**day15\_107856406\_dsdipt\_sudipto\_3july2025**

**Employee Code:** 107856406

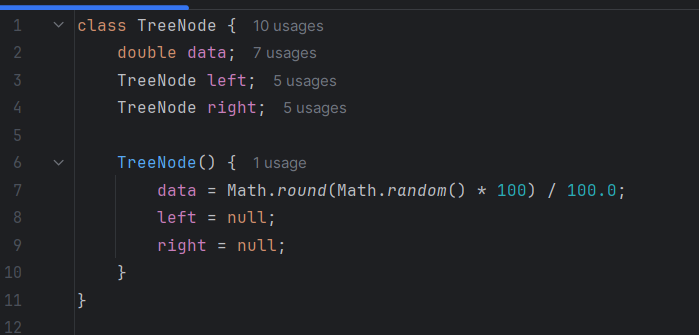
**Login ID:** dsdipt

**Email :** dsdipt@amazon.com

**Name:** Sudipto Das

**Date:** 3 July 2025 (Day 15)

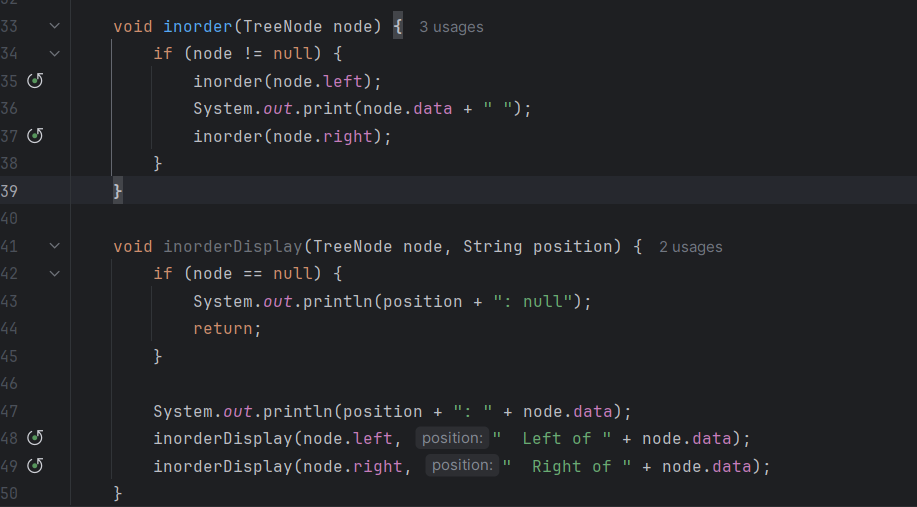
### ***Task 1: Create a node for a tree and include a constructor (empty)***



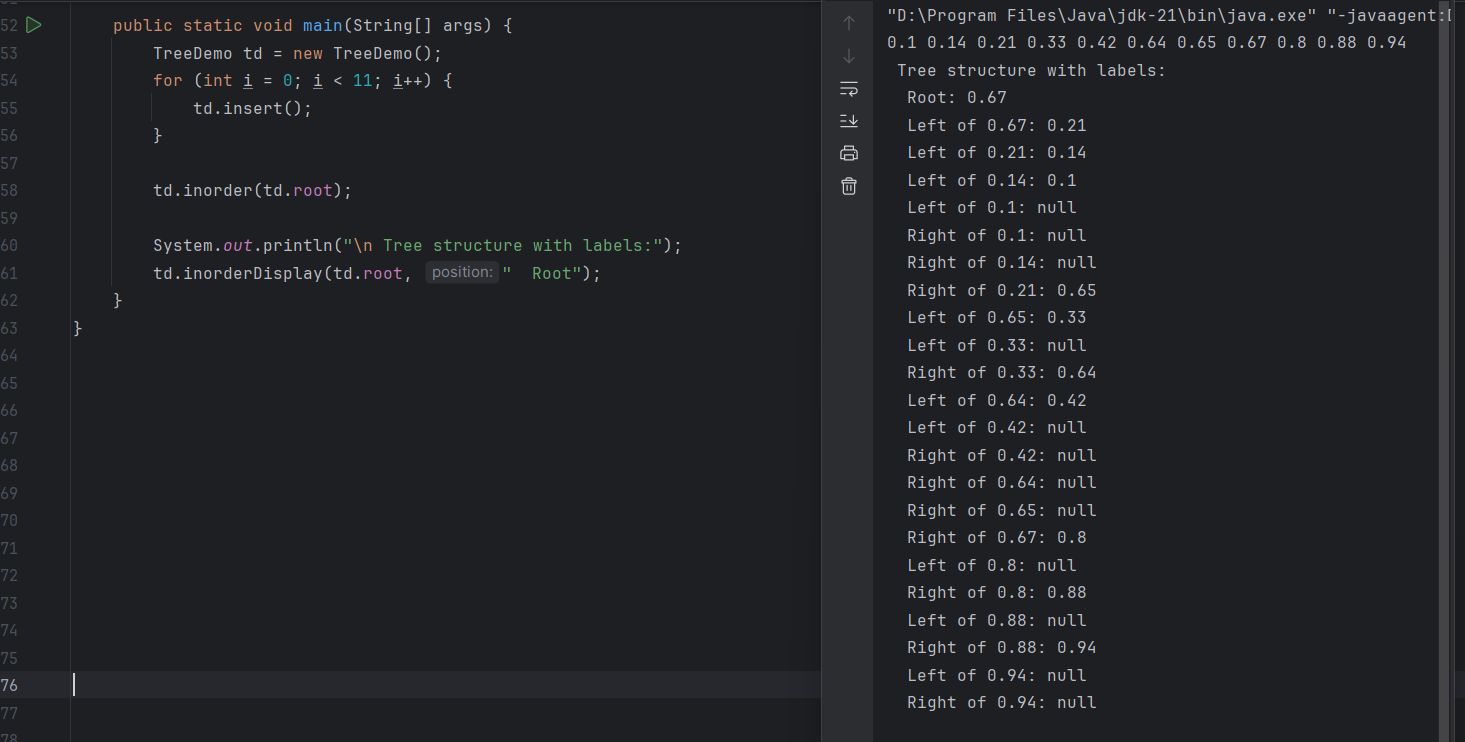
### ***Task 2: Create a tree in which you have 2 insert operations***



