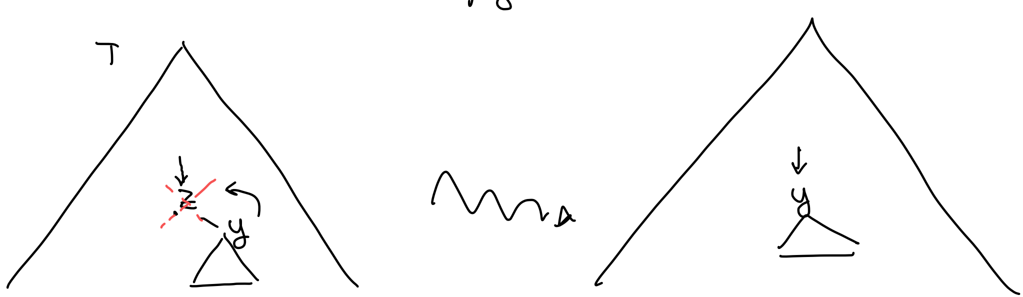
} $O(h)$ — h = altezza dell'albero

* Cancellazione

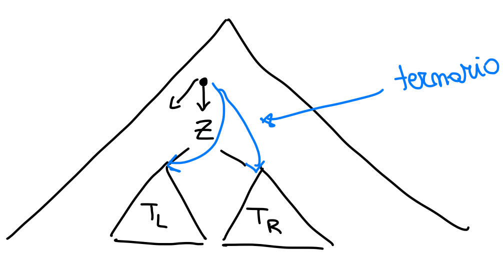
dato z in T e lo vogliamo eliminare

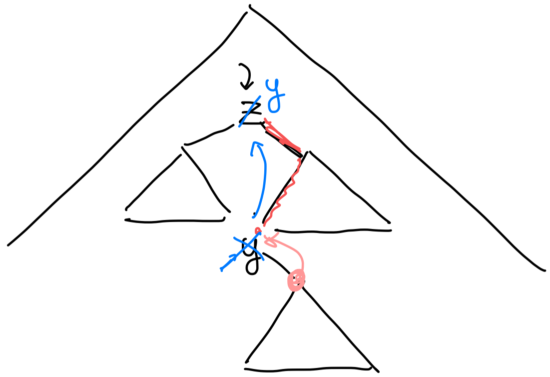
due possibilità

① z ha al massimo un figlio



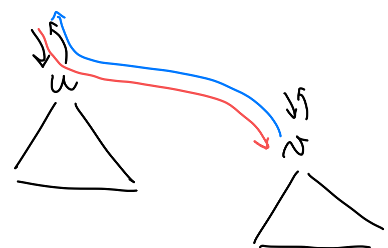
② z ha due figli





copia i dati di y in z : pericoloso, meglio spostare i nodi!

Transplant (T, u, v) // u ≠ nil




```
if u.p = nil
   T.root = v
else if u = u.p.left
        u.p.left = v
else
        u.p.right = v

if v ≠ nil
    v.p = u.p
```

$O(1)$

Delete (T, z) // z ≠ nil

```
if (z.left = nil)
   Transplant (T, z, z.right)
```

$O(1)$

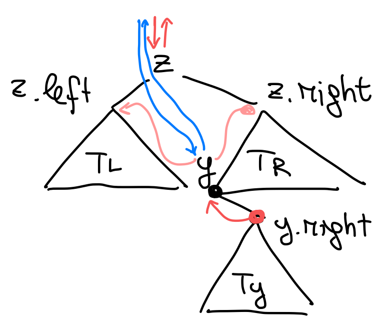
```
else if (z.right = nil)
   Transplant (T, z, z.left)
```

$O(1)$

```
else
```

$\boxed{O(h)}$ { y = Min(z.right)

```
   if (y.p ≠ z)
      Transplant (T, y, y.right)
      y.right = z.right
      y.right.p = y
   y.left = z.left
   y.left.p = y
```

