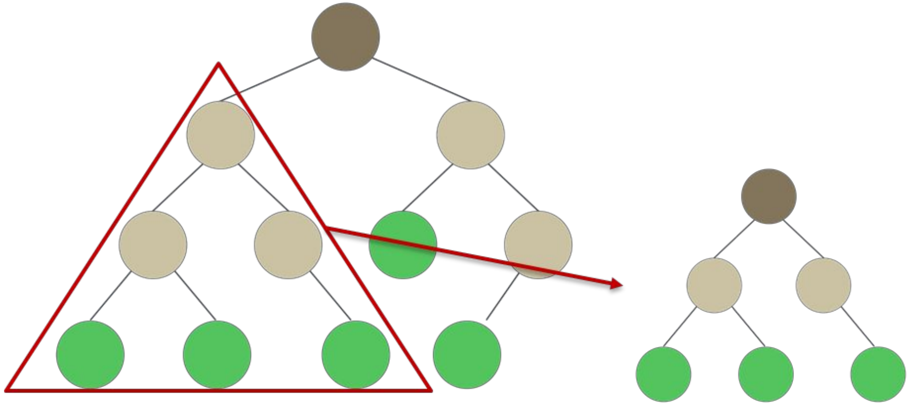
# BLG223E - Recitation 4

## Midterm Solutions & Binary Search Tree Exercise

Batuhan Can, Enes Erdoğan

Midterm questions have solved in the class, but they were removed from these slides.

# Recap : Tree Data Structure

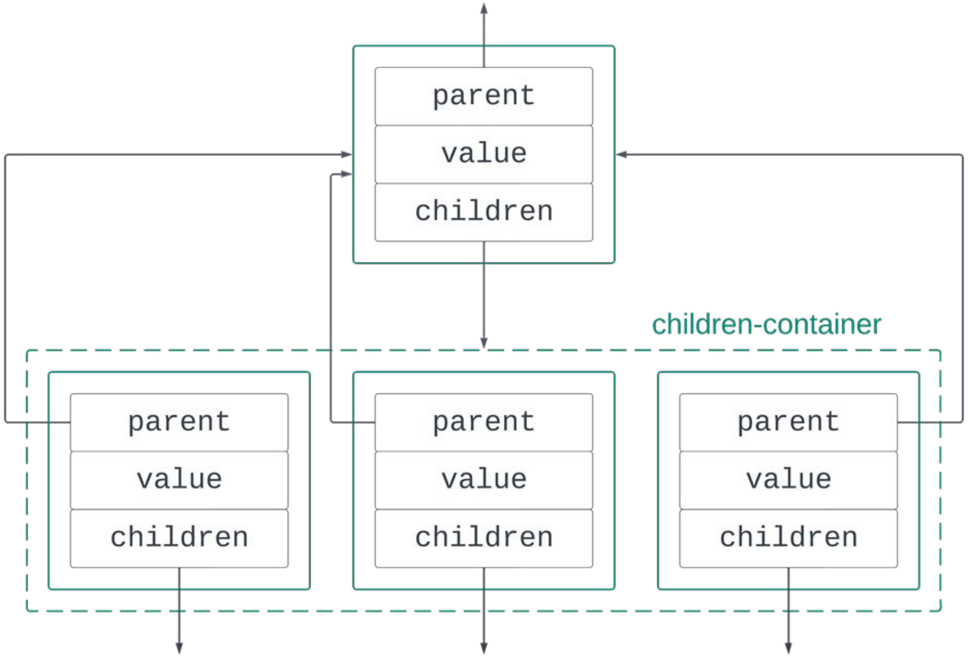


- Hierarchical data structure
  - ➢ `root` : the node at the top
  - ➢ `children` : nodes under the root
  - ➢ `leaves` : nodes at the bottom
- Recursion is a key method
  - ➢ a child may be the root of a **subtree**
  - ➢ Thus, a child may has its own children

