

Topic 7

Binary Trees

CSE1303 Part A

Data Structures and Algorithms

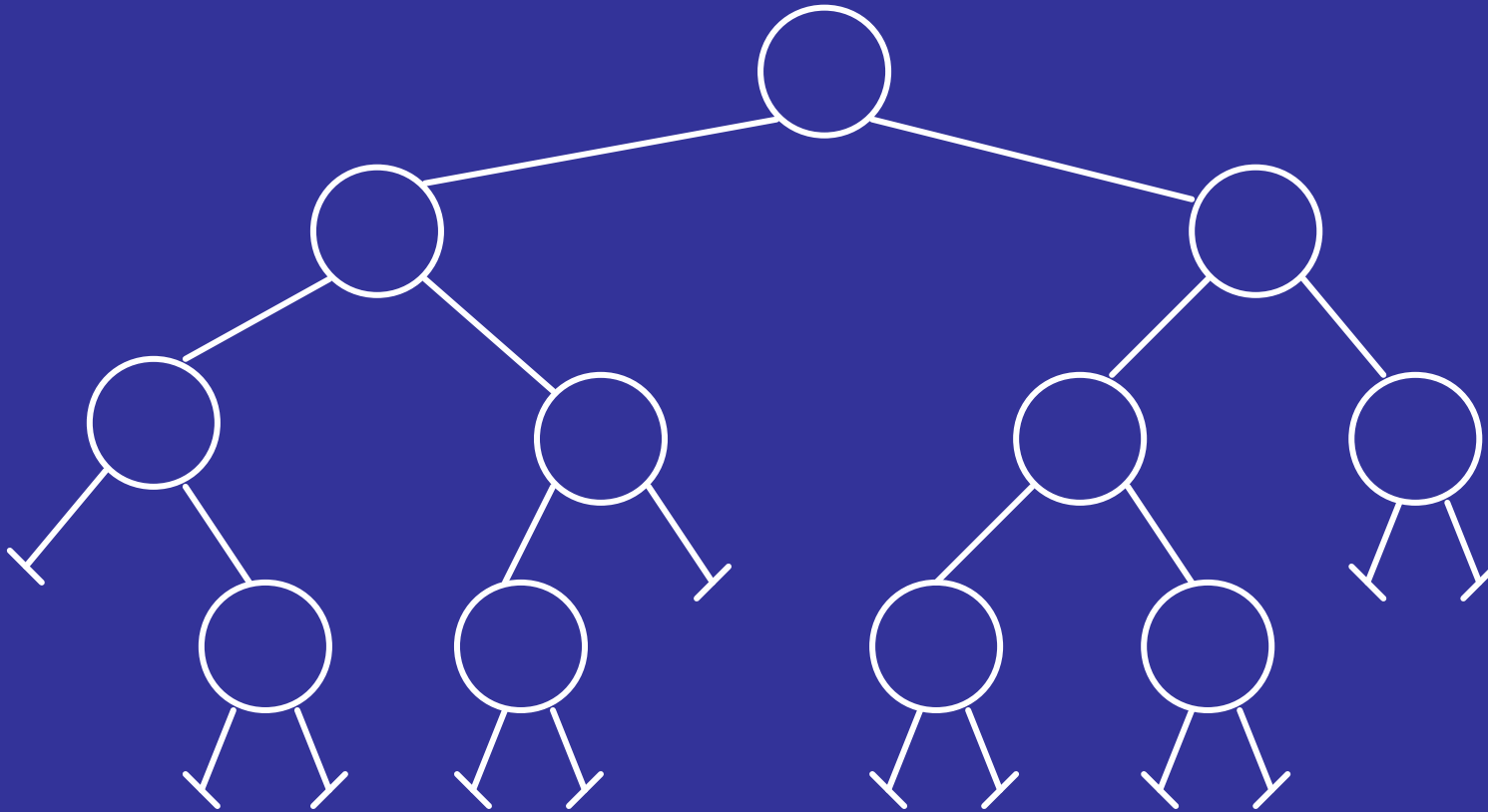
Overview

- Trees.
- Terminology.
- Traversal of Binary Trees.
- Expression Trees.
- Binary Search Trees.

Trees

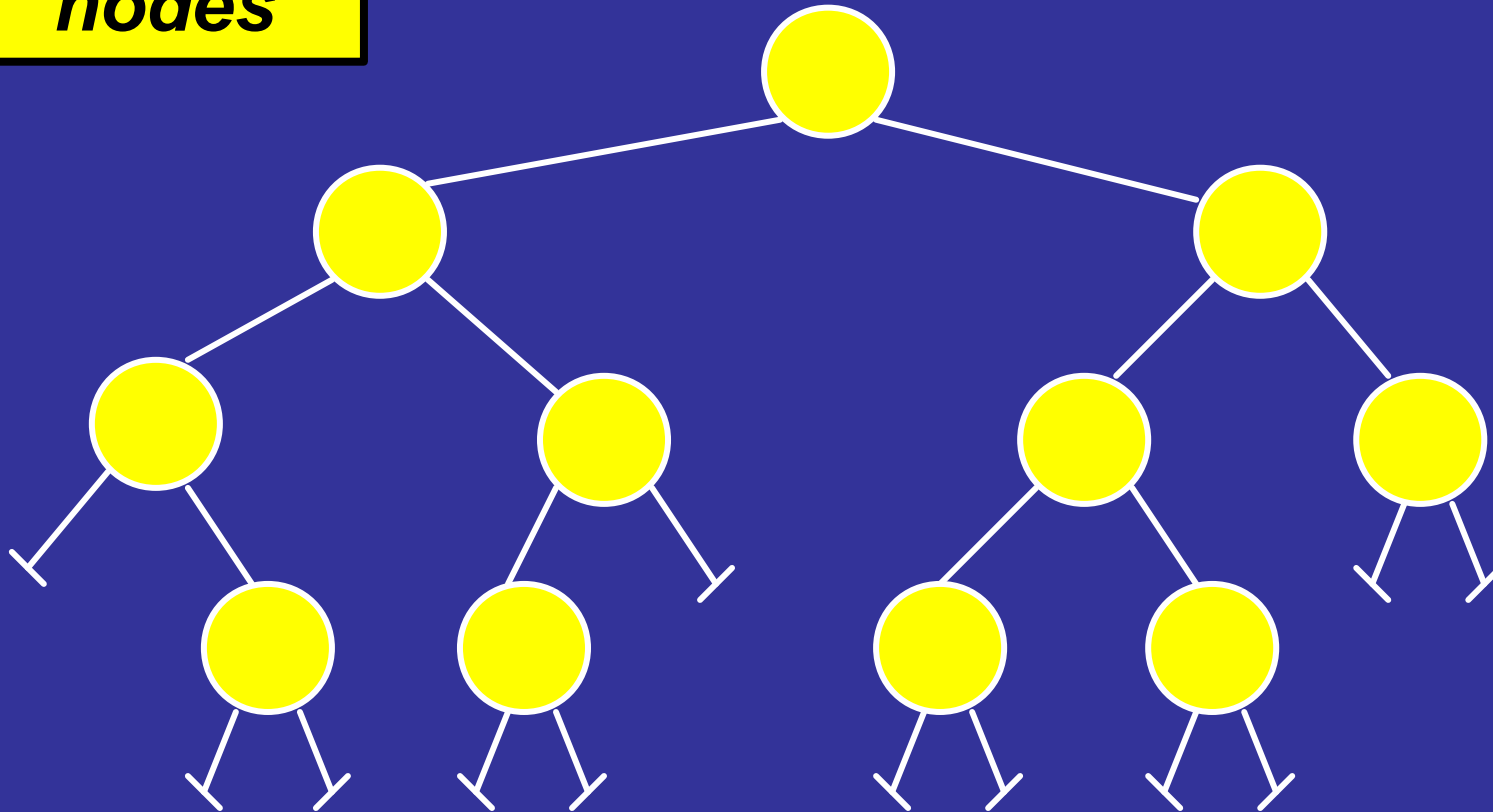
- Family Trees.
- Organisation Structure Charts.
- Program Design.
- Structure of a chapter in a book.