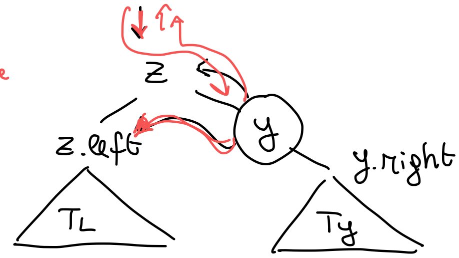


$\boxed{\text{Transplant}(T, z, y)}$

else

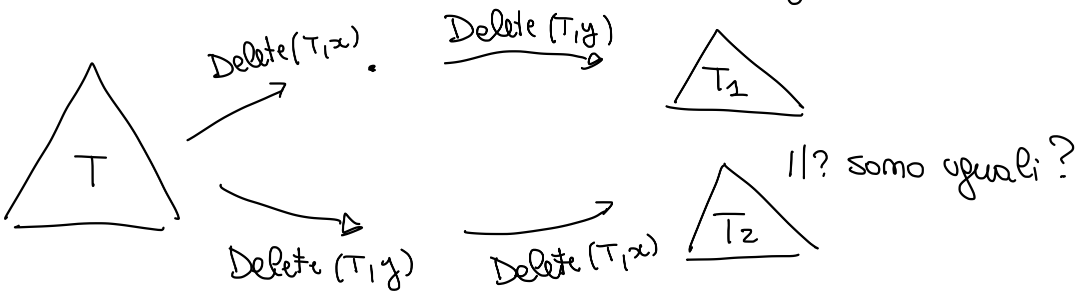
$$\boxed{\begin{array}{l} y.left = z.left \\ y.left.p = y \\ \text{Transplant}(T, z, y) \end{array}}$$

fattorizzare



Complessità : $O(h)$

ESERCIZIO : Dato BST T e due nodi x, y (diversi)



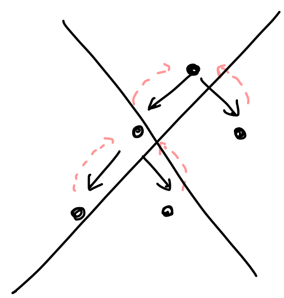
ESERCIZIO : Alberi binari di ricerca

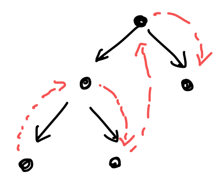
$x.left$

$x.right$

~~$x.p$~~

$x.succ$

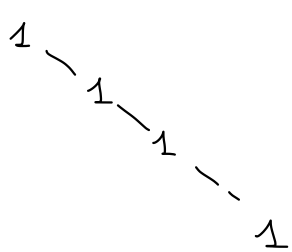




re-implementare le varie operazioni in tempo $O(h)$?
(forse)

complessità $O(h)$

Insert$(T, 1)$ Insert$(T, 1)$ -- Insert$(T, 1)$



* <u>Red Black Trees</u>

sono ABR in cui i nodi hanno un campo

x. color < RED / BLACK

si utilizza T.nil nodo a tutti gli effetti con i vari campi

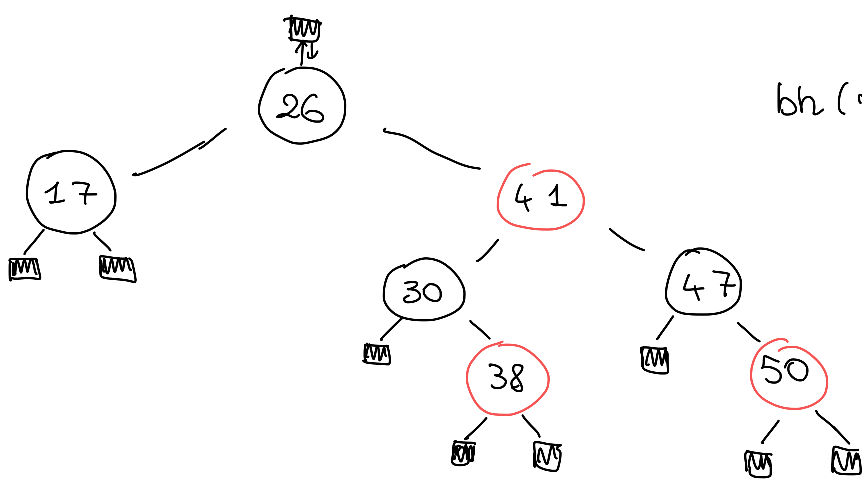
T.nil.color = BLACK

<u>Def</u>: Un RB-tree è un ABR

1. ogni nodo ha uno e un solo colore BLACK / RED
2. root è BLACK
3. foglie (T.nil) sono BLACK
4. i figli di un nodo RED sono BLACK
5. per ogni nodo x

per ogni cammino x ⇝ foglia il numero di nodi BLACK è sempre lo stesso

↳ bh(x) = numero di BLACK in un cammino x ⇝ foglie (escluso x)



