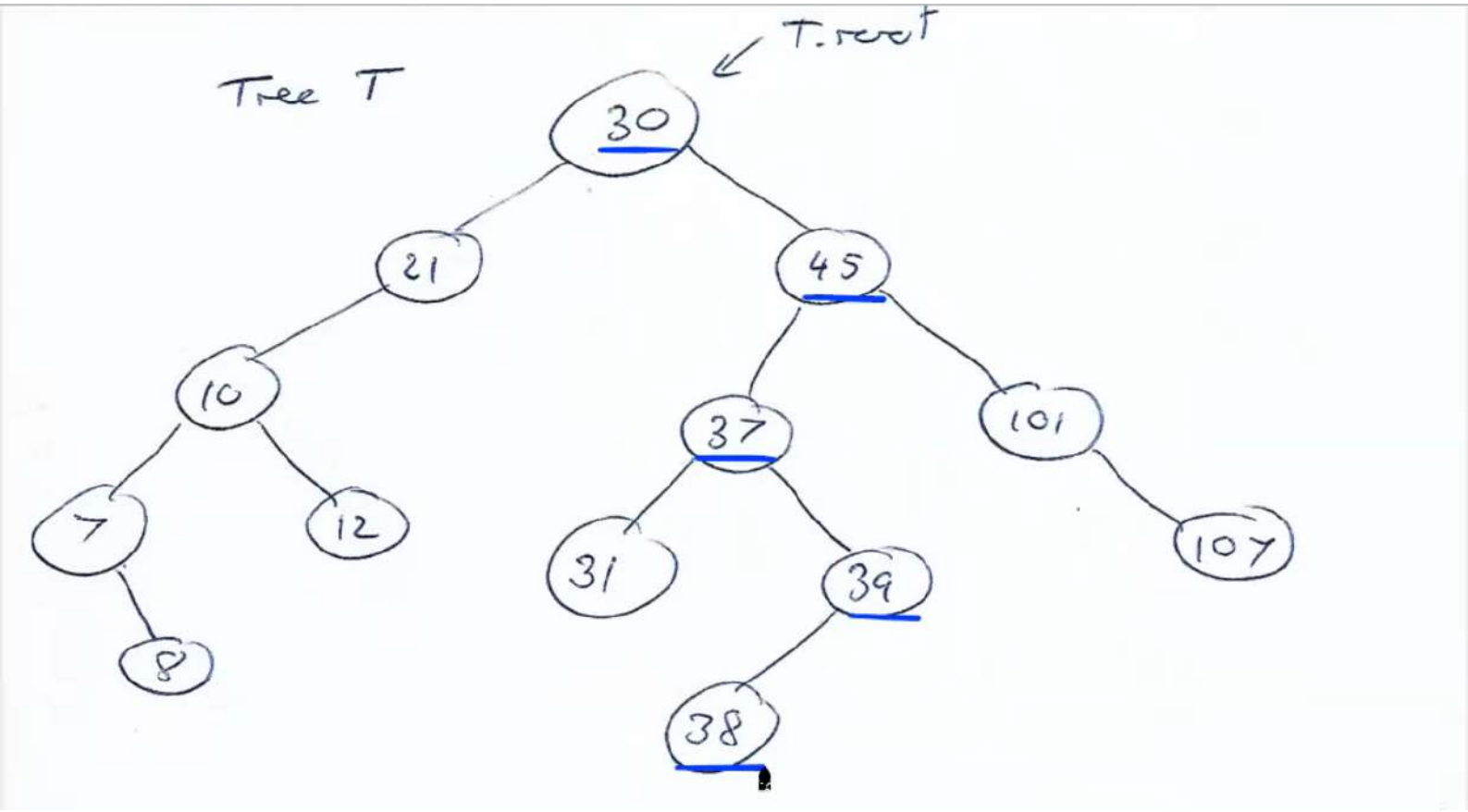
12.2 Querying a binary search tree

```
Tree-Search(x, k)
  if x = NIL or k = x.key
      return x
  if k < x.key
      return Tree-Search(x.left, k)
  else return Tree-Search(x.right, k)
```


Tree T
T.root
30
21
45
10
37
101
7
12
31
39
107
8
38

run-time of Tree-Search is $O(h)$

h is height of tree - the length of the longest simple path from the root to a leaf.

run-time of Tree-Maximum and Tree-Minimum is $O(h)$

Tree-Maximum is similar.