Parts of a Tree



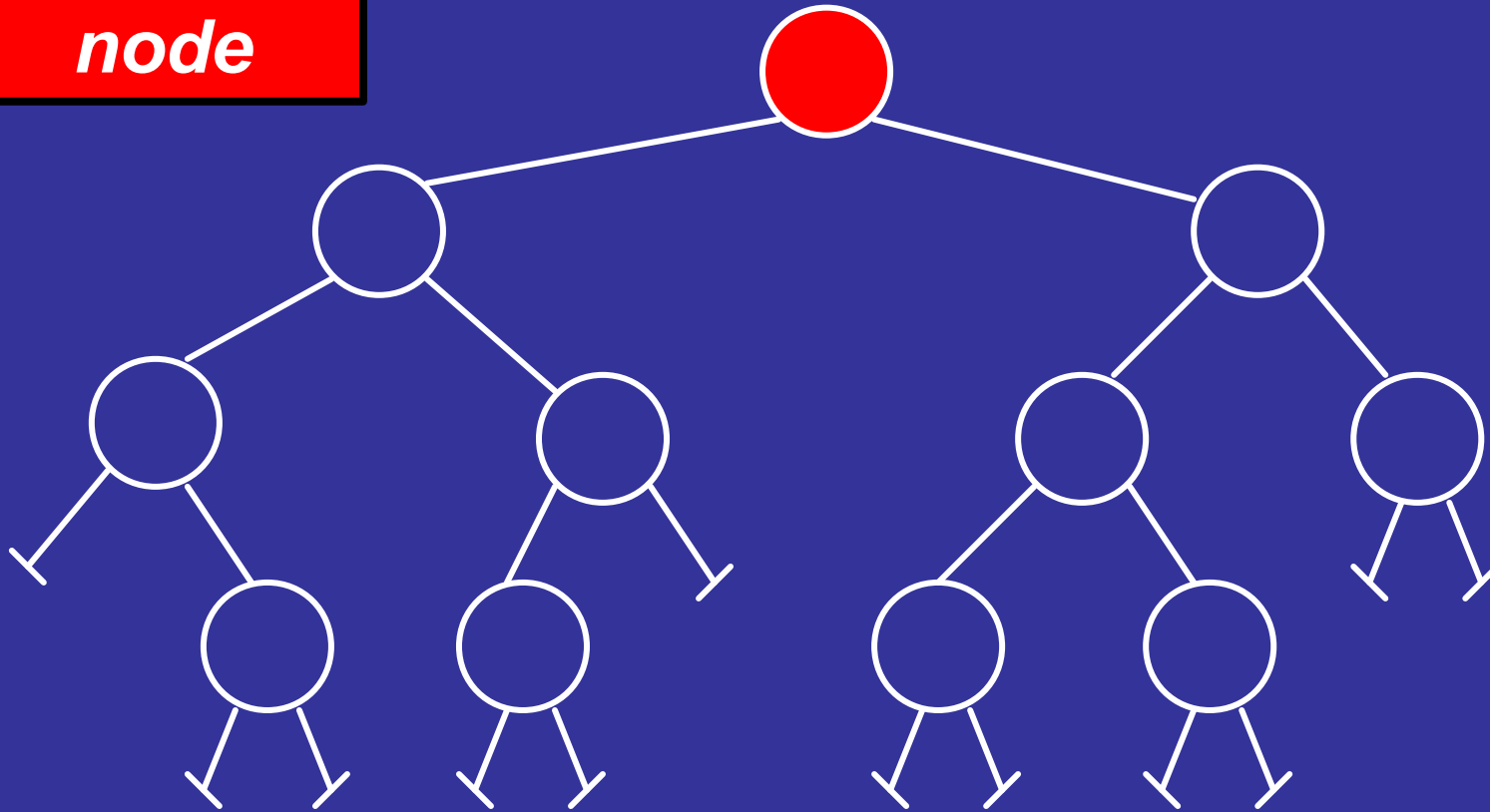
Parts of a Tree

nodes



Parts of a Tree

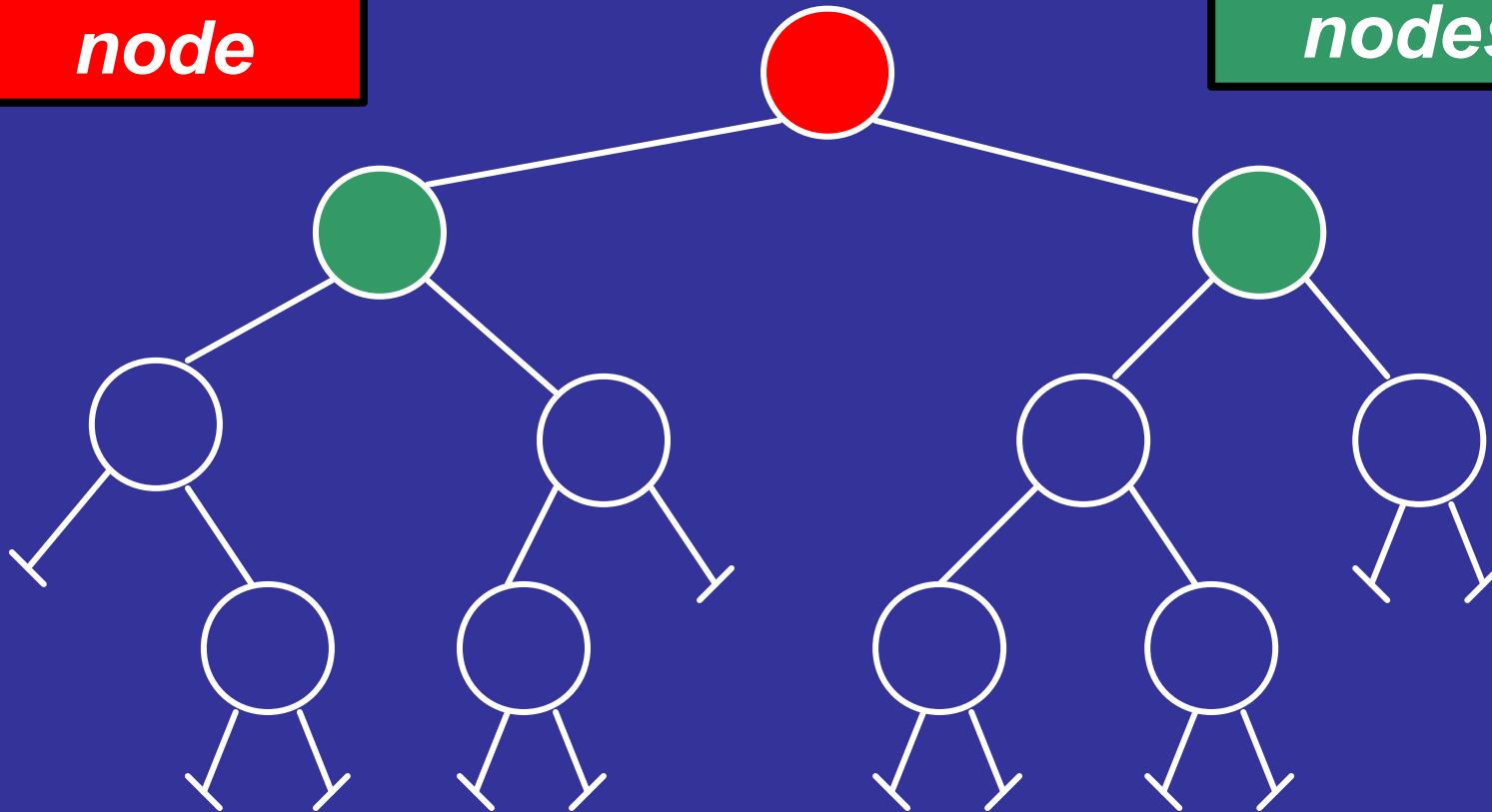
*parent
node*



Parts of a Tree

*parent
node*

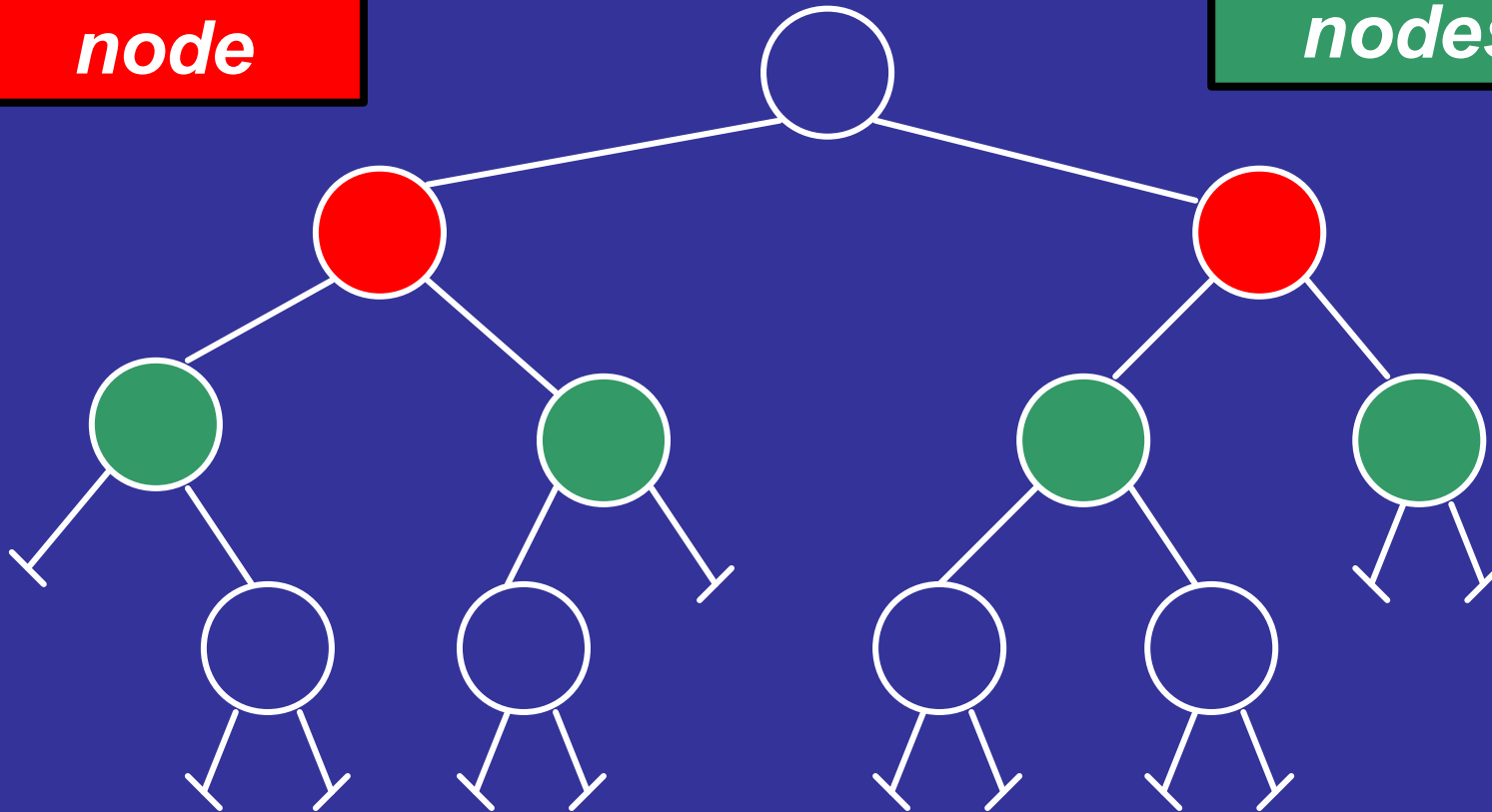
*child
nodes*



Parts of a Tree

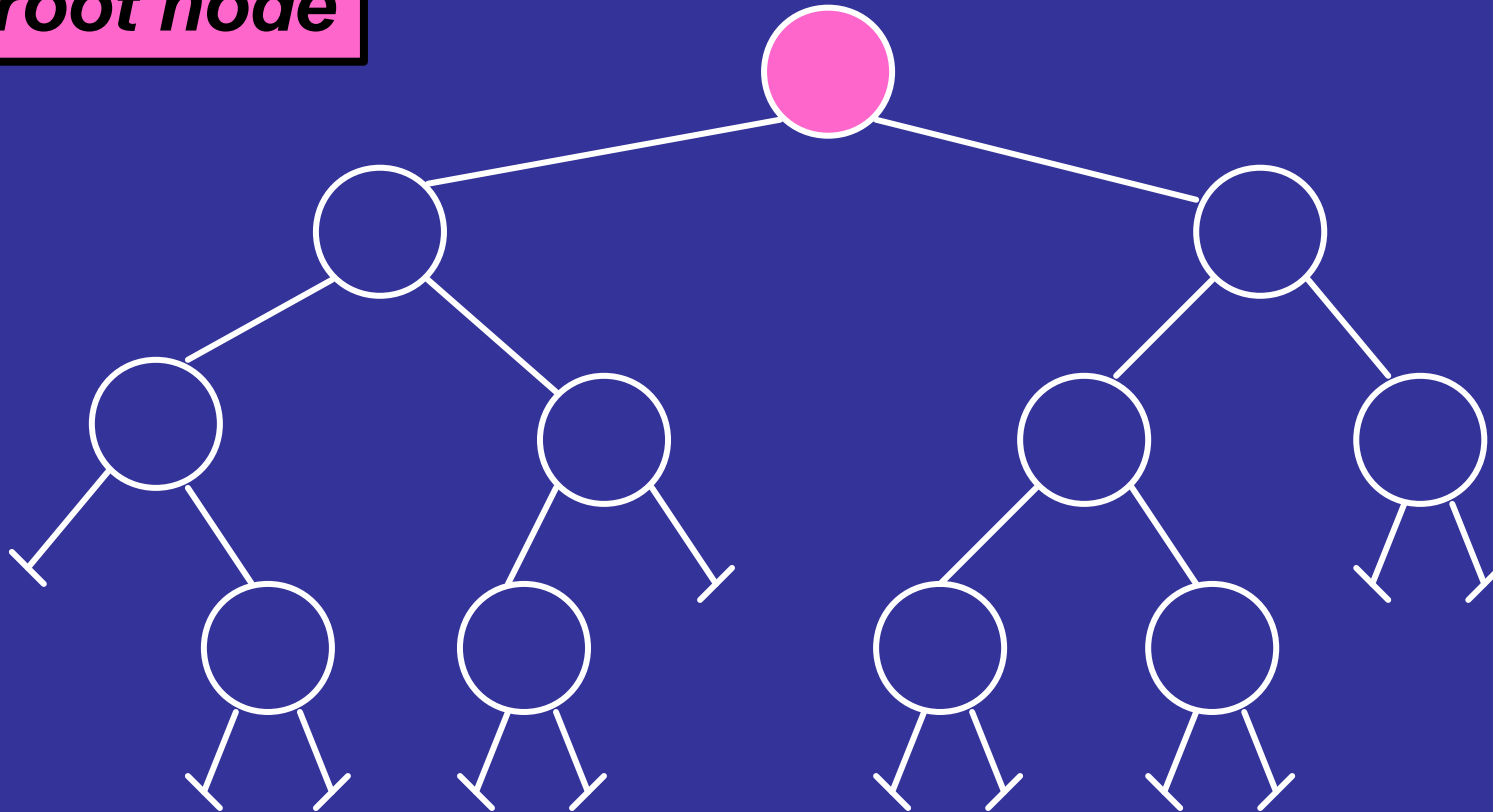
*parent
node*

*child
nodes*



