Dictionary implementation using Binary Search Tree ADT as data source

Command	Process	# of Nodes in Tree	# of Nodes Visited	Log(n) – Base 2
INSERT 4	Root is none, Insert 4 at root	1	1	Log (1) = 1
INSERT 3	Root is 4, left child of 4 is none, insert 3 at left child of 4	2	1	Log (2) = 1
INSERT 7	Root is 4, right child of 4 is none, insert 7 at right child of 4	3	1	Log (3) = 1
INSERT 1	Root is 4, left child of 4 is 3, left child of 3 is none, insert 1 at left child of 3	4	2	Log (4) = 2
INSERT 7	Root is 4, right child is 7, 7 is already in tree	4	2	Log (4) = 2

Optimal speed is O(log2n) which can be observed by the pattern you see here on the right-most column. The speed is rounded down since we are using integers.

In this implementation of the dictionary, we just use the dictionary as a mask for our Binary Search Tree, calling all the functions associated with the Binary Search Tree, which explains the one line methods for the Dictionary.