Homework #6 - Red Black Trees

Note: Follow my naming exactly!

Write a red black tree class to hold any one data type using generics.

Function signatures / Variables - Swift:

RedBlackTree<T : Comparable>
var height : Int
var isEmpty : Bool
var size : Int
var elements : [T] // in order
init () // make an empty tree
func insert(element : T)
func contains(element : T) -> Bool
func search(element : T) -> T? //return the stored element if you find it, nil if you don't
func makeBreadthFirstArray() -> [T] //Top to bottom, left to right
Optional
init (fromSortedData : [T])
func delete(element : T)

Function signatures / Variables - C++:

RedBlackTree
RedBlackTree() // makes an empty one
int getHeight() const
bool isEmpty() const
int getSize() const
std::vector<T> elementVector() const
void insert(const T&)
bool contains(const T&) const

RedBlackTree
T search(const T&) const
std::vector<T> makeBreadthFirstVector() const //Top to bottom, left to right
Optional
BinarySearchTree(const std::vector<T>&) //builds tree from sorted array of ints
void delete(const T&)

Optional!

Turn your tree into a dictionary.

KeyValuePair<K : Comparable, V> : Comparable
var key : K
var value : V?
init(key: K, value: V?)
static func == (lhs: KeyValuePair<K, V>, rhs: KeyValuePair<K, V>) -> Bool
static func < (lhs: KeyValuePair<K, V>, rhs: KeyValuePair<K, V>) -> Bool

Dictionary<K : Comparable, V>
var tree : RedBlackTree<KeyValuePair<K, V>>
var size : Int
var isEmpty : Bool
var keys : [K]
func insert(key: K, value: V?)
func findValue(key : K) -> V?
func contains(key : K)
func delete(key: K) //only if you managed to make a delete function in the tree!