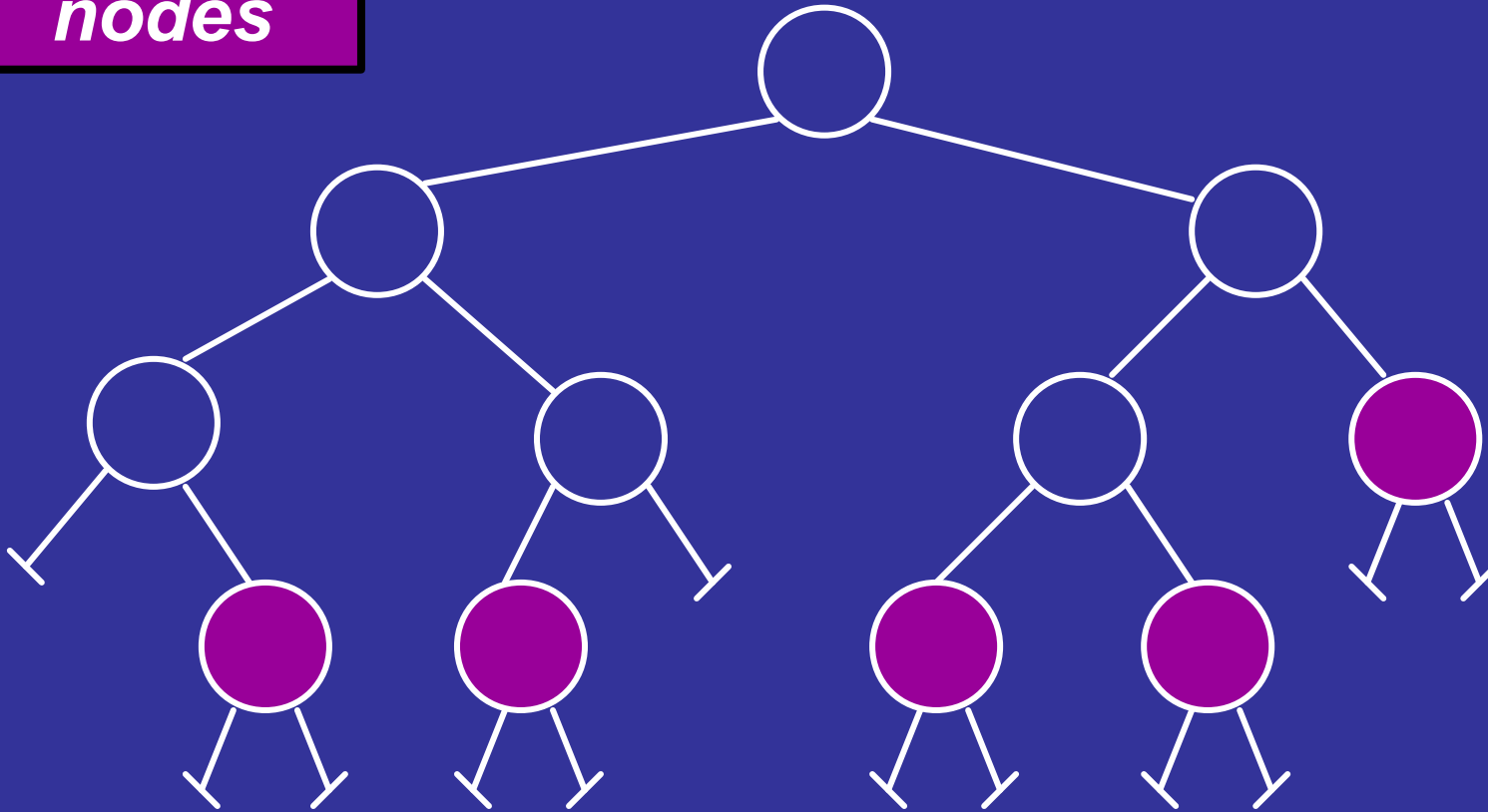
Parts of a Tree

root node



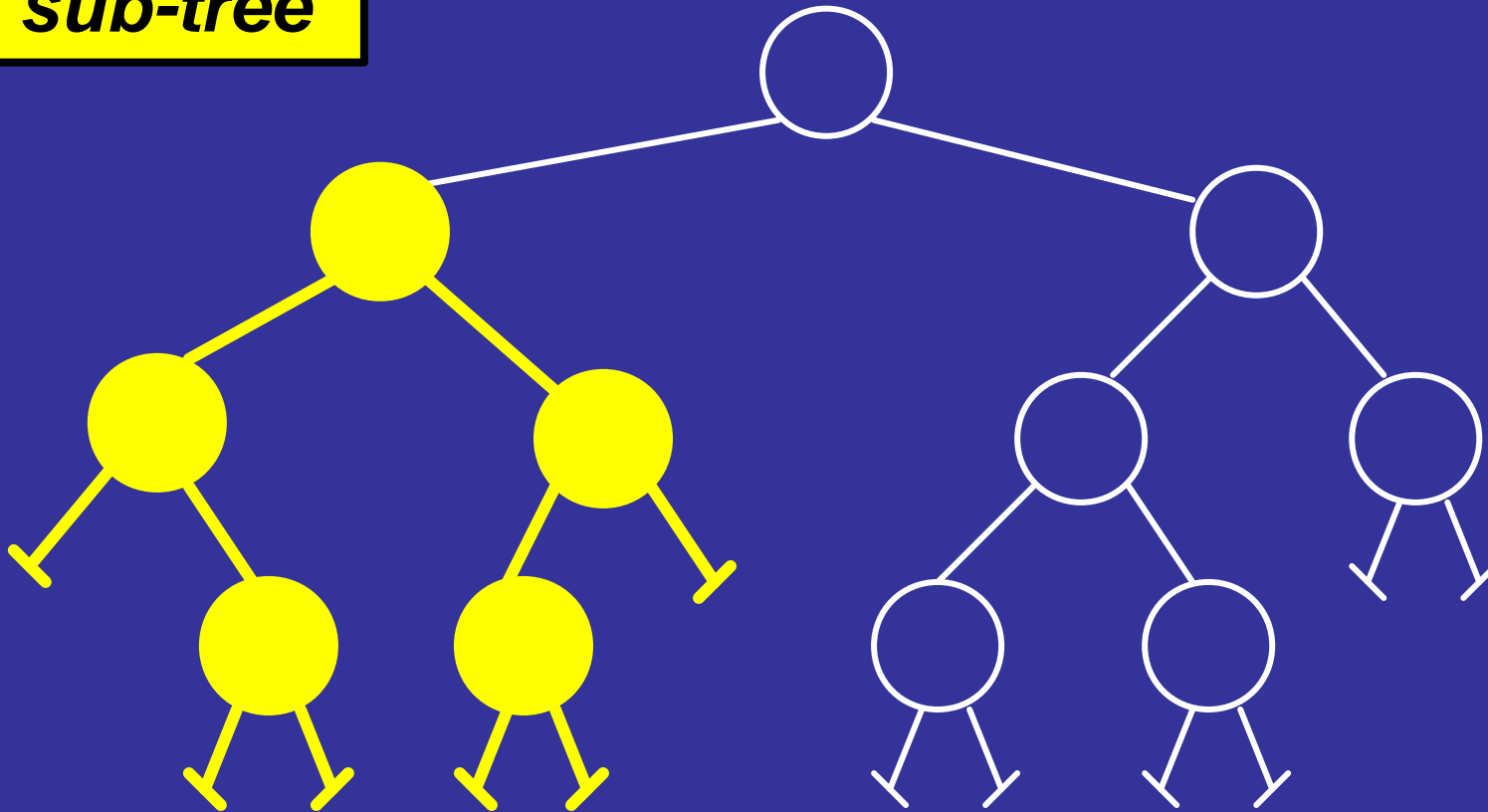
Parts of a Tree

*leaf
nodes*



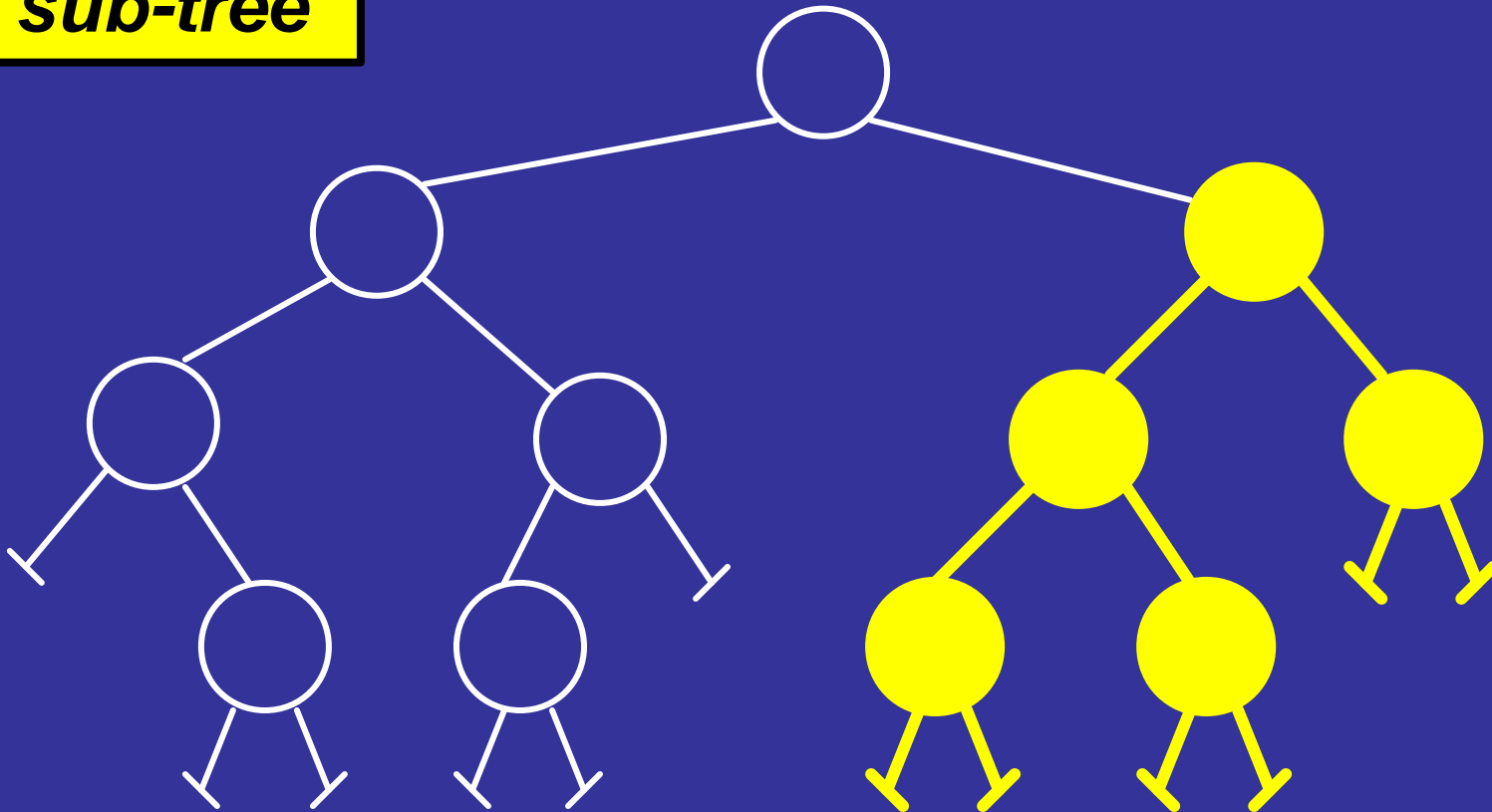
Parts of a Tree

sub-tree



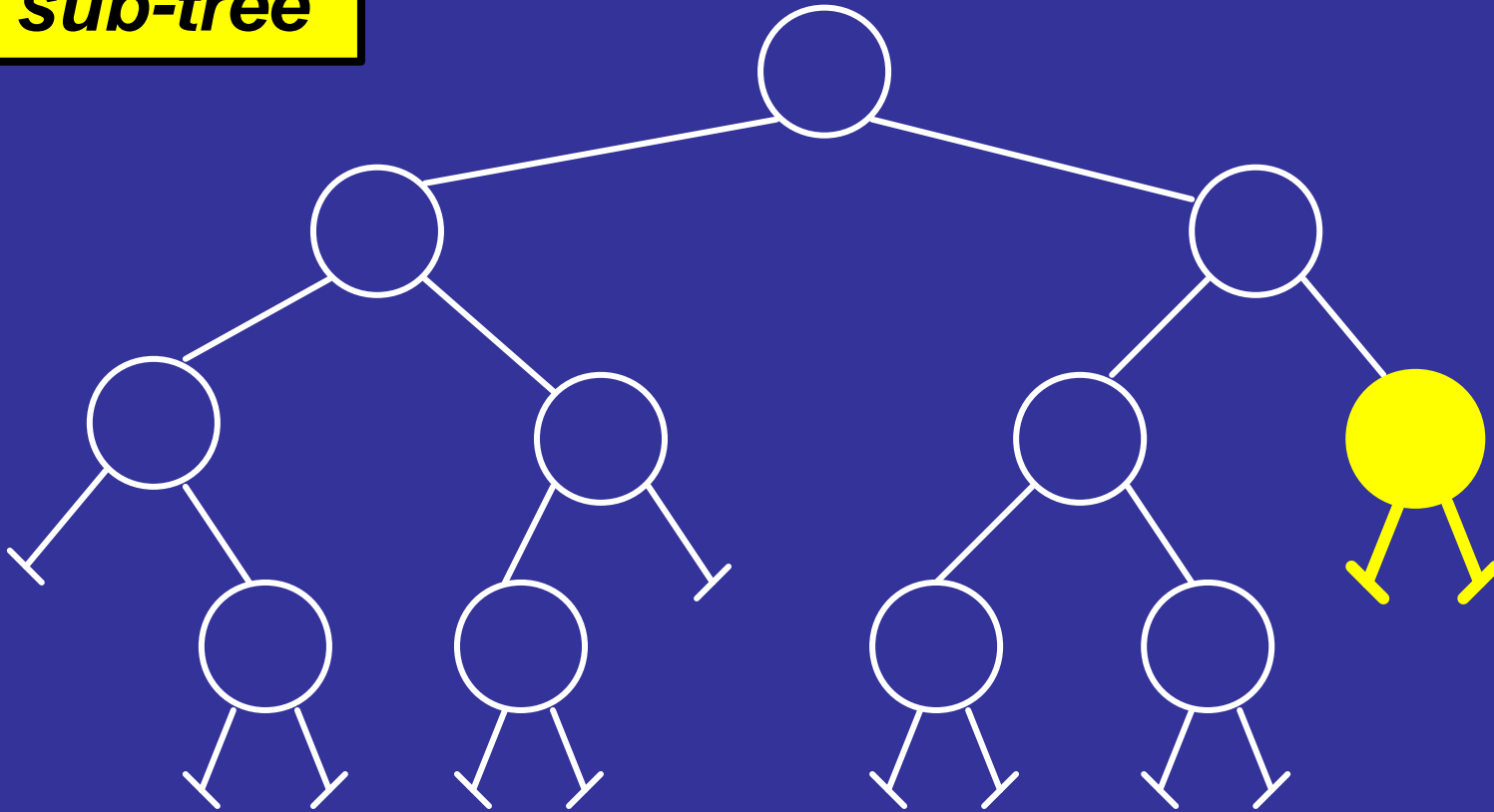
Parts of a Tree

sub-tree



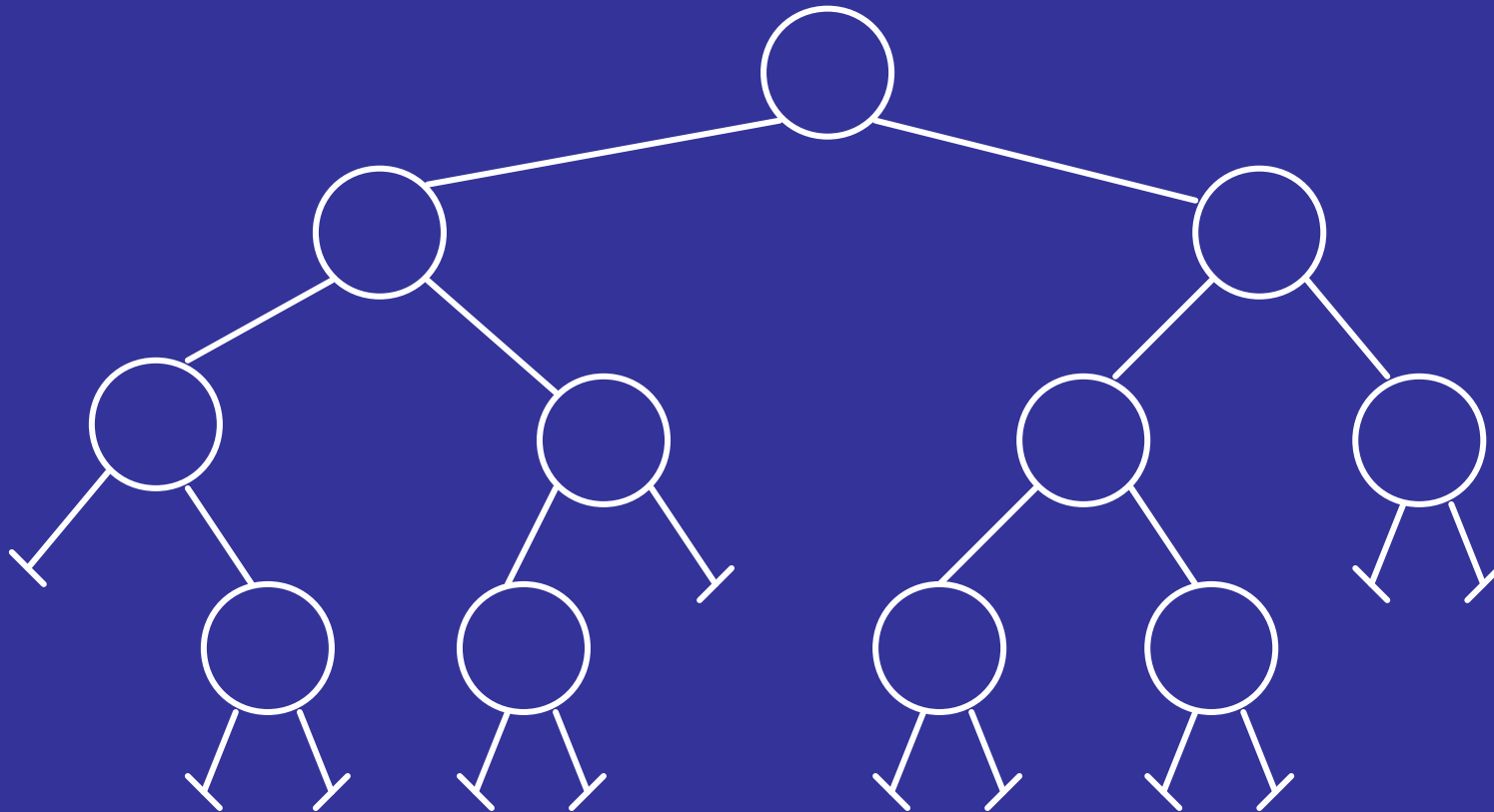
Parts of a Tree

