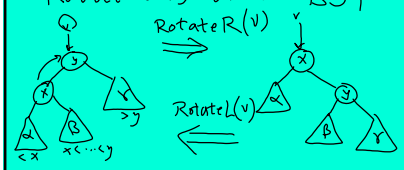
Treaps (cont.)

Rotations of a BST




```
TreapInsert(T, x)
// insert item x into treap T.
1. Usual BST insertion of x
     into T
2. Going up a path from x
   to the root:
     for each v along the path:
       if v.priority > v.parent.priority
         Rotate(v.parent)
           // left or right
           // rotate as appropriate
       else stop.
```

Time is $O(d)$ worst case, where d is the depth of the treap.

Strategy: We insert keys into a treap — for each insertion, assign the key a random priority (e.g., a uniformly chosen real number in $[0,1]$)

Random priorities force the treap to look like a BST subject to randomized insertion order:

[Inserting keys in order of decreasing priority causes no rotations, resulting in a treap identical to the BST subject to this insertion order

- Treap shape does not depend on insertion order.]

∴ Treap with random priorities has same analysis as a standard BST with random order of insertion.

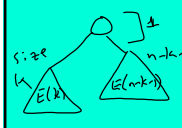
E(depth of treap w/ rand priorities)

$= E$(depth BST with uniformly random insertion order)

$=: \underset{\text{depth}}{E(n)}$ $\quad n$ = treap size = BST

Recurrence?

$E(n) \overset{?}{=} 1 + \frac{1}{n}\sum_{k=0}^{n-1} \max(E(k), E(n-k-1))$



Not quite right because $E(\max(\cdot,\cdot)) \neq \max(E(\cdot), E(\cdot))$

Analysis is in CLRS: get $E(n) = \Theta(\lg n)$.