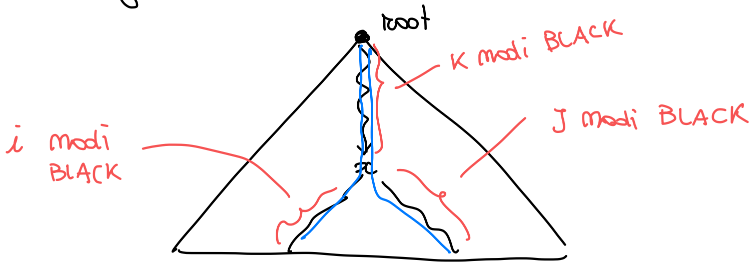
bh(root) = 2

<u>NOTA</u>

proprietà (5) vale
sse
ogni cammino dalla radice ha lo stesso numero di nodi BLACK

(⇒) ovvio

($\Leftarrow$) sia x generico nodo



$\cancel{k}+i = \cancel{k}+j$

$i = j$

OSSERVAZIONI :

① se eliminiamo i nodi RED otteniamo un albero nel quale (come in un albero completo perfettamente bilanciato) ogni cammino nodo → foglia ha la stessa lunghezza

② i nodi BLACK sono almeno la metà dei nodi (in ogni cammino)

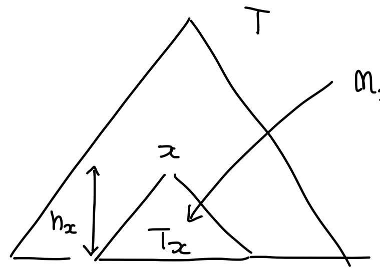
OSSERVAZIONE : Sia T RB-tree con n = # nodi interni (non foglie T.nil)

h = altezza

$$h \leq 2 \log(n+1)$$

dim

dato un qualunque nodo x



n_x = num. nodi interni T_x

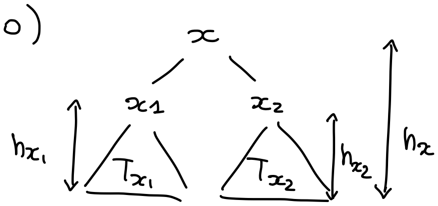
$$n_x \geq 2^{bh(x)} - 1$$

per induzione su h_x

($h_x = 0$) x è foglia

$$n_x = 0 \geq 2^{bh(x)} - 1 = 2^0 - 1 = 0$$

($h_x > 0$)



$h_{x_1}, h_{x_2} < h_x$

$n_{x_1} \geq 2^{bh(x_1)} - 1$

$n_{x_2} \geq 2^{bh(x_2)} - 1$

$$n_x = n_{x_1} + n_{x_2} + 1$$

$$\geq (2^{bh(x_1)} - 1) + (2^{bh(x_2)} - 1) + 1$$

$$= 2^{bh(x_1)} + 2^{bh(x_2)} - 1 \qquad bh(x_1) \geq bh(x) - 1$$

$$\geq 2^{bh(x)-1} + 2^{bh(x)-1} - 1 \qquad bh(x_2) \geq bh(x) - 1$$

$$= 2 \cdot 2^{bh(x)-1} - 1$$

$$= 2^{bh(x)} - 1$$

per ogni x

$$n_x \geq 2^{bh(x)} - 1$$

per la radice

$$n = n_{root} \geq 2^{bh(root)} - 1 \qquad bh(root) \geq \frac{h}{2}$$

$$\geq 2^{h/2} - 1$$

$$2^{h/2} \leq n+1$$

$$h/2 \leq \log_2(n+1)$$

$$h \leq 2 \log_2(n+1)$$

OPERAZIONI

Search
Max
Min
Succ
Prec

$$O(h) = O(\log n)$$

DIFFICILE MANTENERE LA COLORAZIONE per Insert / Delete!

Insert (T, z)