sub-tree



Binary Tree

- Each node can have **at most 2** children



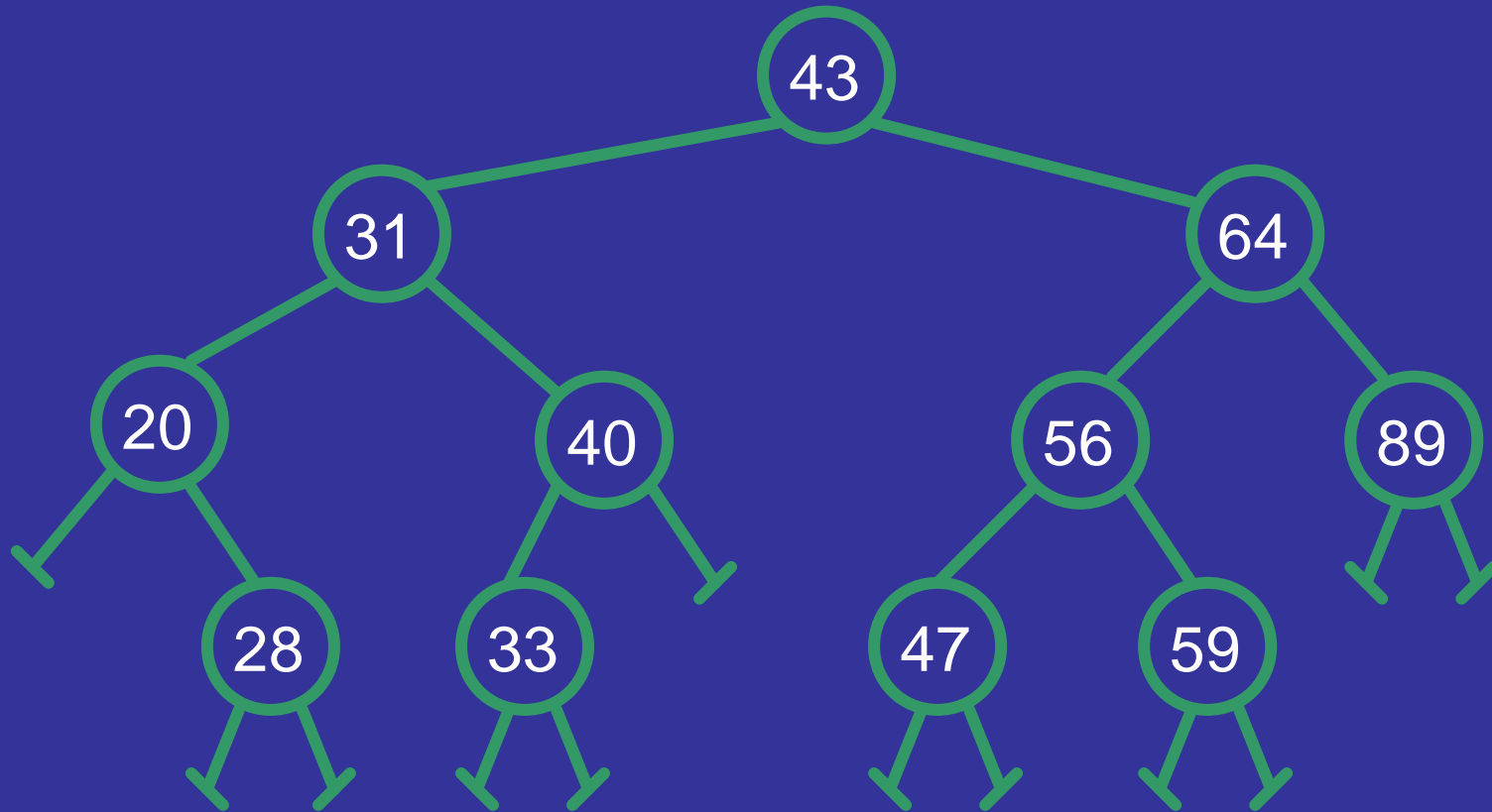
Traversal

- Systematic way of visiting all the nodes.
- Methods:
 - Preorder, Inorder, and Postorder
- They all traverse the left subtree before the right subtree.
- The name of the traversal method depends on when the node is visited.

