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 № 5 з дисципліни

«Основи програмування - 2»

«Дерева»

Варіант 4

Виконав: Бондаренко М.В. ІП-13

Перевірила: Вечерковська А.С.

Київ 2022

**Лабораторна робота 4**

**Тема:** дерева.

**Постановка задачі**

****

**Код C#:**

**Program.cs:**

*using* System;  
  
*namespace* lab5  
{  
 *internal class* Program  
 {  
 *public static void* Main(*string*[] args)  
 {  
 *int* positive\_counter = 0;  
 *int* negative\_counter = 0;  
   
 Tree tree = *new* Tree();  
 Console.Write("Enter amount of elemets: ");  
 *int* size = Convert.ToInt32(Console.ReadLine());  
 *for* (*int* i = 0; i < size; i++)  
 {  
 Console.Write("Enter element: ");  
 *int* num = Convert.ToInt32(Console.ReadLine());  
 tree.AddNode(num);  
 }  
 tree.PrintTree();  
 tree.FindElements(*ref* positive\_counter, *ref* negative\_counter);  
 Console.WriteLine($"Positive elements: {positive\_counter}, negative elements: {negative\_counter}");  
 }  
 }  
}

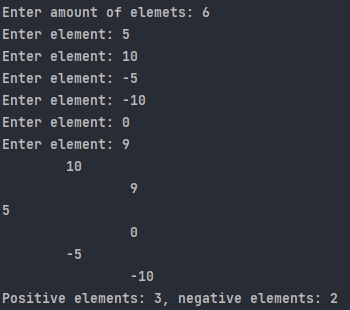
**Node.cs:**

*namespace* lab5  
{  
 *public class* Node  
 {  
 *public int* Num { get; set; }  
 *public* Node Right;  
 *public* Node Left;  
  
 *public* Node(*int* value)  
 {  
 Num = value;  
 Right = *null*;  
 Left = *null*;  
 }  
 }  
}

**Tree.cs:**

*using* System;  
  
*namespace* lab5  
{  
 *public class* Tree  
 {  
 *private* Node Root;  
  
 *public* Tree()  
 {  
 Root = *null*;  
 }  
  
 *public void* AddNode(*int* num)  
 {  
 AddNodeRecursion(num, *ref* Root);  
 }  
  
 *private void* AddNodeRecursion(*int* num, *ref* Node parent)  
 {  
 *if* (parent == *null*) parent = *new* Node(num);  
 *else* {  
 *if* (parent.Num < num)  
 {  
 AddNodeRecursion(num, *ref* parent.Right);  
 }  
 *else if* (parent.Num > num)  
 {  
 AddNodeRecursion(num, *ref* parent.Left);  
 }  
 }  
 }  
  
 *public void* FindElements(*ref int* positive, *ref int* negative)  
 {  
 FindElementsRecursion(*ref* Root, *ref* positive, *ref* negative);  
 }  
  
 *private void* FindElementsRecursion(*ref* Node parent, *ref int* positive, *ref int* negative)  
 {  
 *if* (parent == *null*) *return*;  
 FindElementsRecursion(*ref* parent.Left, *ref* positive, *ref* negative);  
 FindElementsRecursion(*ref* parent.Right, *ref* positive, *ref* negative);  
   
 *if* (parent.Num > 0) positive++;  
 *if* (parent.Num < 0) negative++;  
 }  
  
 *public void* PrintTree()  
 {  
 PrintTreeRecursion(*ref* Root, 0);  
 }  
   
 *private void* PrintTreeRecursion(*ref* Node parent, *int* height)  
 {  
 *if* (parent == *null*) *return*;  
   
 PrintTreeRecursion(*ref* parent.Right, ++height);  
  
 *for* (*int* i = 1; i < height; i++)  
 {  
 Console.Write("\t");  
 }  
  
 Console.WriteLine(parent.Num);  
 PrintTreeRecursion(*ref* parent.Left, height);  
 }  
 }  
}

**Console:**

****