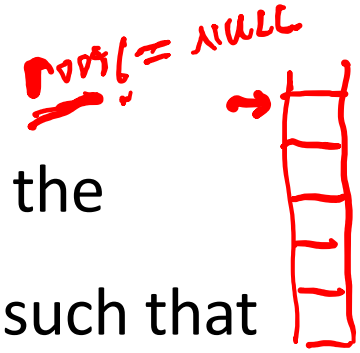
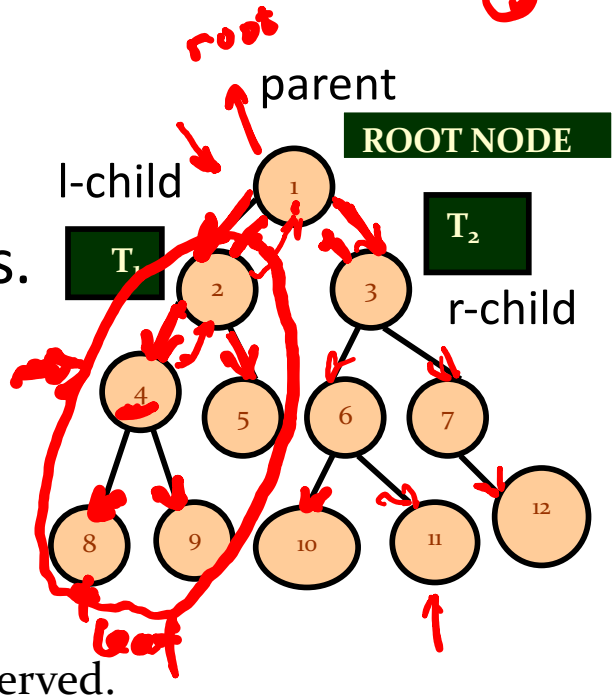
Tree

- A tree is a non-linear data structure which consists of a collection of nodes arranged in a hierarchical order.



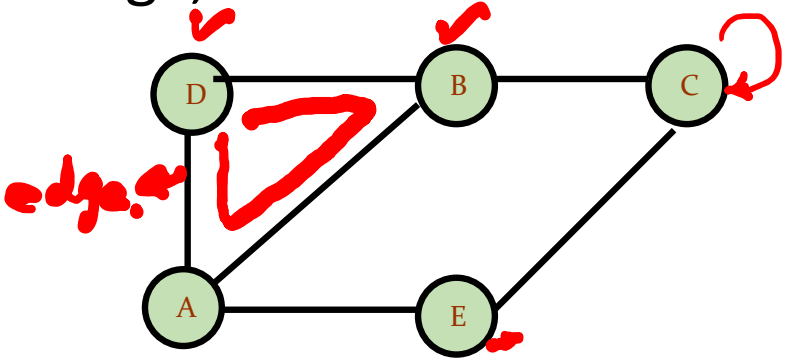
- One of the nodes is designated as the root node, and the remaining nodes can be partitioned into disjoint sets such that each set is the sub-tree of the root.
- A binary tree is the simplest form of tree which consists of a root node and left and right sub-trees.



- The root element is pointed by 'root' pointer.
- If root = NULL, then it means the tree is empty.

Graph

- A graph is a non-linear data structure which is a collection of *vertices* (also called *nodes*) and *edges* that connect these vertices.
- A graph is often viewed as a generalization of the tree structure, where instead of a having a purely parent-to-child relationship between nodes, any kind of complex relationship can exist.
- Every node in the graph can be connected with any other node.
- When two nodes are connected via an edge, the two nodes are known as neighbors.



Abstract Data Type

An ***Abstract Data Type*** **(ADT)** is a way of encapsulating data and operations on that data into a single unit.

For example, stacks and queues are perfect examples of an abstract data type. We can implement both these ADTs using an array or a linked list. This demonstrates the "abstract" nature of stacks and queues.

Operations on Data Structures

Traversing – accessing each data item exactly once so that it can be processed

Searching – find the location of a data item satisfying a given constraint

Inserting – add a new item to a collection of data items

Deleting – remove a data item from a collection of data items

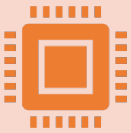
Sorting – arrange data items in a particular order

Merging – combine 2 or more collections of data items

Algorithm

An “algorithm" is a formally defined procedure for performing some calculation. It provides a blueprint to write a program to solve a particular problem.

It is considered to be an effective procedure for solving a problem in finite number of steps. That is, a well-defined algorithm always provides an answer and is guaranteed to terminate.



Algorithms are mainly used to achieve **software re-use**. Once we have an idea or a blueprint of a solution, we can implement it in any high-level language like C, C++, Java, so on and so forth.

Algorithm

Write an algorithm to find whether a number is even or odd

Step 1: Input the first number as A

Step 2: IF A%2 =0

Then Print "EVEN"

ELSE

PRINT "ODD"

Step 3: END

Time and Space Complexity of an Algorithm

- Time complexity of an algorithm depends on the number of instructions executed. This number is primarily dependent on the size of the program's input and the algorithm used.
- The space needed by a program depends on:

✓ Fixed part includes space needed for storing instructions, constants, variables, and structured variables.

✓ Variable part includes space needed for recursion stack, and for structured variables that are allocated space dynamically during the run-time of the program.

TA and Graders

- TA Name: **Zhenyu Xu**
- Email: zhenxu@ttu.edu

- Grader Name: **Agrawal, Daniel**
- Email: daagrawa@ttu.edu

- Lab Grader Name**: Anthony Hill**
- Email: anthohil@ttu.edu

- Every Lab will be announced in advance on Blackboard. Stay tuned.