**🔗 Types of Nodes in Data Structures**

**1. 🧱 Singly Linked List Node**

Each node contains:

* **Data**
* **Pointer to next node**

public class ListNode

{

public int Data;

public ListNode Next;

}

**2. 🔁 Doubly Linked List Node**

Each node contains:

* **Data**
* **Pointer to next node**
* **Pointer to previous node**

public class DoublyListNode

{

public int Data;

public DoublyListNode Prev;

public DoublyListNode Next;

}

**3. 🌲 Binary Tree Node**

Each node contains:

* **Data**
* **Left child**
* **Right child**

public class TreeNode

{

public int Value;

public TreeNode Left;

public TreeNode Right;

}

**4. 🌳 N-ary Tree Node**

Each node contains:

* **Data**
* **List of children**

public class NaryTreeNode

{

public int Value;

public List<NaryTreeNode> Children;

}

**5. 📍 Graph Node (Vertex)**

Each node represents:

* **Value or label**
* **List of connected neighbors (adjacency list)**

public class GraphNode

{

public int Value;

public List<GraphNode> Neighbors;

}

**6. 🔡 Trie Node (Prefix Tree)**

Each node contains:

* **Character (optional)**
* **Map of child nodes (Dictionary)**
* **Flag for end of word**

public class TrieNode

{

public bool IsEndOfWord;

public Dictionary<char, TrieNode> Children;

public TrieNode()

{

IsEndOfWord = false;

Children = new Dictionary<char, TrieNode>();

}

}

**🔄 Node Relationships (Terms)**

| **Term** | **Description** |
| --- | --- |
| **Parent** | Node that links to other nodes (tree) |
| **Child** | Node that is pointed to by a parent |
| **Leaf** | Node with no children (tree) |
| **Root** | Top node of a tree |
| **Sibling** | Nodes sharing the same parent |
| **Head/Tail** | First and last nodes in a list |
| **Neighbor** | Nodes directly connected in a graph |