БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра Информационных технологий

Факультет Компьютерных технологий

Специальность Программное обеспечение информационных технологий

Лабораторная работа № 9

по дисциплине «Структуры и алгоритмы обработки данных»

Вариант № 14

Выполнил студент: Шичко Павел Викторович

группа 981064

Минск 2020

Задание №1:

Выполнить симметричную прошивку бинарного дерева поиска. Обойти его согласно симметричному порядку следования элементов. Реализовать вставку и удаление элементов из симметрично прошитого бинарного дерева.

Решение:

class Program  
{  
 static void Main(string[] args)  
 {  
 var tree = new PrTree<int>();  
 tree = Ent(tree);  
 var input = "";  
 while (input != "5")  
 {  
 Console.WriteLine("Action: 1)add 2)remove 3)search 4)show 5)exit");  
 input = Console.ReadLine();  
 switch (input)  
 {  
 case "1":  
 try  
 {  
 Console.WriteLine("Enter data:");  
 var data = Convert.ToInt32(Console.ReadLine());  
 tree.Add(data);  
 }  
 catch (Exception e)  
 {  
 Console.WriteLine(e.Message);  
 }  
 break;  
 case "2":  
 try  
 {  
 Console.WriteLine("Enter data:");  
 var data = Convert.ToInt32(Console.ReadLine());  
 tree.Remove(data);  
 }  
 catch (Exception e)  
 {  
 Console.WriteLine(e.Message);  
 }  
 break;  
 case "3":  
 try  
 {  
 Console.WriteLine("Enter data:");  
 var data = Convert.ToInt32(Console.ReadLine());  
 Console.WriteLine(tree.Search(data));  
 }  
 catch (Exception e)  
 {  
 Console.WriteLine(e.Message);  
 }  
 break;  
 case "4":  
 Console.WriteLine("Sim:");  
 Console.WriteLine(tree.Show());  
 break;  
 case "5":  
 break;  
 default:  
 input = "";  
 break;  
 }  
 }  
 }  
  
 private const int **maxN** = 4236;  
 private const int **minN** = 1;  
  
 static int[] ManEnt(int count)  
 {  
 var input = new int[count];  
 for (var i = 0; i < count; i++)  
 {  
 Console.Write("Number {0}:", i + 1);  
 var val = Console.ReadLine();  
 if (int.TryParse(val, out var number))  
 {  
 if (number > **maxN** || number < **minN**)  
 {  
 Console.WriteLine("Number must be in range {0}-{1}",  
 **minN**, **maxN**);  
 i--;  
 }  
 else  
 input[i] = number;  
 }  
 else  
 {  
 Console.WriteLine("Attempted conversion of '{0}' to int failed.",  
 val ?? "<null>");  
 i--;  
 }  
 }  
 return input;  
 }  
  
 static int[] RandEnt(int count)  
 {  
 var input = new int[count];  
 var rnd = new Random();  
 for (var i = 0; i < count; i++)  
 {  
 Console.Write("Number {0}:", i + 1);  
 var number = rnd.Next(**minN**, **maxN**);  
 Console.WriteLine(number);  
 input[i] = number;  
 }  
 return input;  
 }  
  
 static PrTree<int> Ent(PrTree<int> tree)  
 {  
 const int **n** = 15;  
 int[] input;  
 while (true)  
 {  
 Console.Write("Use Random? (Y/N)");  
 var val = Console.ReadLine();  
 if (val == "Y" || val == "y")  
 {  
 input = RandEnt(**n**);  
 break;  
 }  
 if (val == "N" || val == "n")  
 {  
 input = ManEnt(**n**);  
 break;  
 }  
 }  
 foreach (var number in input)  
 tree.Add(number);  
 return tree;  
 }  
}

