Module 6a: Binary Search Trees

CPSC 110

Peyton Seigo

2018-MM-DD

Module 6a: Binary Search Trees

Learning goals

- Be able to reason informally about the time required to search data.
- Be able to identify problem domain information that should be represented using binary search trees.
- Be able to check whether a given tree conforms to the binary search tree invariants.
- Be able to use the design recipes to design with binary search trees.

Notes

Linear search:

- Best case: `1` operation, constant O(1)
- Worst case: `n` operations, linear O(n)
- Average case: `n/2` operations, linear O(n)

Binary Search Trees:

- every node has 0, 1, or 2 children
- ALL nodes in left sub-tree have a value less than the parent
- ALL nodes in right sub-tree have a value greater than the parent
  - this rule is invariant; holds true ALL the way down a branch!

Terminology

- x

Questions

In terms of time complexity and how things work under the hood, does Racket implement lists similar to linked lists? i.e. is access O(1) or O(n)?