UNIVERSITÀ DI PISA

Dipartimento di Informatica
Corso di Laurea in Informatica

Assignment 01

Competitive Programming and Contests

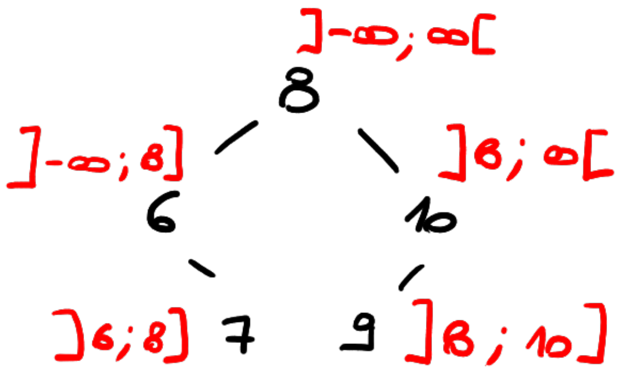
Prof. Rossano Venturini

Giacomo Trapani - 600124

Academic Year 2024/2025

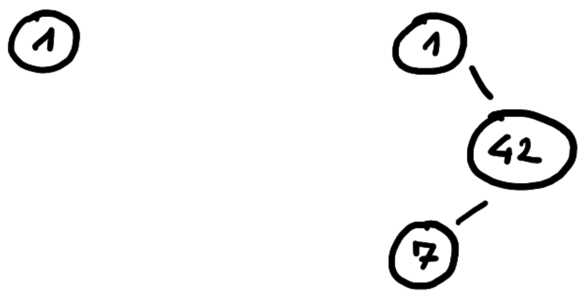
Exercise 1 This exercise requires the implementation of a method to check if a given tree is a **binary search tree**, which means that for each node its key K must be greater than that of its left child and less than that of its right child. For the purposes of this exercise, a relaxation of this condition is chosen: duplicates are allowed on one side of the binary search tree.

The idea behind the algorithm implemented is to work with lower and upper bounds. For each node, we check whether its value lies between the bounds, the values of which depend on its parent (node).



The algorithm implemented takes O(n) time.

Exercise 2 This exercise requires the implementation of a method to calculate the maximum path sum between two leaves in a binary tree. For the purposes of this exercise, the value `-i32::MIN` has been chosen to be returned whenever we cannot find a path between two leaves; also, the implementation of the `Tree` struct has been changed to work with `i32` values as the previous implementation would not have allowed for negative values.



The algorithm implemented measures two different things (ref. to `rec_maximum_path_sum`):

- the return value is the maximum path sum for a root-to-leaf path i.e. the maximum path sum from `index` to any of the leaves below,
- the value inside the pointer `global_max` is the maximum path sum for a leaf-to-leaf path i.e. the maximum path sum in the tree rooted in `index` between two of the leaves below it.

  The algorithm implemented takes O(n) time.