# Recap : Tree Data Structure

Tree nodes have **three** key components

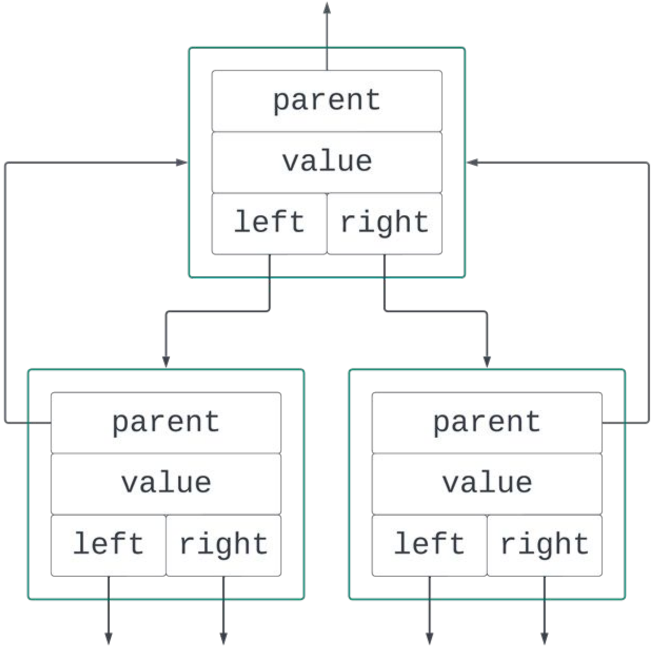
- parent
- value
- children



# Recap : Binary Search Tree

Binary Search Tree (BST) is a special structure design.

- All the internal nodes has two children, at most.
- Node structure of a BST can be updated as it has two pointers:
  - ➢ refers to the *left* child
  - ➢ refers to the *right* child



# Exercise : Binary Tree Construction

You are given a 2D integer array in the form of

- arr[i] = [parent_i, child_i, isLeft_i]
- parent_i = parent of child_i
- isLeft_i = is child_i the left child of parent_i

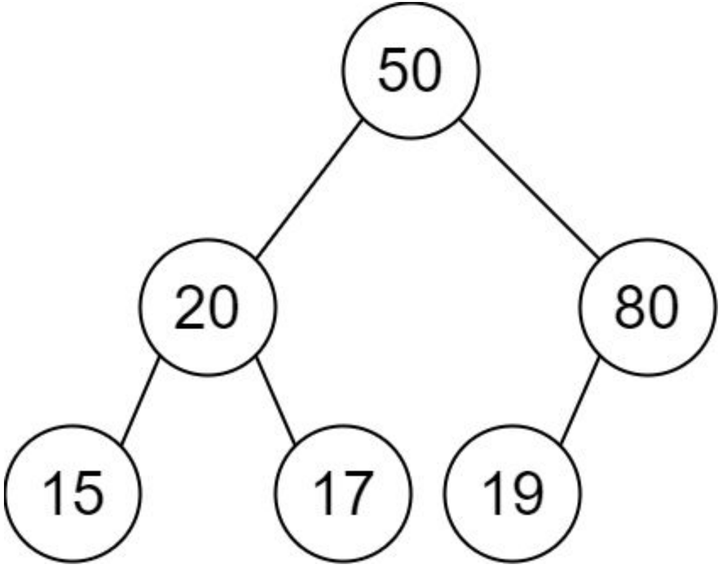
You are expected to construct the binary tree described by the array, then return its root.

Node ids are unique and less than 1000

**Example :**

input_arr = [[20,15,1], [20,17,0], [50,20,1], [80,19,1], [50,80,0]]

output_arr = [50, 20, 80, 15, 17, 19]



# Exercise : Binary Tree Construction

We are going to define/implement

- A struct to represent the binary tree
- An initialization function (init, create, etc.), and its evil twin a destroy function (free, delete, destroy, etc.)
- Fundamental functions of the binary tree (insert, delete, search)
- A function to print out the tree (in-order, pre-order, post-order)
- An algorithm to construct the tree from description array
- The famous main function to run our program

# Exercise : Binary Tree Construction

Representation of a binary tree can be done with a single struct => TreeNode

A tree node (in case of binary tree) has the following attributes:

- Data field (int, float, char, custom-structure)
- Pointer to left child
- Pointer to right child

# Exercise : Binary Tree Construction

- To initialize the tree, we are going to create a single node with some data, and null child pointers
- To destroy the structure, we may use the post-order traversal
- In basic functions such as insert, remove, and search; we are going to benefit from the ordered characteristic of the binary tree
- Print function can be implemented in many ways (in-, pre-, post-)
- And the final boss : An algorithm to construct the tree, as given in the question.