**TREES**

Suppose you want to maintain hierarchy of employees then we always use trees

Suppose you want to maintain folders and files then we always use trees

Whenever we want to keep data in hierarchy then always use trees

* NODE
  + It holds data and address of next node
* Root
  + It is the starting or top most element of tree through which we can access each and every element of tree
* Children
  + All those nodes which are connected to current node.
* Parent
  + Node which is used to access current node or node at level (i-1)
* Ancestor
  + All the nodes which lies above current node, involve for accessing current node
* Descendants
  + All the nodes which lies below current node, involve for accessing current node
* Siblings
  + All nodes which have same parent node
* Leaves
  + Which do not have child node

Numbering for level starts at 0 i.e root is at level 0

Degree of a node = no of children of a node

Depth of a node = length of path from root to that node (could be defined via 2 ways 1: via edge or 2: via nodes )

Height of a tree = distance from root to deepest node = (n+1)

Where n denotes level of that node

What to keep in class of node

Data and Arraylist of references as there could be multiple child