### ***Task 3: Inorder travel of the above code snippets from task 1 and Task 2***



### ***Task 4: Create a main method and check Task 1, 2 and 3***



### ***Task 5: Application of Tree***

**1. File System Organization**

* Use: Storing files and folders hierarchically.
* Example: Windows/Linux directory structures.
  + C:/Users/Sudipta/Documents/Project/

**2. Database Indexing (B-Trees, B+ Trees)**

* Use: Fast searching, inserting, deleting records in databases.
* Example: MySQL uses B+ Trees for indexing.

**3. Routing Algorithms (Trie Trees)**

* Use: Fast prefix matching.
* Example: IP routing in networks or autocomplete in search engines.

**4. Expression Evaluation (Binary Expression Trees)**

* Use: Parsing and evaluating mathematical expressions.
* Example: Converting infix to postfix and calculating (a + b) \* c.

**5. Decision Trees (AI/ML)**

* Use: Used in machine learning for classification problems.
* Example: Predicting loan approval based on income, age, etc.

**6. HTML/XML Parsing**

* Use: Document Object Model (DOM) is tree-based.
* Example: Browsers parse HTML as a tree to render web pages.

**7. Organizational Structures**

* Use: Represent hierarchy in companies.
* Example: CEO → Managers → Team Leads → Employees.

