CS-371 Artificial Intelligence Minor

Assignment # 3

Title: Performing DFS, BFS and Pre-order traversal on a Tree

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY ISLAMABAD

Name	CMS
Muhammad Fateh Mehmood	369409

Submitted to: **Ma'am Ayesha Sarwar**

SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Workflow

1. First, I made a basic python code to implement a tree structure and print it.
2. I modified the code for the given tree and added BFS and DFS functions. Used these functions to iteratively search first for the starting country and then same function to search for the target country.
3. For pre-order traversal I changed the complete implementation to consider the left and right child separately.
4. Then I added matplotlib and networkx library to visualize the tree.
5. At last, I converted the whole code to C++ and made slight modifications to achieve the same functionality minus the visual representation.

Code Explanation

Explanation:

- The code implements a binary tree structure to represent countries and cities.
- It performs both breadth-first search (BFS) and depth-first search (DFS) to find specific countries within the tree structure.
- BFS and DFS are two common algorithms used for traversing or searching trees or graph structures.
- The tree is constructed with country nodes and city nodes, where each country node has left and right children representing neighboring countries and cities within the country.

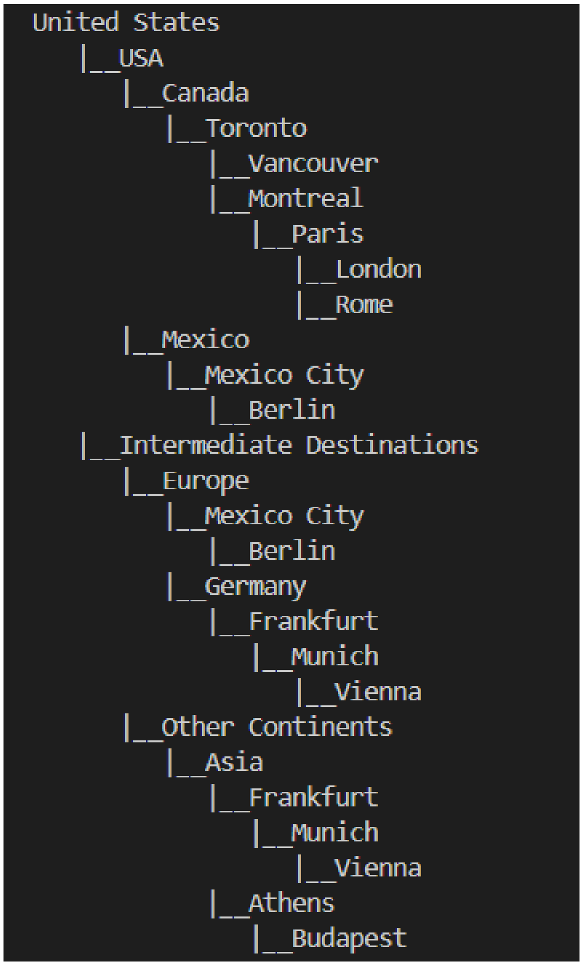
Functions:

- **TreeNode** class: Represents a node in the binary tree. It includes methods for adding left and right children, getting the level of the node, printing the tree, and performing a preorder traversal.
- **build_product_tree()**: Constructs the binary tree with country and city nodes.
- **bfs_search(TreeNode* root, string target)**: Performs a breadth-first search to find a target node in the tree.
- **dfs_search(TreeNode* node, string target)**: Performs a depth-first search to find a target node in the tree.
- **main()**: Entry point of the program where the tree is built, printed, and search algorithms are executed.

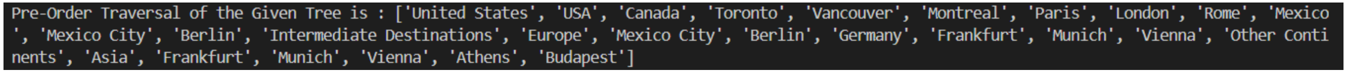
Data Flow:

- The program starts by building the binary tree representing countries and cities.
- The tree is printed to visualize its structure.
- The user is prompted to enter a starting country and a target country.

