CSC148: DELETION IN BST

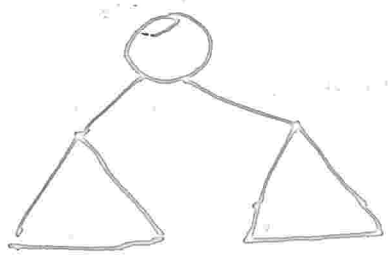
20161114

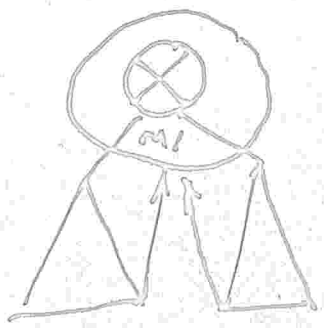
```
DEF DELETE(SELF, ITEM)
    """ REMOVE ONE OCCURRENCE OF AN ITEM FROM THE TREE """
    IF SELF.IS_EMPTY:
        PASS
    ELIF SELF._ROOT == ITEM:
        SELF.DELETE_ROOT()
    ELIF ITEM < SELF._ROOT:
        SELF._LEFT.DELETE(ITEM)
    ELSE:
        SELF._RIGHT.DELETE(ITEM)

DEF DELETE_ROOT(SELF):
```

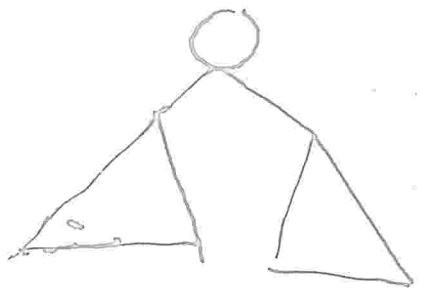
1

2





BST EFFICIENCY



- EACH RECURSIVE CALL GOES DOWN BY ONE, WHICH MEANS THAT WHAT MATTERS IN DETERMINING COMPLEXITY IS THE HEIGHT OF THE TREE.

THE LAST RECURSIVE CALL IS ON THE LEAF NODE IN THE LAST.

OF RECURSIVE CALLS = TREE HEIGHT (h+1)

∴ THE RUNNING TIME OF DELETE IS PROPORTIONAL TO O(h).

HOWEVER, THIS IS NOT A FUN FACT.

WHAT ABOUT SEARCH?

- H is the worst case scenario.