~~add (1, 15)~~ HW #4 * Treap *

* we Know there could be more than one BST for a given nodes. depends on the order of insertion

* worst case, when key is sorted (increasing or decreasing)
↳ ~~list~~ BST degenerate into list
order of insertion affect the height of the tree.

⇒ Qustion: How do we minimize the height of BST
① we could change the order of insertions.

② simple shuffle in the list of key is sufficint to make the tree balanced.

ex: 1 2 3 4 5 6 7 8 9 10
7 6 8 2 1 9 4 10 5 3
or 4 8 2 3 6 9 5 7 1 10

Intersting BUT useless because we have to read all insertion first.

Solution: Treap.
Data structure help to shuffle the key after each insertion.

idea: insert key in any order, give each key random Priorty → make sure to shuffle in "real time" after each insertion.

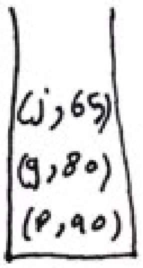
~~Ex~~ Add operation. add (i , 93)

$\boxed{1}$ Ignore priorty and insert item ~~[illegible]~~ like simple BST

hint (use iterative rather than recursion)

$\boxed{2}$ While Tracing the path from root to find appropriate Place for the new item.

insert nodes along the path in a stack.



$\boxed{3}$ use reheap function to compare the priority of new inserted item with all of parents if there is violation of max heap property $\rightarrow$ reheap. (same with insert into heap.) but we can't do simple swap because it will violate BST property

$\Rightarrow$ The Tow simple operation that allows us to modify Tree and Keep the BST property is left and right rotations.

P.S. This the operations are comon in self_balancing BST. such as AVL and red-black. Trees.

if the node with the heighest priority is

① left child $\rightarrow$ rotate right

② right child $\rightarrow$ rotate left.