Preorder Traversal

- Visit the node.
- Traverse the left subtree.
- Traverse the right subtree.

Example: Preorder



43	31	20	28	40	33	64	56	47	59	89
----	----	----	----	----	----	----	----	----	----	----

Inorder Traversal

- Traverse the left subtree.
- Visit the node.
- Traverse the right subtree.

Example: Inorder



20	28	31	33	40	43	47	56	59	64	89
----	----	----	----	----	----	----	----	----	----	----

Postorder Traversal

- Traverse the left subtree.
- Traverse the right subtree.
- Visit the node.

Example: Postorder

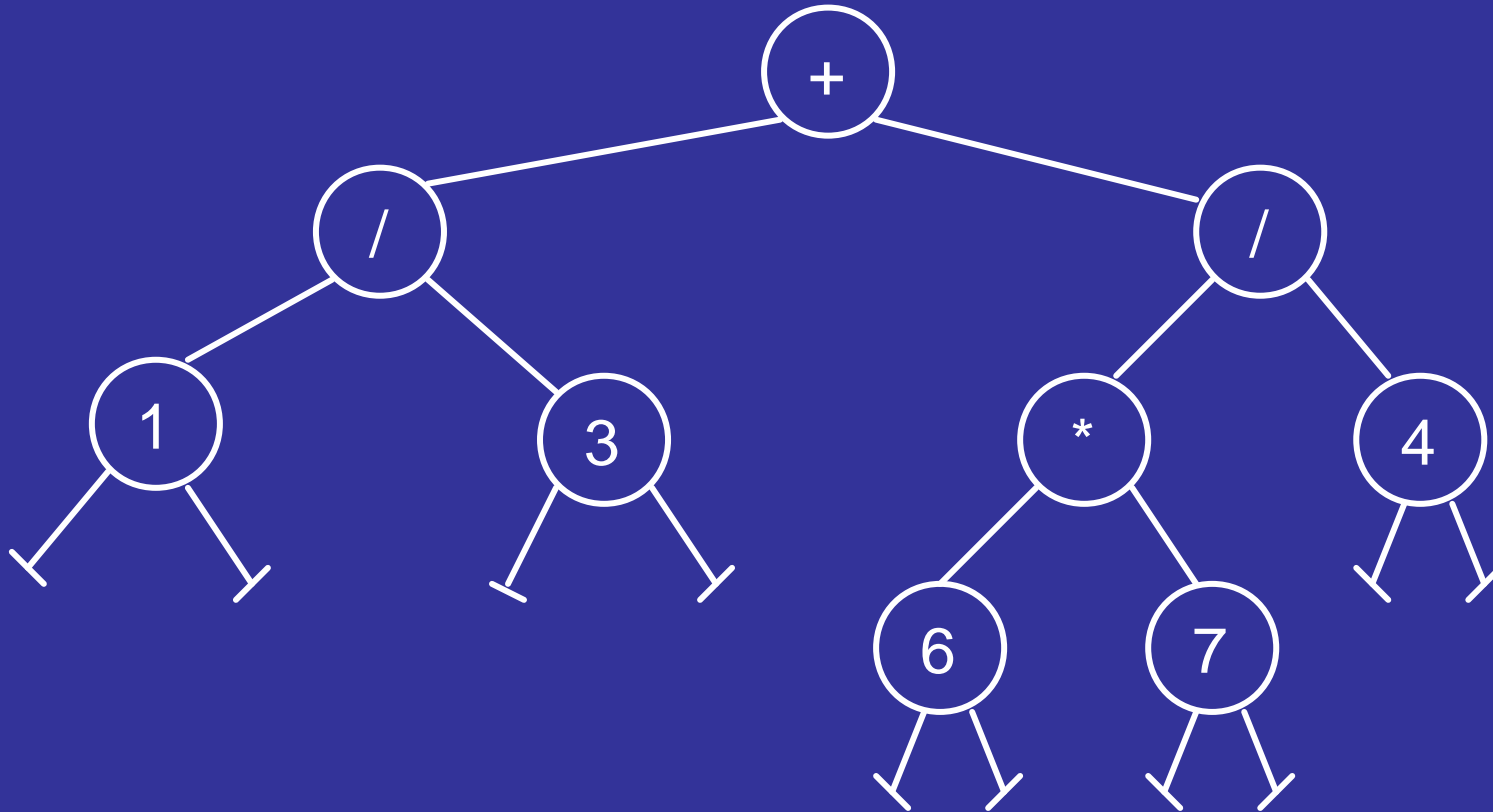


28	20	33	40	31	47	59	56	89	64	43
----	----	----	----	----	----	----	----	----	----	----

Expression Tree

- A Binary Tree built with operands and operators.
- Also known as a parse tree.
- Used in compilers.