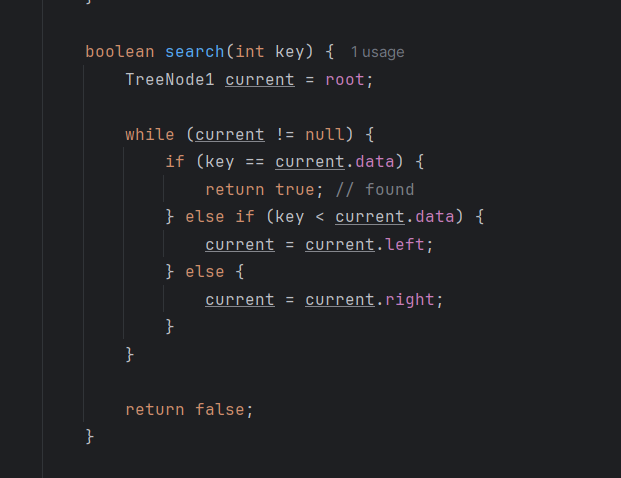
**8. Game Development (Minimax Trees)**

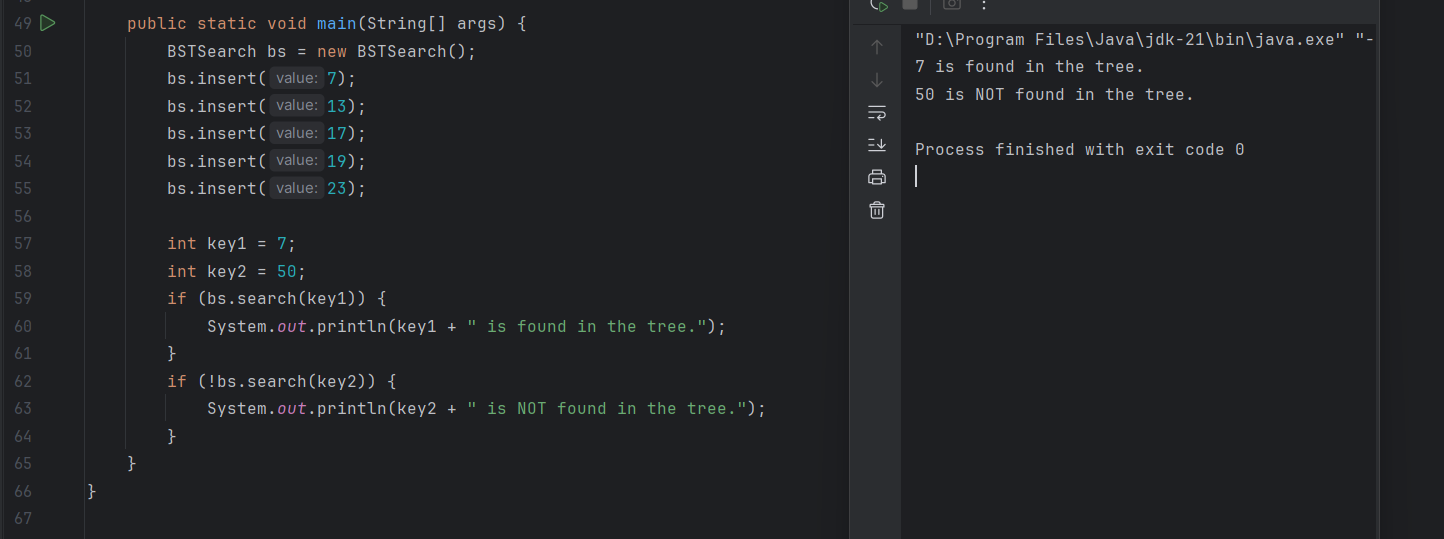
* Use: Making decisions in AI (chess, tic-tac-toe).
* Example: Minimax tree for choosing the best move.

**9. Compiler Design (Syntax Tree)**

* Use: Parsing source code to build Abstract Syntax Trees (AST).
* Example: Converting Java code to machine code.

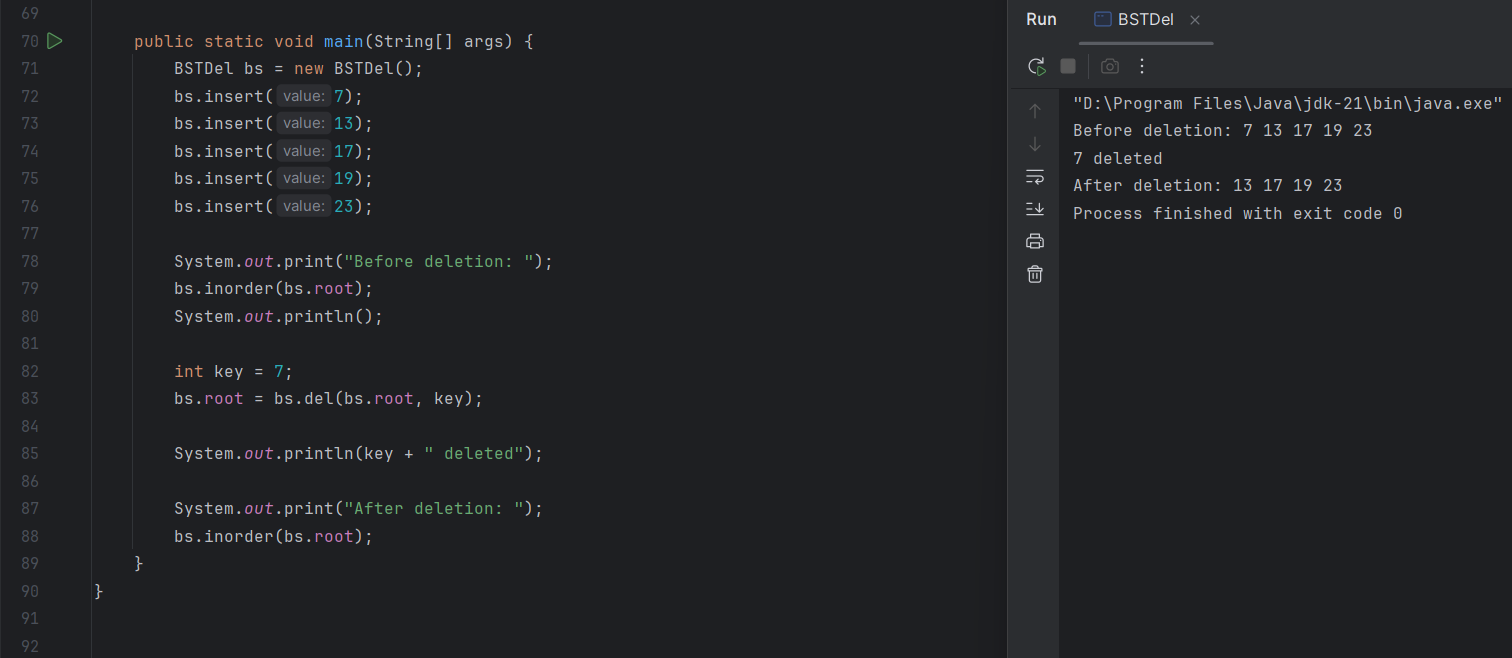
***Task 6: Create search operation on a BST***





