

1. Worst case: $O(n)$

2. ~~Node~~ ^{int} min-node (Node root) {
 Node curr = root;
 while (curr -> left != NULL) {
 curr = curr -> left;
 }

 return curr -> key;
}

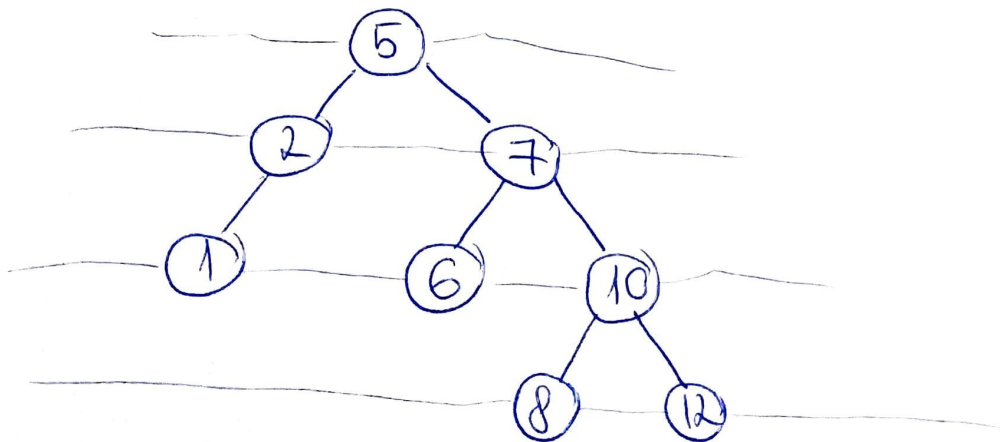
3. left subtree: 300
 right subtree: 400

preorder: left - root - right \Rightarrow 300

inorder: root - left - right \Rightarrow 0

postorder: left - right - root \Rightarrow 700

5.



max height: 3 (if h starts from 0)

4 (if h starts from 1)

depending on implementation

6. Before

*

Insert "rocketed":

* (0)
↓
r (0)
↓
c (0)
↓
c (0)
↓
k (0)
↓
e (0)
↓
t (0)
↓
e (0)
↓
d (1)

OR

*
↓
↓
↓
o
↓
c
↓
k
↓
e
↓
t
↓
e
↓
d
↓
*

Insert "rock":

* (0)
↓
r (0)
↓
c (0)
↓
c (0)
↓
k (1)
↓
e (0)
↓
t (0)
↓
e (0)
↓
d (1)

OR

*
↓
↓
↓
o
↓
c
↓
k
↓
e
↓
t
↓
e
↓
d
↓
*

Insert "rocket":

* (0)
↓
r (0)
↓
o (0)
↓
c (0)
↓
k (1)
↓
e (0)
↓
t (1)
↓
e (0)
↓
d (1)

OR

*
↓
o
↓
c
↓
k
↓
e
↓
t
↓
e
↓
d
↓
*


*
↓
e
↓
t
↓
e
↓
d
↓
*


- depending on implementation of stop

7. Insert 9:  $t=3 \Rightarrow 2 \leq k \leq 5$

Insert 8: 

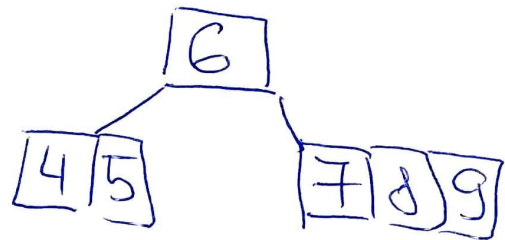
Insert 7: 

Insert 6: 

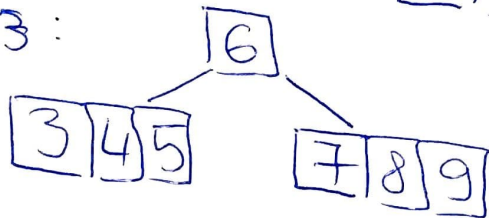
Insert 5: 

Insert 4: before split: 

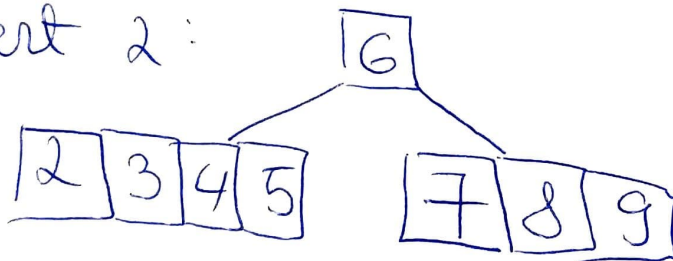
after split:



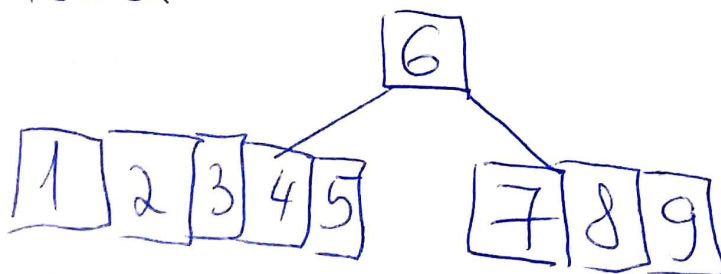
Insert 3:



Insert 2:

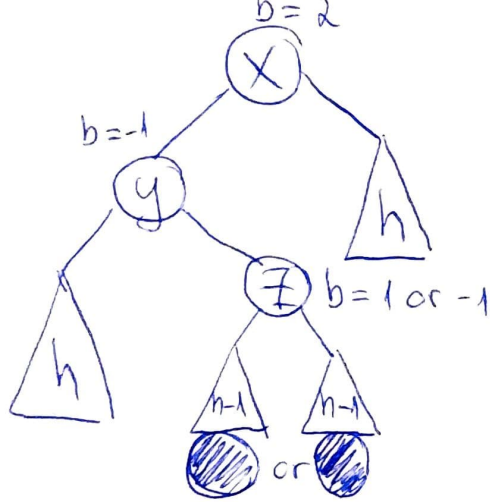


Insert 1:

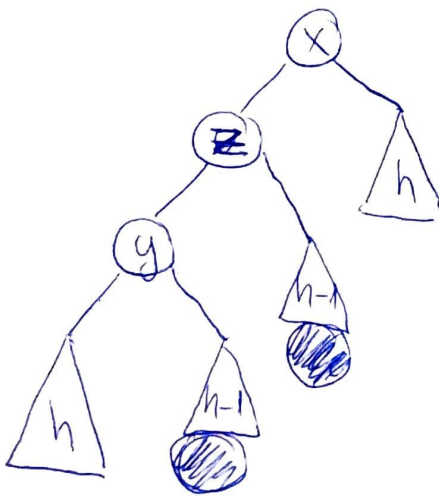


4.

LEFT-RIGHT DOUBLE ROTATION



after left rotation (y and z)



after right rotation (x and z)