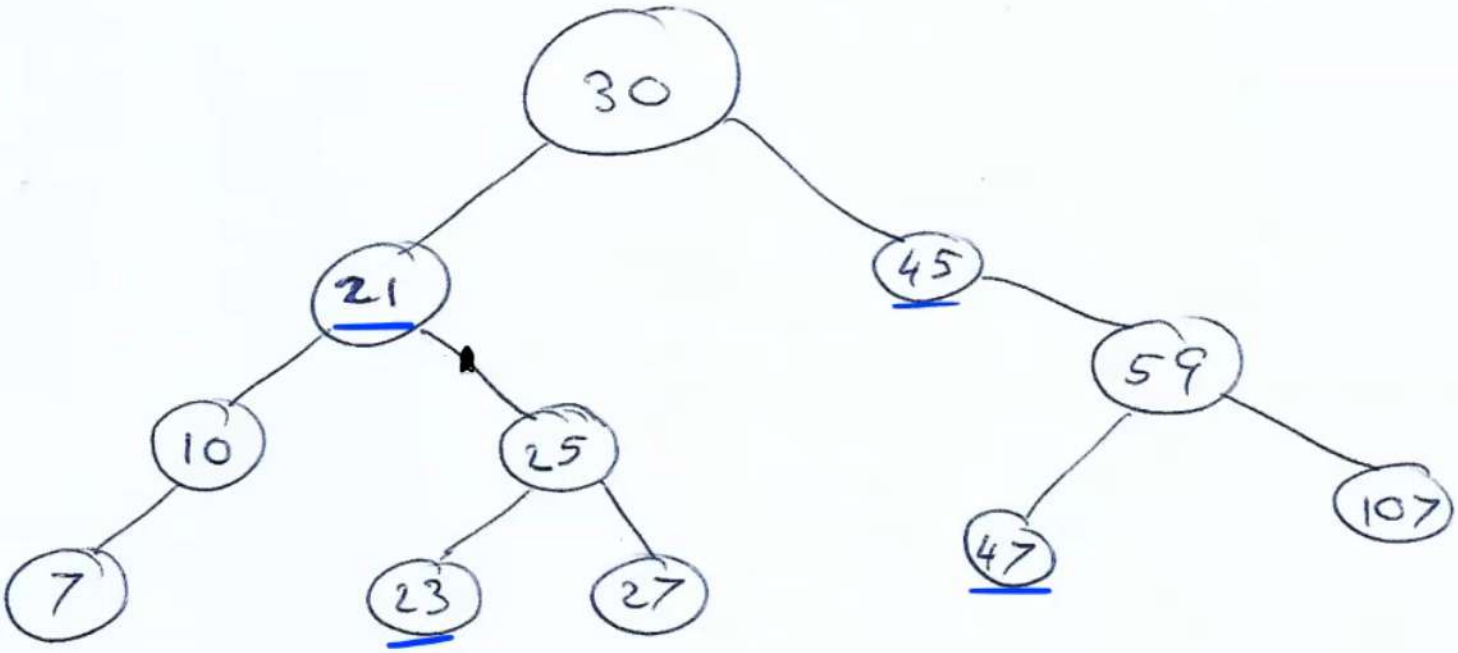
run-time of Tree-Search is $O(h)$

h is height of tree - the length of the longest simple path from the root to a leaf.

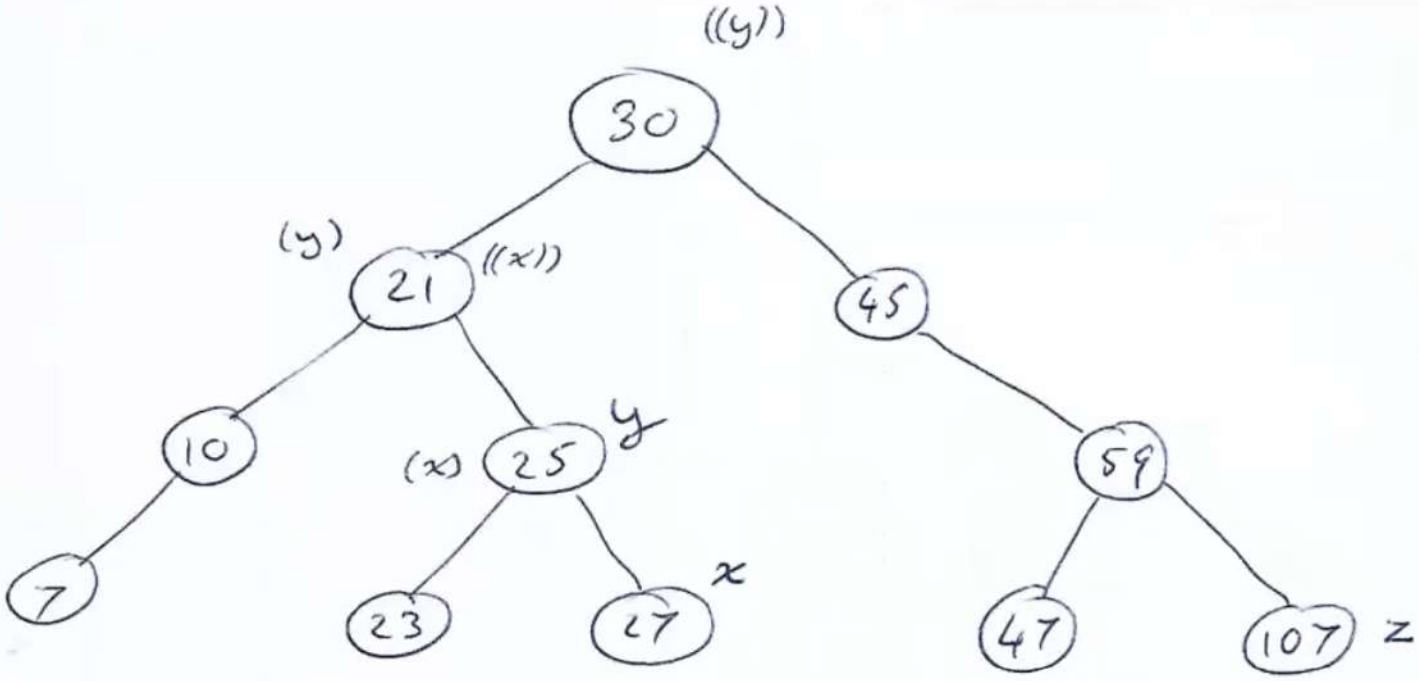
run-time of Tree-Maximum is $O(h)$

Tree-Successor(x)

return the node with the successor key to x.key (ie, next biggest key).


30
21
45
10
25
59
7
23
27
47
107

```
TREE-SUCCESSOR(x)
  if x.right ≠ NIL
      return TREE-MINIMUM(x.right)
  y = x.p
  while y ≠ NIL and x = y.right
      x = y
      y = y.p
  return y
```



Tree-Successor(x) returns node with key = 30.

Tree-Successors(z) returns NIL

Tree-Predecessor is similar.

Thm: Tree-Predecessor & Tree-Successor run in $O(h)$