Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни

«Основи програмування-2.

Методології програмування»

«Дерева»

Варіант 3

Виконав студент ІП-11 Веремчук Ігор Ігорович

(шифр, прізвище, ім'я, по батькові)

Перевірив

( прізвище, ім'я, по батькові)

Київ 2022

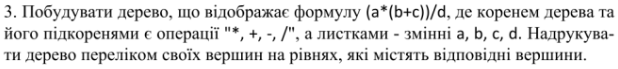
**Лабораторна робота №6**

**Дерева**

**Мета**

****

**Завдання(варіант 3):**



**Постановка задачі**

Для виконання поставленої задачі створимо клас “Node”, який представляє вузли дерева та “FormulaTree ”, який представляє дерево, що відображає формулу, кожен з них помістимо в окремий файл.

**Програма на С#**

Файл 1:

using System;  
using System.Linq;  
using System.Collections.Generic;  
using System.Text;  
using System.Threading.Tasks;  
  
  
namespace Lab6Sharp  
{  
 internal static class Program  
 {  
 static void Main(string[] args)  
 {  
 int[] nums = GetConsoleInput();  
 FormulaTree tree = new FormulaTree('/');  
 object[] values = {nums[3], '\*', nums[0], '+', nums[2], nums[1]};  
  
 foreach (object value in values)  
 {  
 tree.Insert(value);  
 }  
  
 Console.WriteLine($"\nLevel Traverse\n{tree.LevelTraverse()}");  
 Console.WriteLine($"Preorder Traverse\n{tree.PreorderTraverse()}");  
 Console.WriteLine($"Inorder Traverse\n{tree.InorderTraverse()}");  
 Console.WriteLine($"Postorder Traverse\n{tree.PostorderTraverse()}");  
  
 Console.ReadLine();  
 }  
  
 static int[] GetConsoleInput()  
 {  
 Console.WriteLine("Enter 4 numbers in formula (a\*(b+c))/d:");  
 int[] nums = new int[4];  
 for (int i = 0; i < nums.Length; i++)  
 {  
 string? input = Console.ReadLine();  
 while (!int.TryParse(input, out \_))  
 {  
 Console.WriteLine("Wrong input format");  
 input = Console.ReadLine();  
 }  
  
 nums[i] = int.Parse(input);  
 }  
  
 return nums;  
 }  
 }  
}

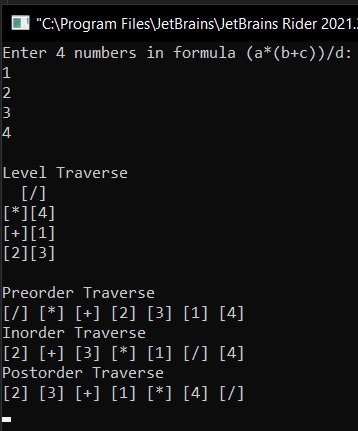
Файл 2:

using System;  
using System.Linq;  
using System.Collections.Generic;  
using System.Text;  
using System.Threading.Tasks;  
  
namespace Lab6Sharp  
{  
 internal class FormulaTree  
 {  
 public Node Root { get; private set; }  
  
 public FormulaTree(object value) => Root = new Node(value);  
 public FormulaTree(Node node) => Root = node;  
  
 public void Insert(object value) => InsertHelper(value, Root);  
 public void Insert(Node node) => InsertHelper(node, Root);  
  
 public string LevelTraverse() =>  
 LevelTraverseHelper(Root, new StringBuilder(Root.ToString().PadLeft(5) + "\n"));  
  
 public string PreorderTraverse() => PreorderTraverseHelper(Root, new List<Node>());  
 public string InorderTraverse() => InorderTraverseHelper(Root, new List<Node>());  
 public string PostorderTraverse() => PostorderTraverseHelper(Root, new List<Node>());  
  
 private void InsertHelper(object value, Node root)  
 {  
 if (root.Right == null) root.Right = new Node(value);  
 else if (root.Left == null) root.Left = new Node(value);  
 else InsertHelper(value, root.Left);  
 }  
  
 private void InsertHelper(Node node, Node root)  
 {  
 if (root.Right == null) root.Right = node;  
 else if (root.Left == null) root.Left = node;  
 else InsertHelper(node, root.Left);  
 }  
  
 private string LevelTraverseHelper(Node root, StringBuilder result)  
 {  
 if (root.Left != null) result.Append(root.Left);  
 if (root.Right != null) result.Append(root.Right + "\n");  
 if (root.Left != null) LevelTraverseHelper(root.Left, result);  
 return result.ToString();  
 }  
  
 private string PreorderTraverseHelper(Node root, List<Node> nodes)  
 {  
 nodes.Add(root);  
 if (root.Left != null) PreorderTraverseHelper(root.Left, nodes);  
 if (root.Right != null) PreorderTraverseHelper(root.Right, nodes);  
 return string.Join(' ', nodes);  
 }  
  
 private string PostorderTraverseHelper(Node root, List<Node> nodes)  
 {  
 if (root.Left != null) PostorderTraverseHelper(root.Left, nodes);  
 if (root.Right != null) PostorderTraverseHelper(root.Right, nodes);  
 nodes.Add(root);  
 return string.Join(' ', nodes);  
 }  
  
 private string InorderTraverseHelper(Node root, List<Node> nodes)  
 {  
 if (root.Left != null) InorderTraverseHelper(root.Left, nodes);  
 nodes.Add(root);  
 if (root.Right != null) InorderTraverseHelper(root.Right, nodes);  
 return string.Join(' ', nodes);  
 }  
 }  
}

Файл 3:

namespace Lab6Sharp  
{  
 internal class Node  
 {  
 public Node(object value)  
 {  
 Value = value;  
 Left = Right = null;  
 }  
  
 public object Value { get; private set; }  
  
 public Node? Left { get; set; }  
  
 public Node? Right { get; set; }  
  
 public override string ToString() => $"[{Value}]";  
 }  
}

**Виконання на C#**



**Висновок:**

На лабораторній роботі я опанував технологію написання власних структур даних, і опанував особливості організації та обробки дерев, виконавши поставлене завдання на мові програмування C#.