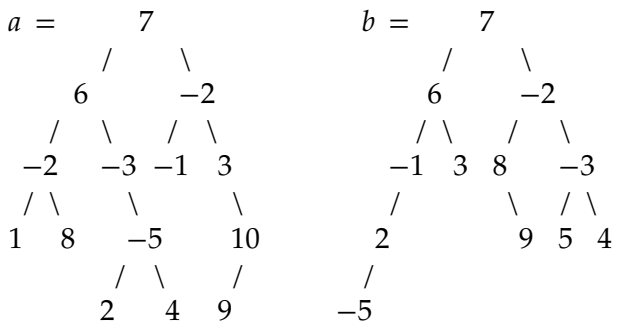
Preferential path in a tree (BinTree) X78332_en

Remember that the *leaf* of a tree is a node without any children. A *path* within a tree is a succession of nodes from the root to a leaf.
Given a binary tree **a** of elements of any kind, we define the *preferential path* in a tree **a** as follows: if **a** is empty the preferential path of **a** is also empty; in any other case, the preferential path of **a** is composed by the root of **a** followed by the preferential path of the child of **a** with more elements. If **a** has two non-empty children with the same number of elements, the preferential path of the left child is chosen.
We want a function that allows us to know which is the preferential path of a tree of integers, representing this path as a stack of integers, sorted in a way that the first element of the path (if it exists) is in the top of the stack. Trave the tree only once. Avoid copies and assignments of stacks and keep the use of additional space to a minimum. Use the following specification:

void *cami_preferent* (**const** *BinTree*<**int**>& *a*, *stack*<**int**>& *c*)
/* Pre: c is empty */
/* Post: c contains the preferential path of a; if it is non−empty, the
first element of the path is on the top of c */

Example: consider the following two trees



- the preferential path of **a** is 7 6 -3 -5 2.
- the preferential path of **b** is 7 -2 -3 5.

Input

The input is a tree of integers.

Output

The output is a stack with the preferential path. The root of the tree is on the top of the stack.

Observation

You only have to submit a file that contains the function with the header in the statement as well as any other auxiliary function that you think necessary, without any function main. Also add the include of the classes BinTree and stack by using
#include "BinTree.hh"
#include <stack>

Problem information

Author: Alberto Moreno (adapter), Borja Valles (responsible)

Generation: 2026-01-25T21:27:35.262Z