Державний торговельно-економічний університет  
Кафедра інженерії програмного забезпечення та кібербезпеки

**ЛАБОРАТОРНА РОБОТА №8**

**З ДИСЦИПЛІНИ «АЛГОРИТМИ ТА СТРУКТУРИ ДАНИХ»**

**НА ТЕМУ «ДЕРЕВА»**

**Виконала**: студентка факультету

інформаційних технологій

групи\_курсу 3-4

Авєріна Наталія Ігорівна

**Перевірила**: Палагута К. О.

Київ 2024

**Лабораторна робота №8**

**Тема**: Дерева.

**Мета**: навчитись створювати програми для роботи з деревами на мові програмування Python.

**Програмне забезпечення**: PyCharm, MS Word.

**1 варіант**

**Хід роботи:**

1. Спочатку був створений клас BinaryTreeNode, який представляє вузол бінарного дерева. Клас містить атрибути value, left і right, що представляють значення вузла і вказівники на лівого та правого нащадків відповідно.

class BinaryTreeNode:  
 def \_\_init\_\_(self, value):  
 self.value = value  
 self.left = None  
 self.right = None

1. Використовуючи клас BinaryTreeNode, було створено бінарне дерево, яке відповідає варіанту завдання. Кожен вузол додається окремо, а його дочірні вузли встановлюються за допомогою атрибутів left і right.

Tree = BinaryTreeNode(8)  
Tree.left = BinaryTreeNode(3)  
Tree.right = BinaryTreeNode(10)  
Tree.left.left = BinaryTreeNode(1)  
Tree.left.right = BinaryTreeNode(6)  
Tree.right.right = BinaryTreeNode(14)  
Tree.left.right.left = BinaryTreeNode(4)  
Tree.left.right.right = BinaryTreeNode(7)  
Tree.right.right.left = BinaryTreeNode(13)

1. Була створена функція dfs(node), яка використовує рекурсію для обходу дерева в глибину. Вона виводить значення поточного вузла, а потім рекурсивно викликає себе для лівої та правої гілки дерева.

def dfs(node):  
 if node:  
 print(node.value, end=', ')  
 dfs(node.left)  
 dfs(node.right)

1. Після цього була написана функція bfs(node), яка використовує чергу для обходу дерева в ширину. Вона виводить значення поточного вузла, а потім додає його дочірні вузли в чергу для подальшого обходу.

def bfs(node):  
 if not node:  
 return  
  
 queue = [node]  
 while queue:  
 node = queue.pop(0)  
 print(node.value, end=', ')  
 if node.left:  
 queue.append(node.left)  
 if node.right:  
 queue.append(node.right)

1. Обидва методи обходу, DFS і BFS, викликаються для створеного дерева Tree, щоб вивести послідовності значень відповідно до обраних алгоритмів.

print('DFS: ', end='')  
dfs(Tree)

print('\nBFS: ', end='')  
bfs(Tree)

**Висновок**: під час виконання даної лабораторної роботи було набуто практичних навичок роботи з деревами. Було створено програму реалізації дерева у вигляді рекурсивної структури та здійснено обхід дерева в глибину та ширину, виведено на друк перелік вершин.