Example: Expression Tree

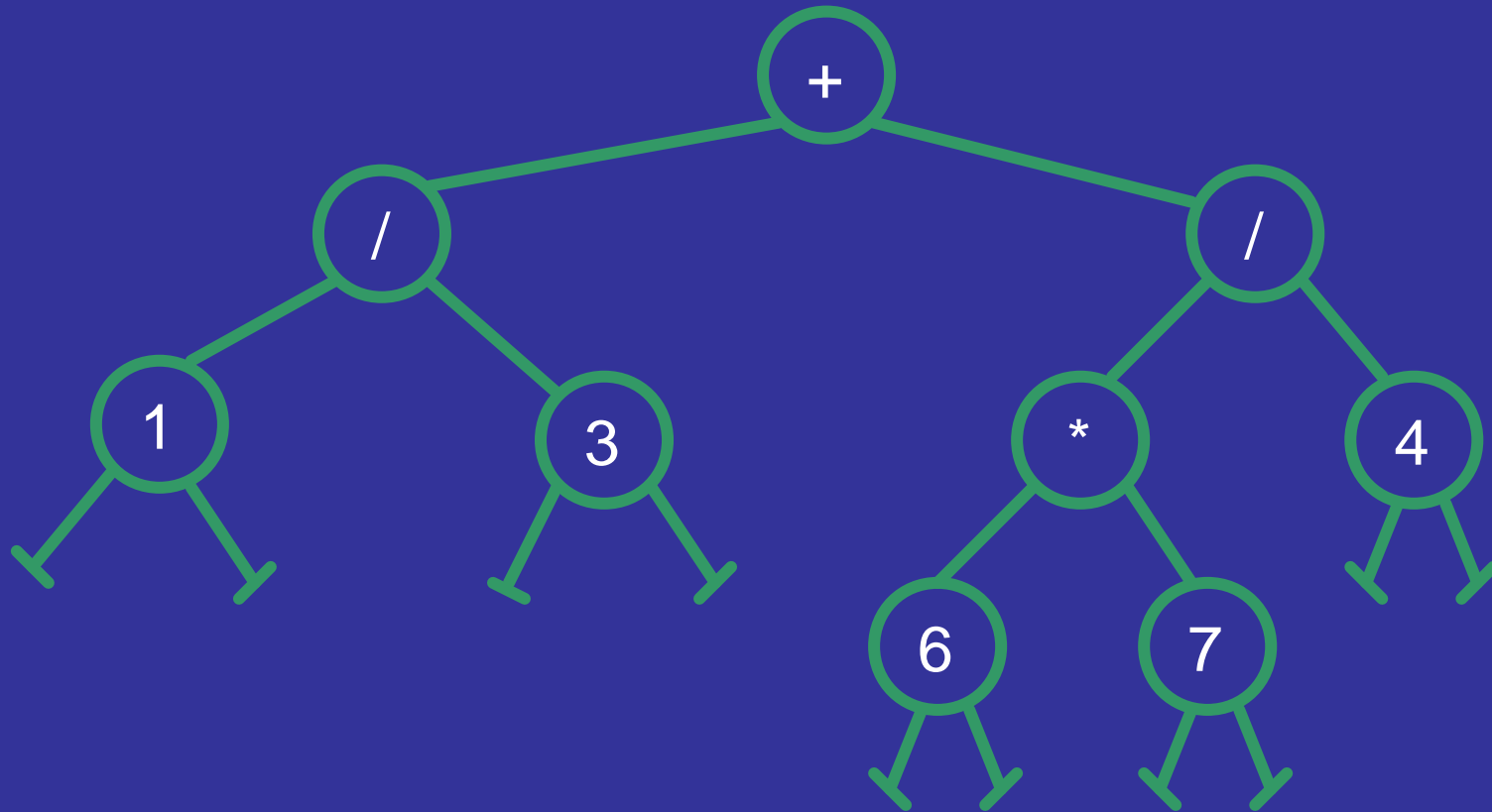


$1/3 + 6*7/4$

Notation

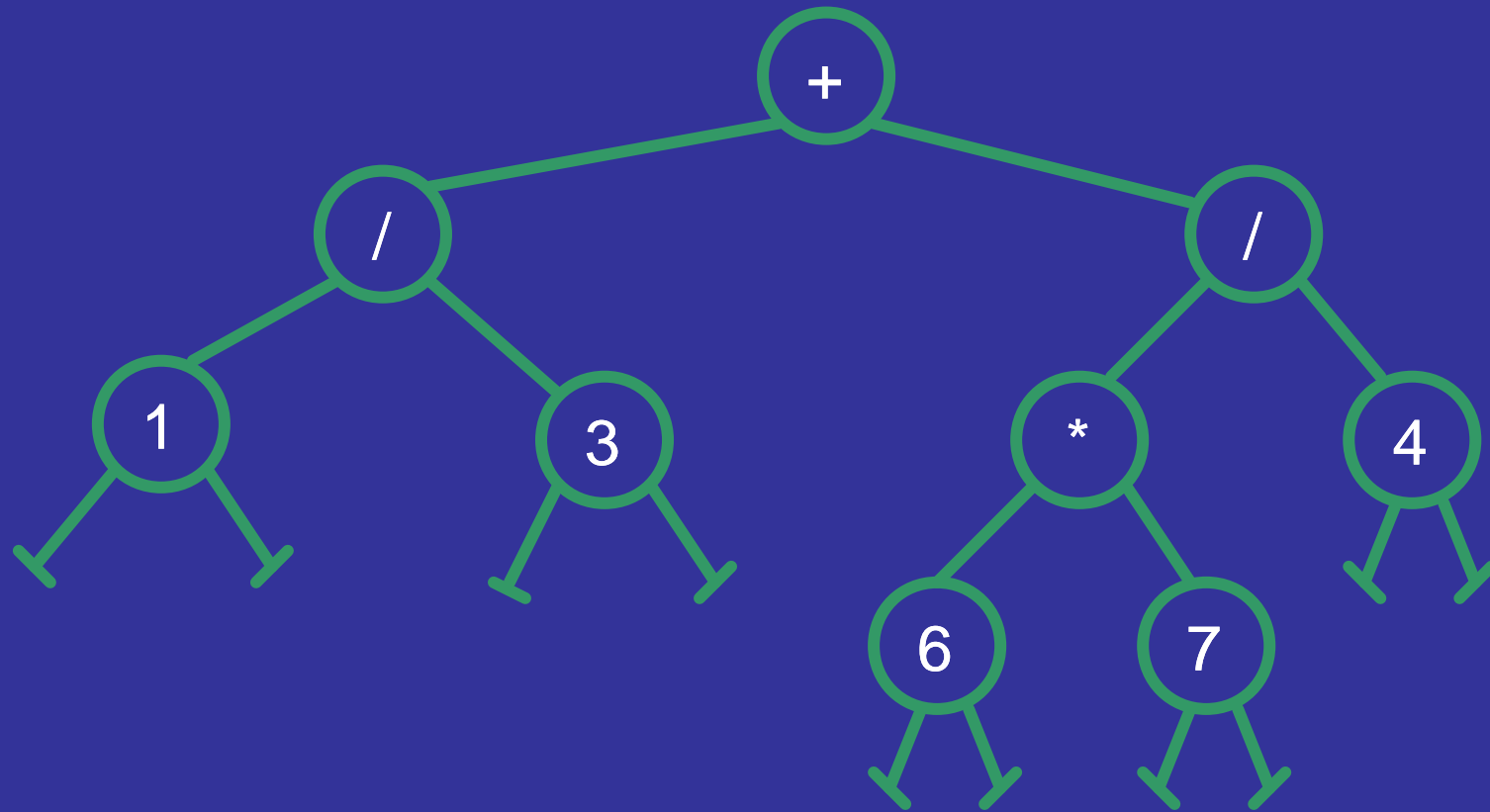
- Preorder
 - Prefix Notation
- Inorder
 - Infix Notation
- Postorder
 - Postfix Notation

Example: Infix



1	/	3	+	6	*	7	/	4
---	---	---	---	---	---	---	---	---

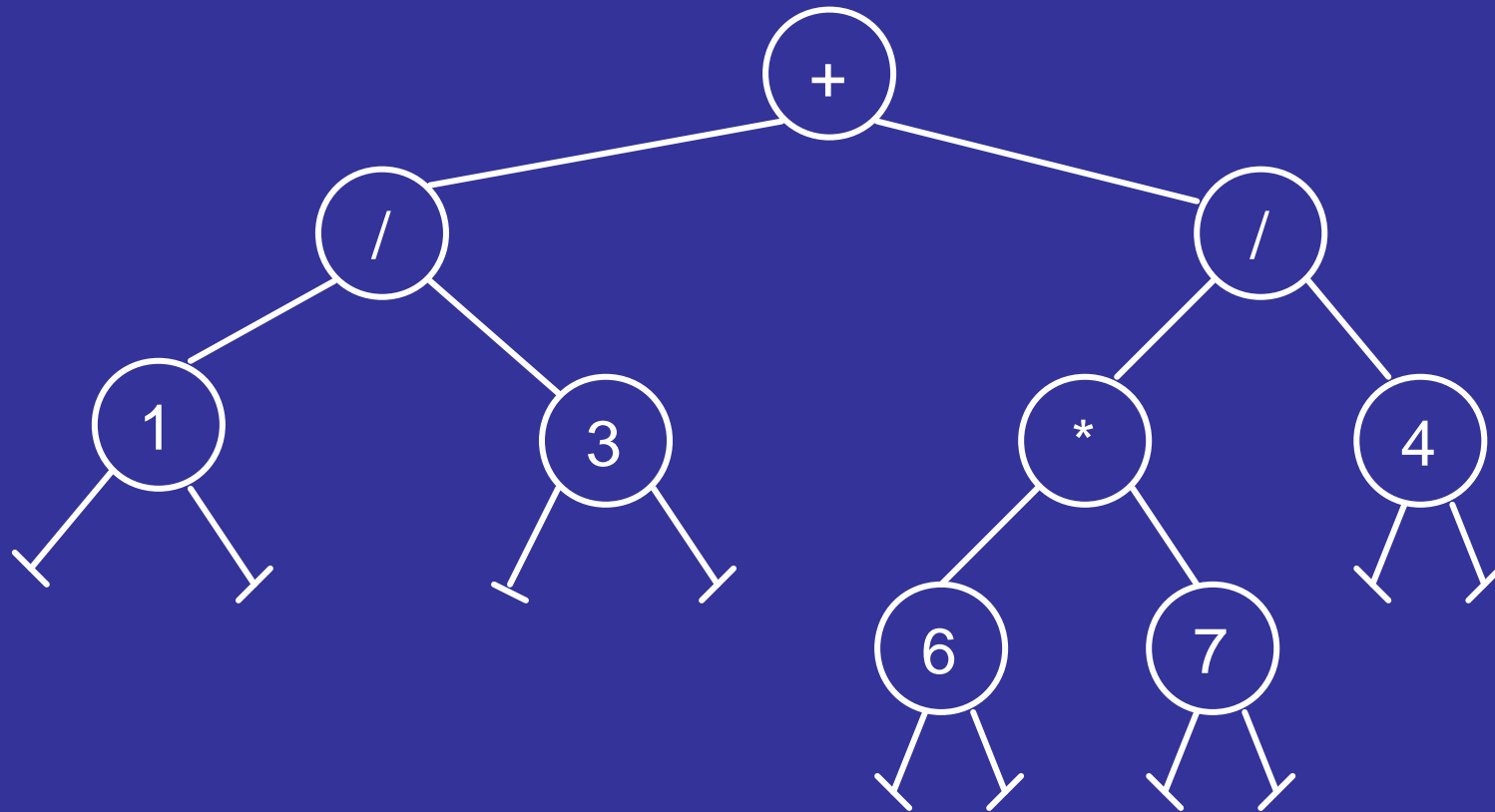
Example: Postfix



Recall: Reverse Polish Notation



Example: Prefix



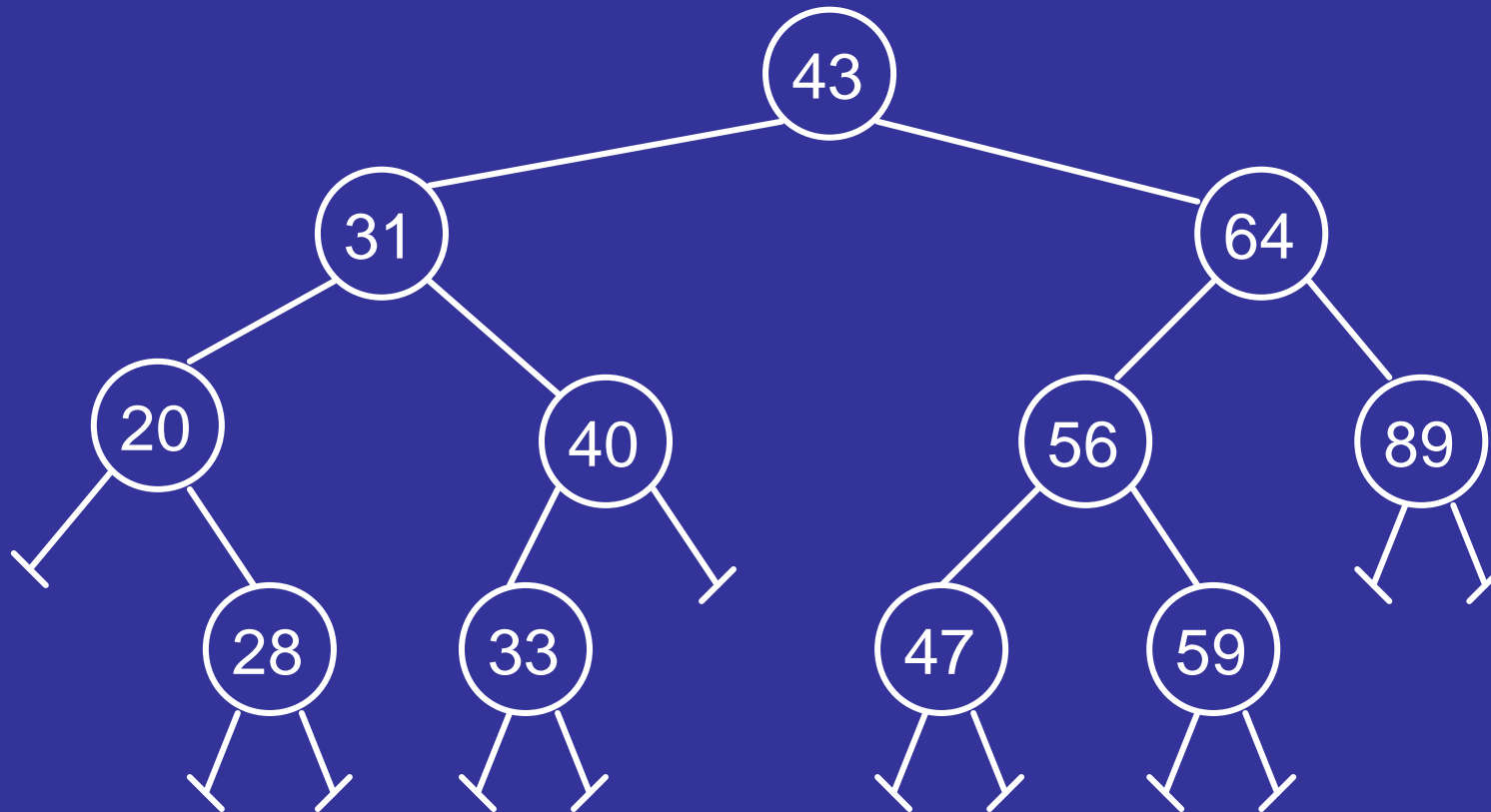
+	/	1	3	/	*	6	7	4
---	---	---	---	---	---	---	---	---

Binary Search Tree

A Binary Tree such that:

- Every node entry has a **unique** key.
- **All** the keys in the **left subtree** of a node are **less** than the key of the node.
- **All** the keys in the **right subtree** of a node are **greater** than the key of the node.

