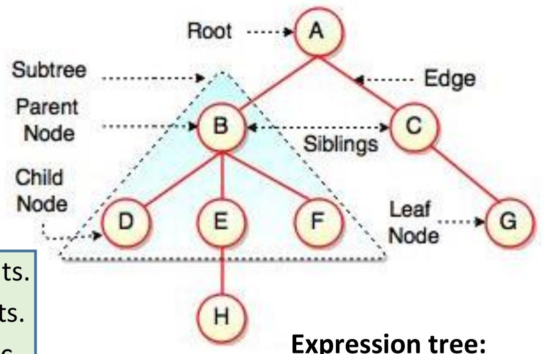
Trees

1. Tree Data Structure

Array list, linked list, stack, and queue are linear data structures – they have logical start and logical end.

A **tree** is a dynamic, **non-linear** data structure of **hierarchically** linked nodes that satisfies the following conditions:

- There is only one node at the top – **root**.
- Each node has only one **parent node** (a node at a higher level).
- There are no cycles/loops, i.e. traversing from a node cannot reach the same node again via different path.

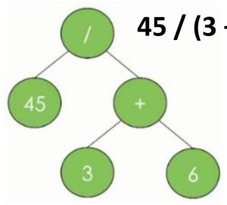


Advantages of Trees:

- They are very efficient for searching, inserting and removal of elements.
- The structure conveys information – the relationships among elements.
- Natural organization of data: family tree, file system, decision tree, etc.

Expression tree:

45 / (3 + 6)



Components of a Tree (terminology):

- **Root** – a node that has no parent node (**ancestor**); iterations start from the root
- **Edge** - parent-child (**not inheritance!**) connected pair
- **Leaf** – a node that has no children nodes (**descendants**); it links down to `null`

Some types of trees:

- Organized by natural data ordering – Search Trees
- Organized in a specific order in reference to the root – Heap
- Organized by element frequency – Huffman Coding Tree

2. Binary Tree

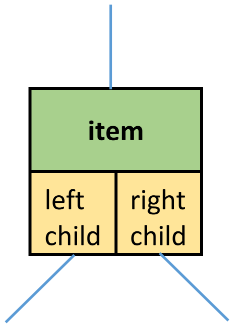
In a general tree every node can have multiple children nodes (a **list** of children nodes).

In a **binary** tree a node can have at most **two children nodes**.

Each tree node consists of:

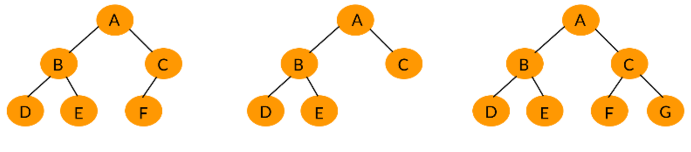
- a data item
- a reference to the left child
- a reference to the right child

```
private class Node<T>{
    private T item;
    private Node<T> left;
    private Node<T> right;
```



Types of Binary Trees:

- **Complete** Binary Tree- all levels are **completely filled**, **except** possibly the last one, and all nodes on the last level are as far left as possible
- **Full** Binary Tree – every node has **zero or two children**
- **Perfect** Binary Tree - every node other than the leaves has exactly two children; all leaves are at the same level

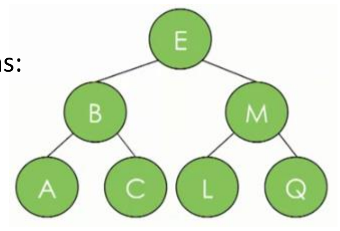


Complete Binary Tree   Full Binary Tree   Perfect Binary Tree

3. Binary Search Tree

To find an item quickly in a binary tree it must satisfy the following conditions:

- Left subtrees' items are **lesser** than parent's item
- Right subtrees' items are **greater** than parent's item



4. The Java TreeSet Class

The `TreeSet` class implements a number of interfaces. Its functionality is beyond the scope of this course.