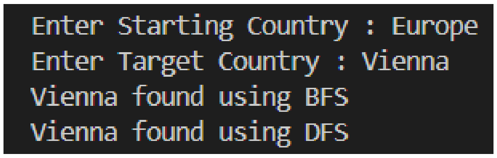
- BFS and DFS search algorithms are executed to find the target country starting from the specified country.
- The results of the search algorithms are printed to the console.



```
 United States
    |__USA
       |__Canada
          |__Toronto
             |__Vancouver
             |__Montreal
                |__Paris
                   |__London
                   |__Rome
       |__Mexico
          |__Mexico City
             |__Berlin
    |__Intermediate Destinations
       |__Europe
          |__Mexico City
             |__Berlin
          |__Germany
             |__Frankfurt
                |__Munich
                   |__Vienna
       |__Other Continents
          |__Asia
             |__Frankfurt
                |__Munich
                   |__Vienna
             |__Athens
                |__Budapest
```

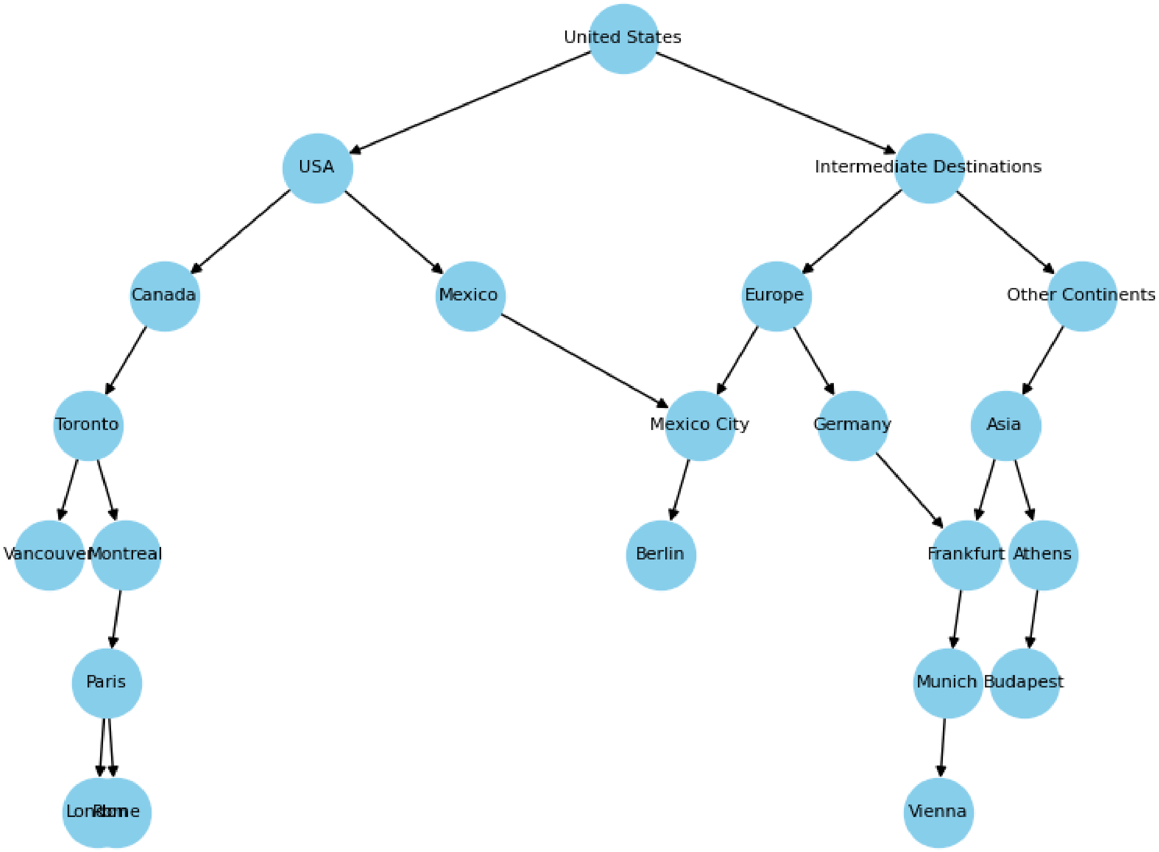


```
Pre-Order Traversal of the Given Tree is : ['United States', 'USA', 'Canada', 'Toronto', 'Vancouver', 'Montreal', 'Paris', 'London', 'Rome', 'Mexico
', 'Mexico City', 'Berlin', 'Intermediate Destinations', 'Europe', 'Mexico City', 'Berlin', 'Germany', 'Frankfurt', 'Munich', 'Vienna', 'Other Conti
nents', 'Asia', 'Frankfurt', 'Munich', 'Vienna', 'Athens', 'Budapest']
```



```
 Enter Starting Country : Europe
 Enter Target Country : Vienna
 Vienna found using BFS
 Vienna found using DFS
```

Implemented using **MatplotLib** and **NetworkX**


United States
USA
Intermediate Destinations
Canada
Mexico
Europe
Other Continents
Toronto
Mexico City
Germany
Asia
Vancouver
Montreal
Berlin
Frankfurt
Athens
Paris
Munich
Budapest
London
Rome
Vienna