```go
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

package main

// O(nlog(n)) time | O(n) space - where n is the length of the array
func MinHeightBST(array []int) *BST {
  return constructMinHeightBst(array, nil, 0, len(array)-1)
}

func constructMinHeightBst(array []int, bst *BST, startIdx, endIdx int) *BST {
  if endIdx < startIdx {
    return nil
  }
  midIdx := (startIdx + endIdx) / 2
  valueToAdd := array[midIdx]
  if bst == nil {
    bst = &BST{Value: valueToAdd}
  } else {
    bst.Insert(valueToAdd)
  }
  constructMinHeightBst(array, bst, startIdx, midIdx-1)
  constructMinHeightBst(array, bst, midIdx+1, endIdx)
  return bst
}

type BST struct {
  Value int

  Left  *BST
  Right *BST
}

func (tree *BST) Insert(value int) *BST {
  if value < tree.Value {
    if tree.Left == nil {
      tree.Left = &BST{Value: value}
    } else {
      tree.Left.Insert(value)
    }
  } else {
    if tree.Right == nil {
      tree.Right = &BST{Value: value}
    } else {
      tree.Right.Insert(value)
    }
  }
  return tree
}

```