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-14 Басан Антон Сергійович

(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна

( прізвище, ім'я, по батькові)

Київ 202 2

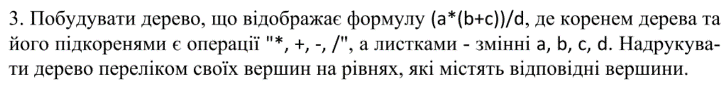
**Лабораторна робота 6**

**Дерева**

*Варіант 3*

*Мета:* вивчити особливості організації і обробки дерев.

*Задача:*



**Розв’язання**

*Постановка задачі*

Результатом розв’язку задачі є дерево, що відображає формулу (a\*(b+c))/d. Побудова дерева відбуватиметься за допомогою рекурсивних методів та об’єктів створеного класу Node.

*Код та його випробування (С#)*

Program.cs

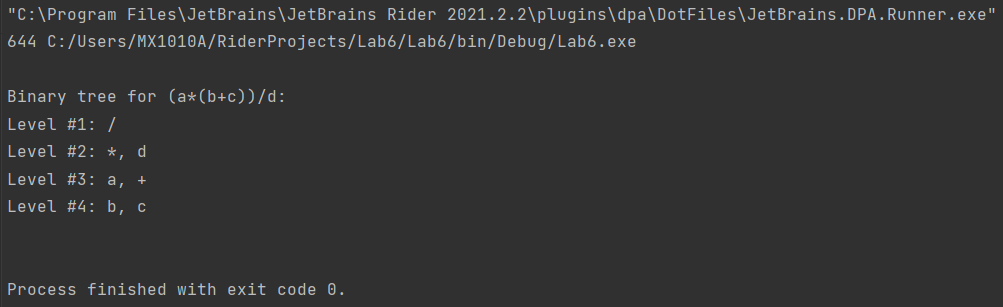
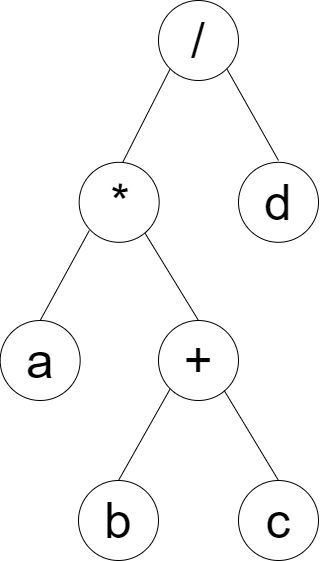
using System;  
using static Lab6.Func;  
  
namespace Lab6  
{  
 internal static class Program  
 {  
 static void Main(string[] args)  
 {  
 string expression = "(a\*(b+c))/d";  
 Node root = GenerateTree(expression);  
 Console.WriteLine($"\nBinary tree for {expression}:");  
 Console.WriteLine($"Level #1: {root}\n{String(root, 2)}");  
 }  
 }  
}

Func.cs

using System;  
  
namespace Lab6  
{  
 public static class Func  
 {  
 public static Node GenerateTree(string expression)  
 {  
 (string expLeft, char opr, string expRight) = Parse(expression);  
 Node root = new Node(opr);  
  
 if (expLeft.Length > 1)   
 root.Left = GenerateTree(expLeft);  
 else root.Left = new Node(expLeft[0]);   
  
 if (expRight.Length > 1)   
 root.Right = GenerateTree(expRight);  
 else root.Right = new Node(expRight[0]);  
   
 return root;  
 }  
   
 public static string String(Node root, int level)  
 {  
 string result = "";  
 if (root.Left != null) result += root.Left + ", ";  
 if (root.Right != null)  
 {  
 result += root.Right + "\n";  
 result = result.Insert(0, $"Level #{level}: ");  
 }  
 level++;  
 if (root.Left != null) result += String(root.Left, level);  
 if (root.Right != null) result += String(root.Right, level);  
   
 return result;  
 }  
   
 private static bool IsOperator(char ch) => ch is '/' or '\*' or '-' or '+';  
   
 private static (string, char, string) Parse(string expression)   
 {  
 int inExp = 0, oprInd = 0;  
 char opr = ' ';  
 string expLeft = "", expRight = "";  
 for(int i = 0; i < expression.Length; i++)  
 {  
 if (expression[i] == ')') inExp--;  
 if (inExp > 0 && oprInd == 0) expLeft += expression[i];  
 else if (inExp > 0 && i > oprInd) expRight += expression[i];  
 if (expression[i] == '(') inExp++;  
   
 if (inExp != 0 || !IsOperator(expression[i])) continue;  
 opr = expression[i];  
 oprInd = i;  
   
 if (oprInd > 0 && Char.IsLetter(expression[oprInd - 1])) expLeft += expression[oprInd - 1];  
 if (oprInd > 0 && Char.IsLetter(expression[oprInd + 1])) expRight += expression[oprInd + 1];  
 }  
   
 return (expLeft, opr, expRight);  
 }  
 }  
}

Node.cs

#nullable enable  
  
namespace Lab6  
{  
 public class Node  
 {  
 public Node(char value)  
 {  
 Value = value;  
 }  
   
 private char Value { get; }  
 public Node? Left { get; set; }  
 public Node? Right { get; set; }  
 public override string ToString() => $"{Value}";  
 }  
}

** 

***Висновок:*** під час виконання завдання були досліджені особливості організації і обробки дерев, крім того були набуті практичні навички розробки програм, які працюють з деревами (а саме програми для побудови дерева для математичного виразу (a\*(b+c))/d) на компільованій мові С#. Результати програми та власноручних обчислень збігаються.