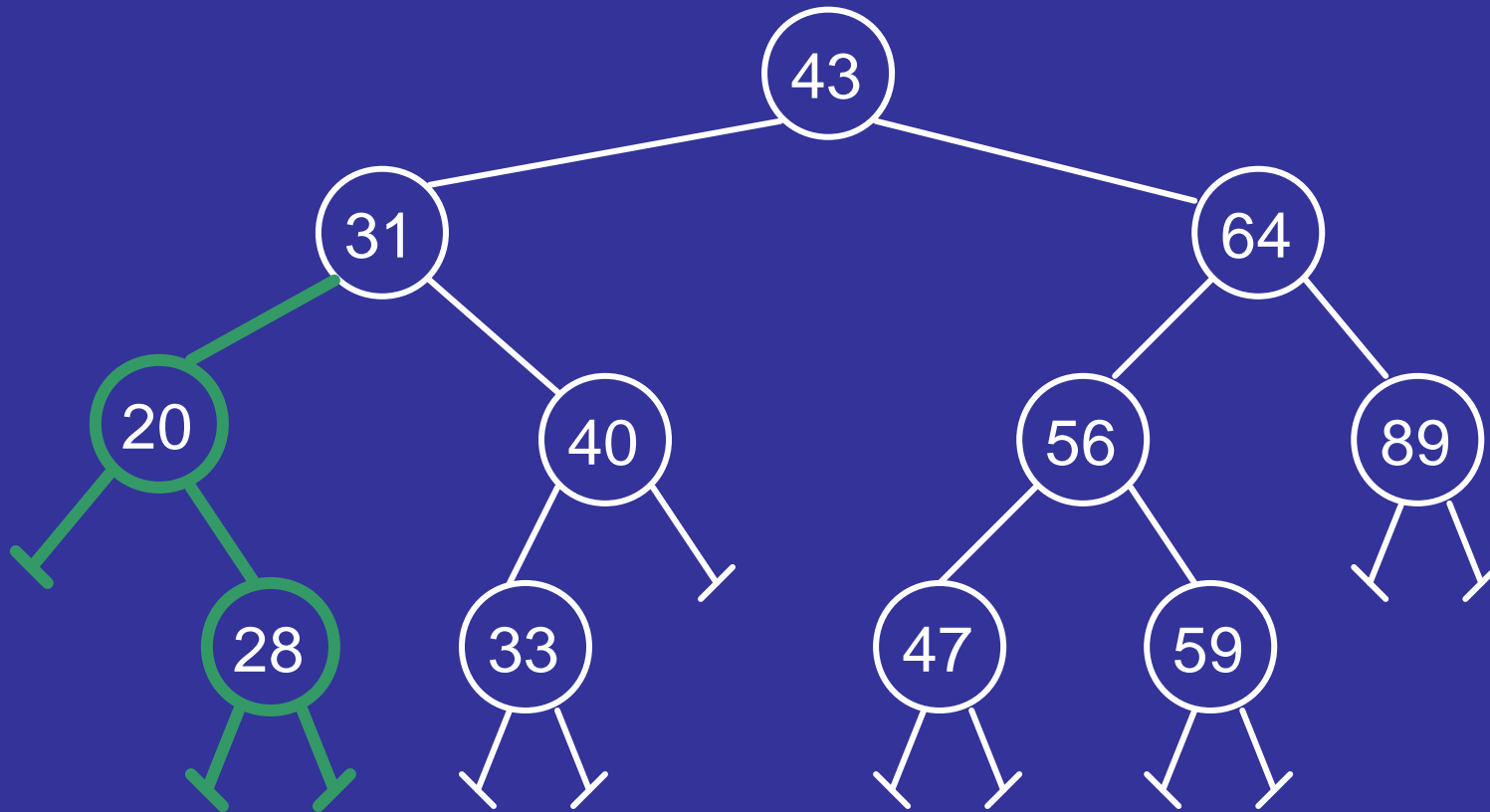
Example 1: *key is an integer*



Example 1: *key is an integer*



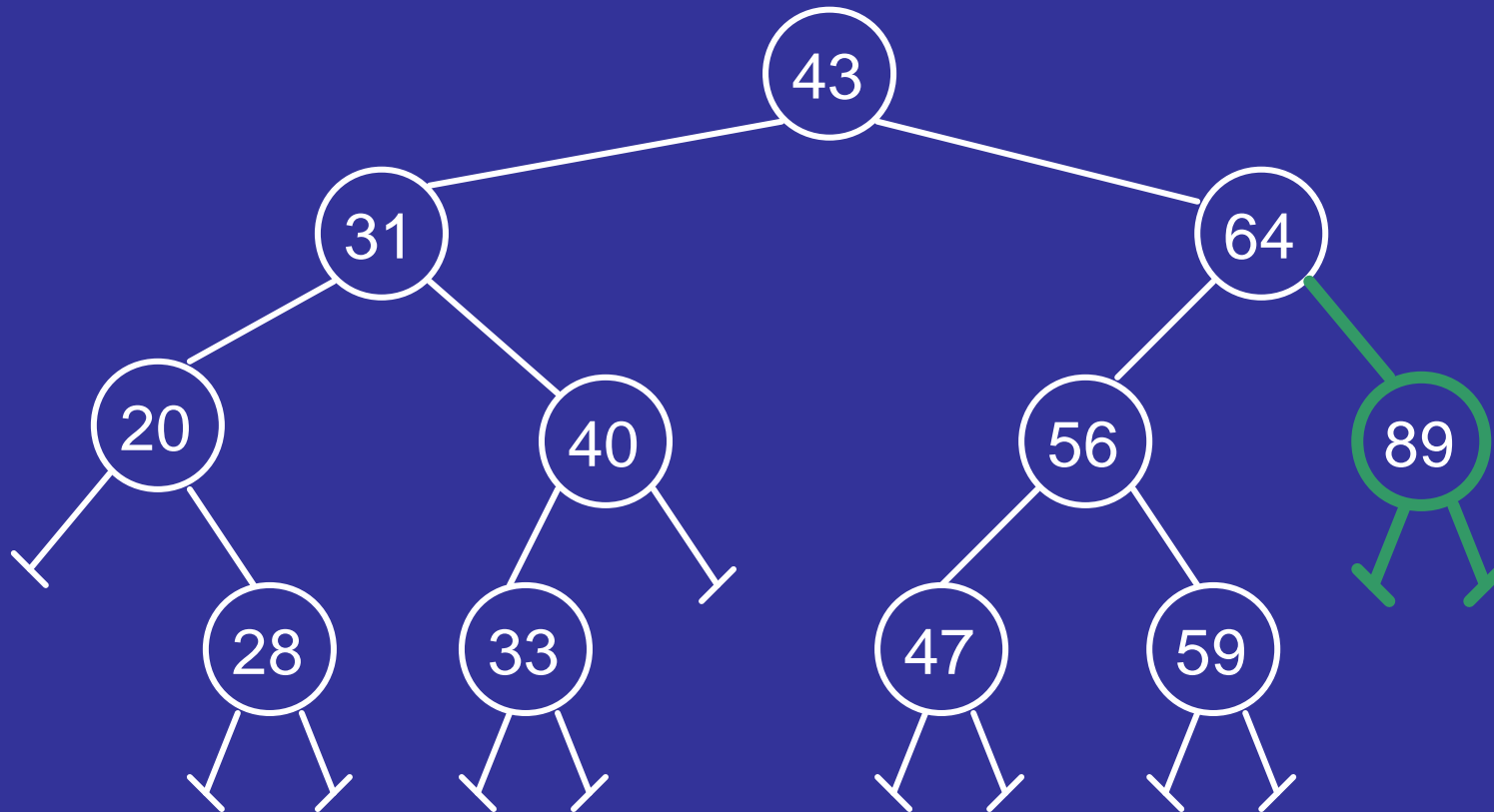
Example 1: *key is an integer*



Example 1: *key is an integer*



Example 1: *key is an integer*

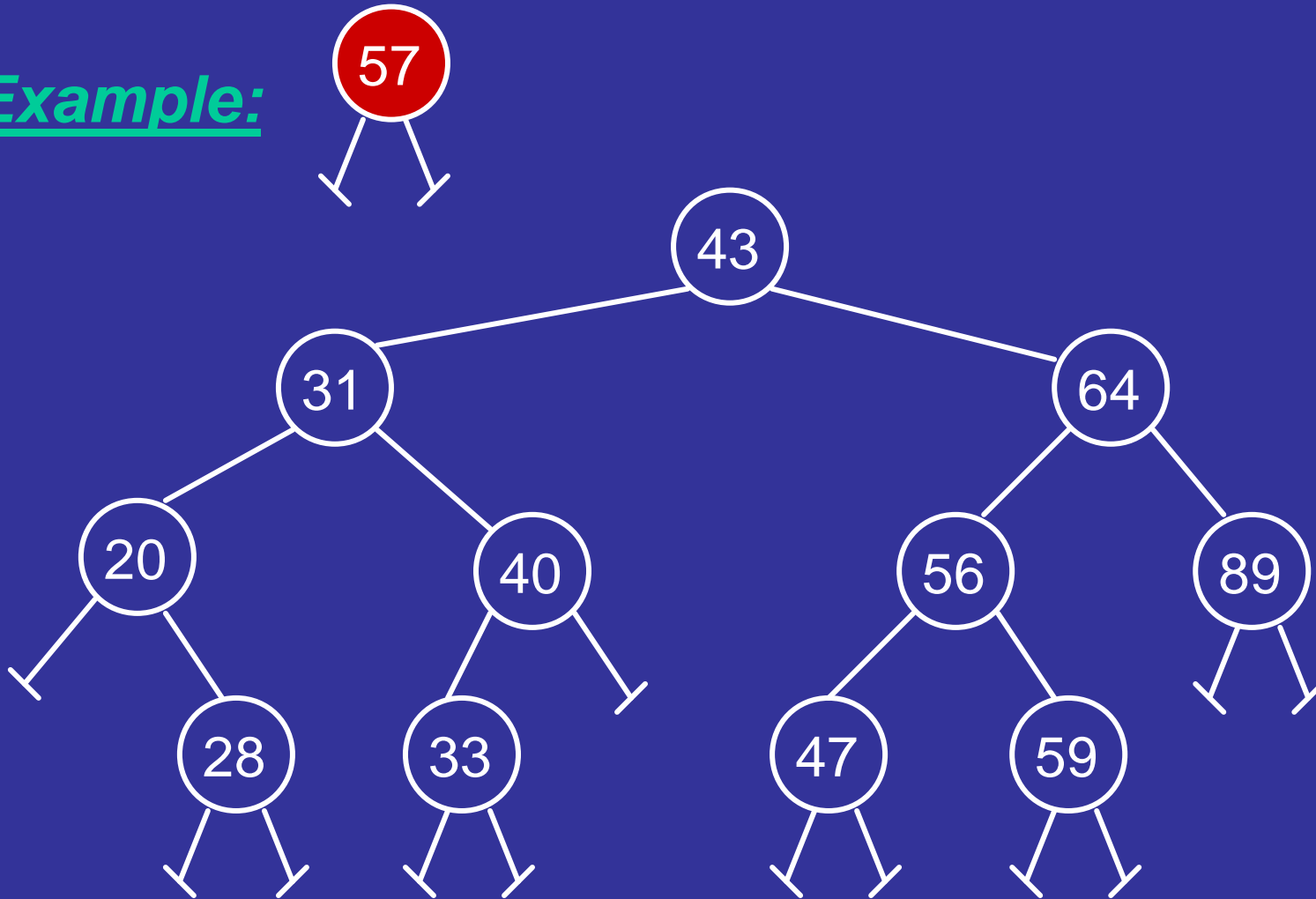


Insert

- Create **new node** for the item.
- Find a **parent node**.
- Attach new node as a **leaf**.

