CSC148 winter 2015

mutating BSTs

week 9

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~heap/148/W14/
416-978-5899

March 9, 2015

Outline

test coverage

Location on course web page, under March 11. Last year's test was later in the term, had more coverage. Our test #2 covers weeks 5–8:

- Tree definitions, inorder, preorder, postorder traversals
- recursive functions on class Tree
- binary trees and binary search trees, recursive functions on class BTNode
- linked lists, class LLNode and LinkedList, iterative functions and mutation

You are allowed one 8.5" by 11" aid sheet.

recall BTNode

```
class BTNode:
    '''Binary Tree node.'''

    def __init__(self, data, left=None, right=None):
        ''' (BTNode, object, BTNode, BTNode) -> NoneType

        Create BTNode (self) with data
        and children left and right.
        '''
        self.data, self.left, self.right = data, left, right
```

insert must obey BST condition

example shows that we expect **insert** to ensure this is a binary **search** tree:

```
    def insert(self: 'BST', data: object) -> None:
        '''Insert data, if necessary, into this tree.

        >>> b = BST()
        >>> b.insert(8)
        >>> b.insert(4)
        >>> b.insert(2)
        >>> b.insert(6)
        >>> b.insert(12)
        >>> b.insert(14)
        >>> b.insert(10)
        >>> print(b)
                14
            12
                10
        8
                6
            4
                2
        <BLANKLINE>
```

insert function

```
def insert(node, data):
    ''' (BTNode, object) -> BTNode

    Insert data starting at node, and return root.

    '''
    return_node = node
    if not node:
        return_node = BTNode(data)
    elif data < node.data:
        node.left = insert(node.left, data)
    elif data > node.data:
        node.right = insert(node.right, data)
    else:  # nothing to do
        pass
    return return_node
```

deletion of data from BST rooted at node?

- what return value?
- what to do if node is None?
- what if data to delete is less than that at node?
- what if it's more?
- what if the data equals this node's data and...
  - this node has no left child
  - ... no right child?
  - both children?