***Task 7: Create delete operation on a BST***





***Task 8: Types of Binary tree***

**Full Binary Tree**  
 ➤ Every node has 0 or 2 children.

**Perfect Binary Tree**  
 ➤ All internal nodes have 2 children and all leaves are at the same level.

**Complete Binary Tree**  
 ➤ All levels are completely filled except possibly the last, which is filled left to right.

**Balanced Binary Tree**  
 ➤ Height of left and right subtrees differs by at most 1 for every node.

**Degenerate (or Skewed) Tree**  
 ➤ Each parent has only one child (like a linked list).

**Binary Search Tree (BST)**  
 ➤ Left child < parent < right child.

**AVL Tree**  
 ➤ A self-balancing BST (height-balanced).  
**Red-Black Tree**  
 ➤ A balanced BST with red/black coloring rules.

***Task 9: Applications of Graphs***

Graphs are powerful data structures used to model relationships and connections in real life. A graph consists of **nodes (vertices)** and **edges (connections)** and is widely used in various fields.

**1. Social Networks**

* In platforms like Facebook or Instagram, users are represented as nodes, and friendships or followers are the edges.
* Graph algorithms help suggest friends, find communities, and detect fake accounts.

**2. Google Maps / GPS Navigation**

* Cities or locations are nodes, and roads are edges.
* Algorithms like Dijkstra’s or A\* are used to find the shortest path from one place to another.

**3. Internet and Web Pages**

* Each web page is a node, and hyperlinks are the edges.
* PageRank Algorithm (used by Google) is based on graphs to rank search results.

**4. Recommendation Systems**

* Used in Amazon, Netflix, or Spotify.
* A graph connects users to products or movies they’ve liked, and similar connections are used to suggest new ones.

**5. Network Routing**

* In computer networks, routers are nodes and connections are edges.
* Graphs help in finding the most efficient data transfer paths and managing traffic.

**6. Biology and Chemistry**

* Used to model molecules (atoms as nodes, bonds as edges) and understand chemical reactions.
* In biology, food chains or neural networks are also graphs.

**7. Job Scheduling / Project Planning**

* Tasks are nodes and dependencies are edges.
* Topological sorting helps determine the order of tasks.

***Task 10: Types of Graphs***

**1. Directed vs Undirected Graph**

* Directed Graph (Digraph):  
   Edges have directions (like one-way streets).  
   → Example: Twitter (A follows B ≠ B follows A).
* Undirected Graph:  
   Edges don’t have direction.  
   → Example: Facebook friends (mutual connection).

**2. Weighted vs Unweighted Graph**

* Weighted Graph:  
   Each edge has a weight or cost (like distance or time).  
   → Example: Google Maps (roads have distances).
* Unweighted Graph:  
   All edges are equal — no weights.

**3. Cyclic vs Acyclic Graph**

* Cyclic Graph:  
   Contains at least one cycle (a path that forms a loop).
* Acyclic Graph:  
   No cycles.  
   → Example: Tree or task dependency graph.

**4. Connected vs Disconnected Graph**

* Connected Graph (Undirected):  
   There’s a path between every pair of vertices.
* Disconnected Graph:  
   Some nodes are not reachable from others.

**5. Complete Graph**

* Every node is connected to every other node.  
   → For n nodes, there are n(n-1)/2 edges.

**6. Bipartite Graph**

* Vertices can be divided into two sets, and edges only connect nodes from different sets.  
   → Example: Matching jobs to people.

**7. Tree and Forest**

* Tree: A connected acyclic graph.
* Forest: A collection of disjoint trees.

***Task 11: Wap to display a graph edges, no. of edges 8 and no. of vertex 5***