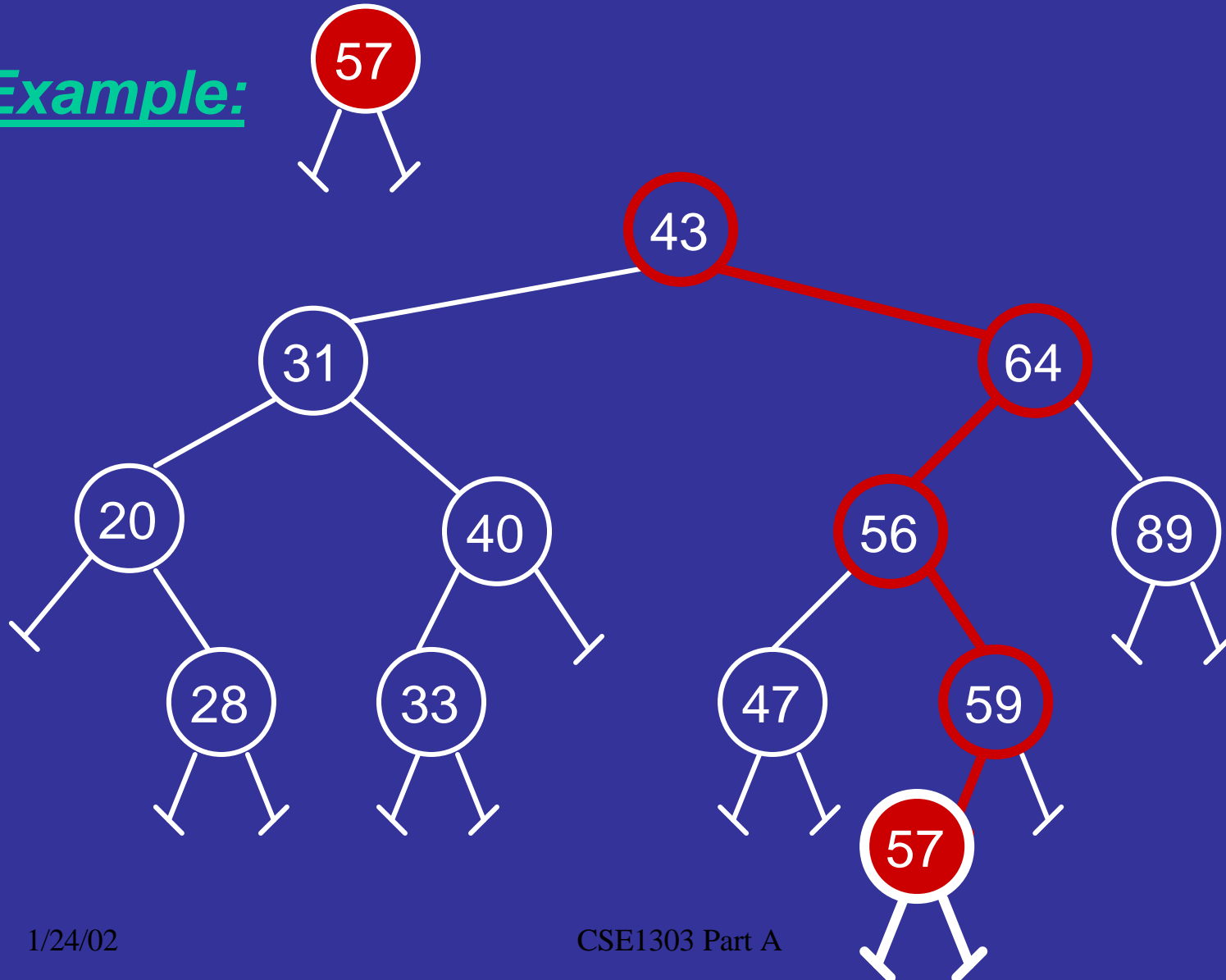
Insert

Example:



Insert

Example:



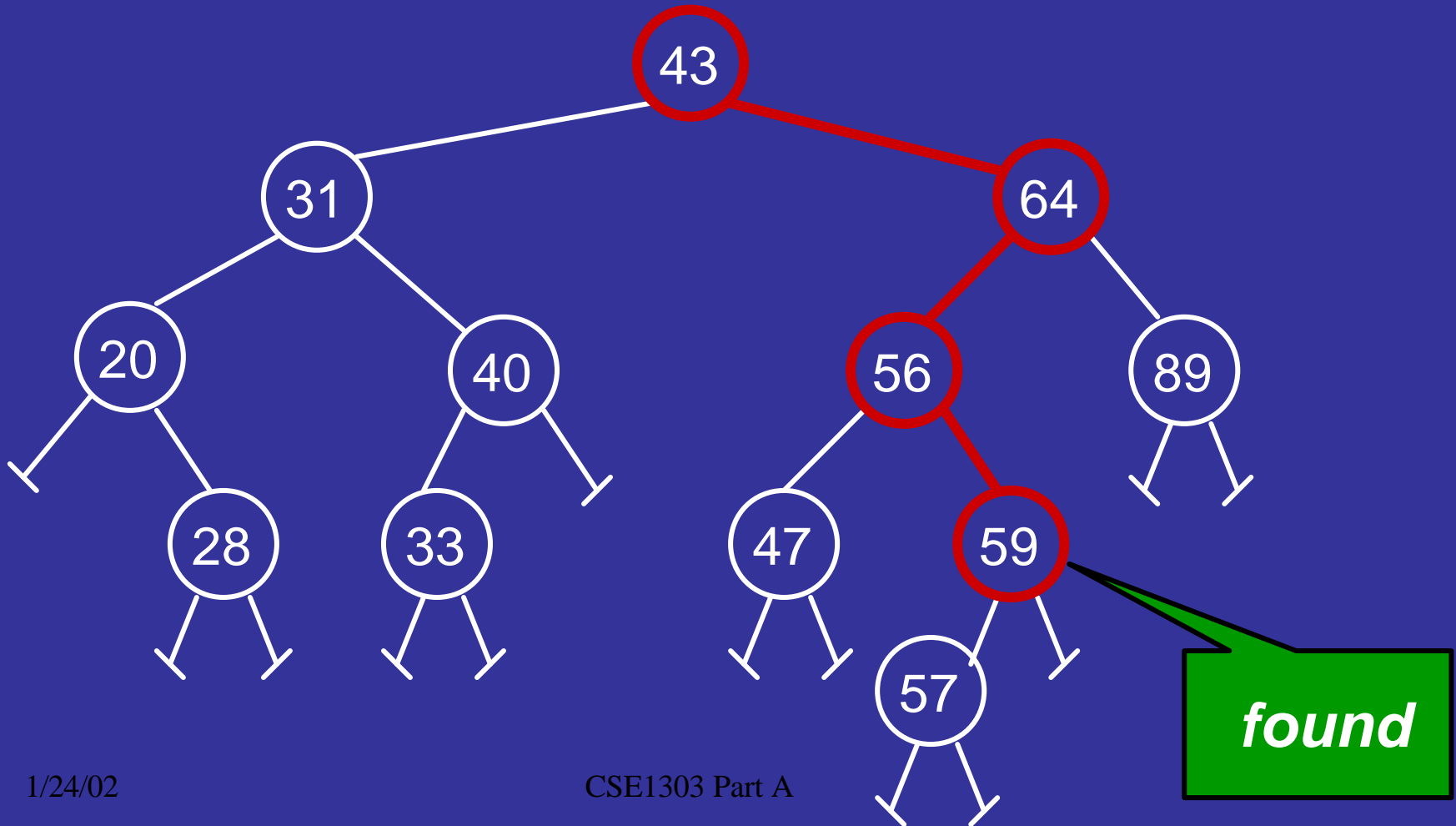
Search: Checklist

- if **target key** is less than current node's key, search the **left sub-tree**.
- else, if **target key** is **greater** than current node's key, search the **right sub-tree**.
- returns:
 - if found, **pointer** to node containing target key.
 - otherwise, **NULL** pointer.

Search

Example:

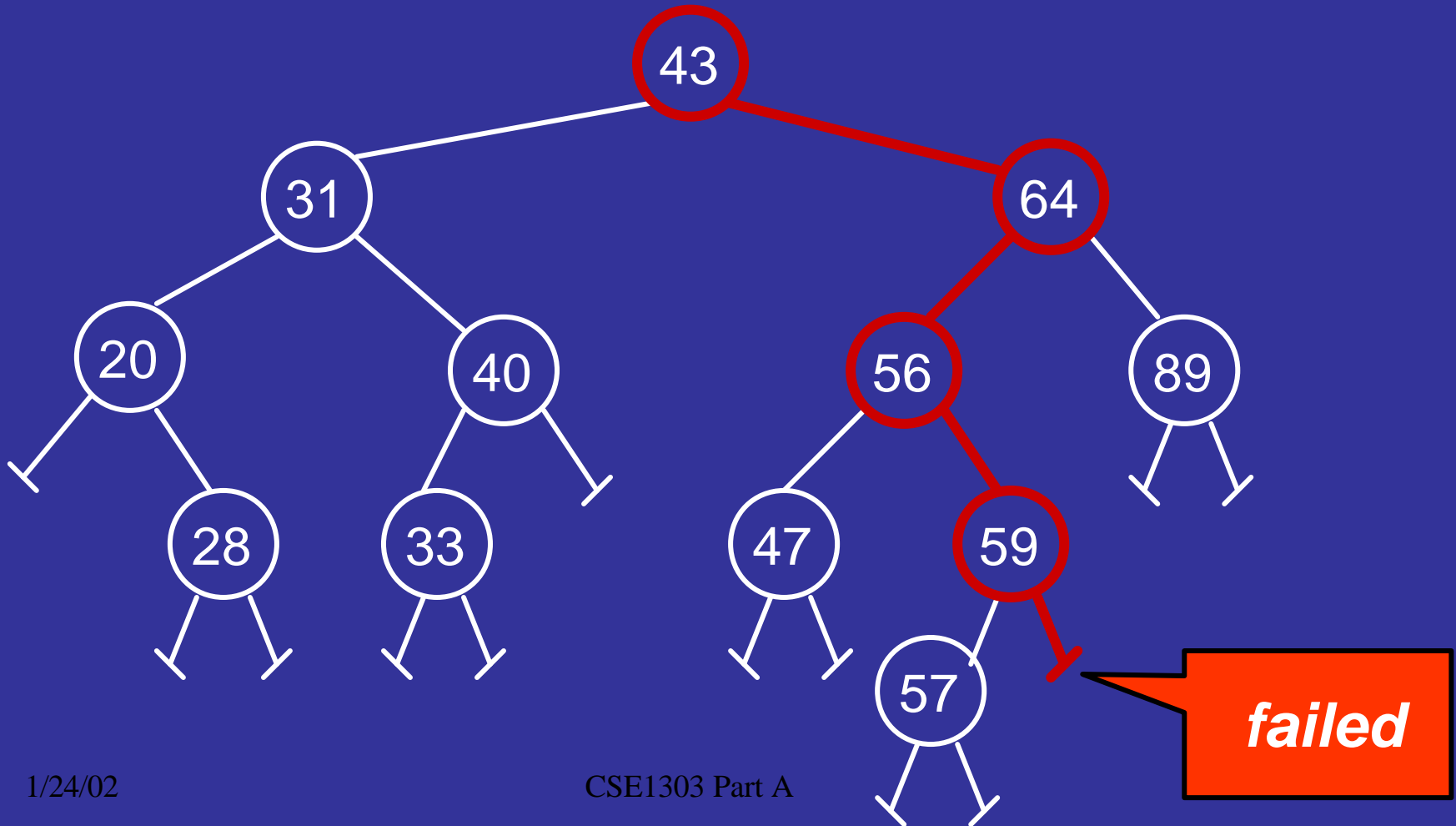
59



Search

Example:

61



Revision

- Binary Tree
- Preorder, Inorder, Postorder Traversal
- Expression Trees
- Prefix, Infix, and Postfix notation
- Insert and Search

Preparation

- Read Chapter 9.2 in Kruse et al.