152113022 VERİ YAPILARI LABORATUVARI

LAB WORK 9

03 December 2024

Objectives:

- Binary Tree Search, AVL

Question 1. This question aims to recall basic BST skills. Perform binary tree traversal with preorder, inorder and postorder structures. Define the operations as a menu. Below is an example of a menu.

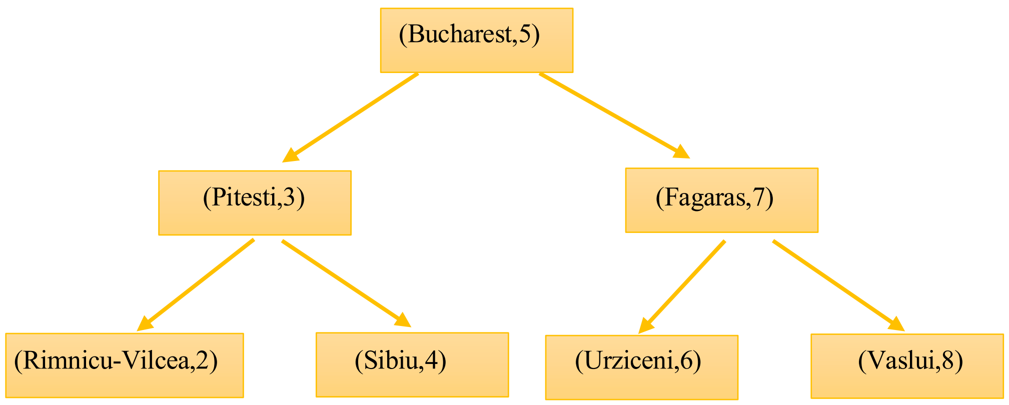
Preorder	**Inorder**	**Postorder**
▪ Process root R. ▪ Travel in the left subtree of R in preorder. ▪ Travel in the right subtree of R in preorder.	▪ Travel in left subtree of R in inorder. ▪ Process root R. ▪ Travel in right subtree of R in inorder.	▪ Travel in left subtree of R in postorder. ▪ Travel in right subtree of R in postorder. ▪ Process root R.

Output:

```
 1. Insert Elements
 2. Preorder
 3. Inorder
 4. Postorder
 5. Exit
 Enter Your Choice: 1

 Enter the Value: 5

 1. Insert Elements
 2. Preorder
 3. Inorder
 4. Postorder
 5. Exit
 Enter Your Choice: 1

 Enter the Value: 1

 1. Insert Elements
 2. Preorder
 3. Inorder
 4. Postorder
 5. Exit
 Enter Your Choice: 1

 Enter the Value: 3

1. Insert Elements
 2. Preorder
 3. Inorder
 4. Postorder
 5. Exit
 Enter Your Choice: 2
 5 1 3
```

Question 2. An X cargo company delivers to 7 cities during the day. Cities and local delivery times are given below in hours. Distribution is made from left to right, from minimum time to maximum time. Make the roadmap coding that gives the desired output for the given tree structure. The code should be in a generic structure. For example, it should give the desired output2 for input2.

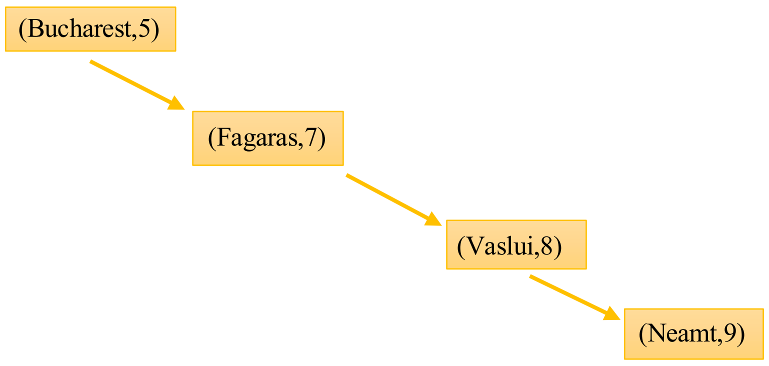
Input1:



Output3:

Rimnicu-Vilcea → Pitesti → Sibiu → Bucharest → Urziceni → Fagaras → Vaslui

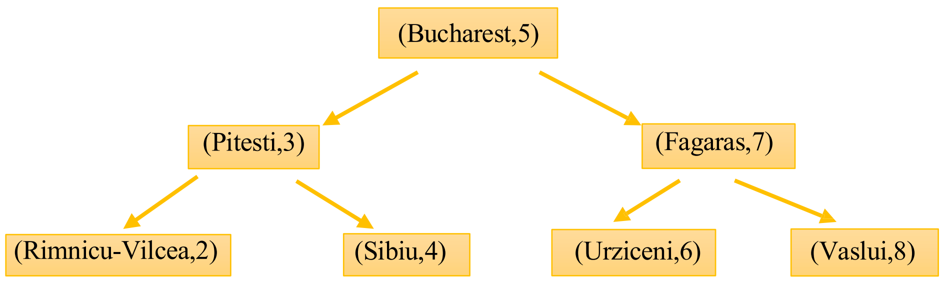
Input2:

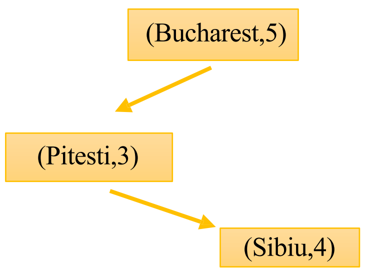
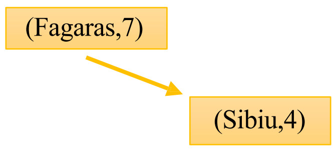


Output1:

Bucharest → Fagaras → Vaslui → Neamt

Question 3. The cargo delivery officer wants to check whether the route he has determined is on the main road map. Write the code that checks whether the entered tree is in the main path.



	
Output: This tree is subtree of input tree	**Output:** This tree is not subtree of input tree

Question 4.1. AVL tree is a self-balancing Binary Search Tree where the difference between heights of left and right subtrees cannot be more than one for all nodes. Create the following menu for the AVL tree. Perform necessary actions according to user selection.

" [1] Insert Element Into the Tree"

" [2] Show Balanced AVL Tree"

" [3] InOrder Traversal"

" [4] PreOrder Traversal"

" [5] PostOrder traversal"

" [6] Exit"

Question 4.1. Calculate the balance factor for question 4.1.

- Balance Factor = The height of the left tree - The height of the right tree.

Good Luck!