МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Алгоритмы и структуры данных»

Тема: Бинарные деревья

Студентка гр. 7383 _____ Ханова Ю.А.

Преподаватель _____ Размочаева Н.В

Санкт-Петербург

2018

Содержание

Цель работы 3
Реализация задачи 3
Тестирование 4
Выводы 4
Приложение А. Код программы 5
Приложение Б. Тестовые случаи 7

Цель работы

Познакомиться с бинарными деревьями и научиться реализовывать их на языке программирования C++.

Для заданного бинарного дерева b типа BT:

а) определить максимальную глубину дерева b, т. е. число ветвей в самом длинном из путей от корня дерева до листьев;

б) вычислить длину внутреннего пути дерева b, т. е. сумму по всем узлам длин путей от корня до узла;

в) напечатать элементы из всех листьев дерева b;

г) подсчитать число узлов на заданном уровне n дерева b (корень считать узлом 1-го уровня);

Вариант 4-в.

Реализация задачи

В данной лабораторной работе была реализована класс node, набор функций для работы с бинарным деревом:

char RootBT(int b); - Возвращает элемент, содержащийся в данном корне дерева;

int Left(int b); - возвращает левое поддерево;

int Right(int b); - возвращает правое поддерево;

int freeND(); - обнуляет элементы массива;

int ConsBT; - объединяет деревья;

Так же были разработаны функции для преобразования данных в соответствии с заданием:

void displayBT(int b, int n); - выводит дерево на экран;

int build(string a, string b, int x); - строит дерево по двум заданным записям (ЛКП и ЛПК): находит последний элемент в массиве с ЛКП (он

будет корнем всего дерева) затем ищет соответствующий элемент в ЛКП и рекурсивно вызывает себя для левого и правого поддерева и оставшихся частей строк;

Тестирование

Программа собрана в операционной системе Ubuntu 17.04 с использованием компилятора g++. В других ОС и компиляторах тестирование не проводилось. Результаты тестирования показали, что поставленная цель выполнена. Результаты тестирования представлены в Приложении Б.

Выводы

В ходе выполнения данной лабораторной работы были освоены основные принципы работы с бинарными деревьями на языке программирования C++. Также была написана программа для выполнения поставленной задачи.

ПРИЛОЖЕНИЕ А. Код программы

binTree.h

```
#pragma once
namespace binTree_modul
{

	struct node {
		char info;
		int lt;
		int rt;
		node(char inputinfo= 0, int inputlt = 0, int inputrt = 0) {
			info = inputinfo;
			lt=inputlt;
			rt=inputrt;
		};
	};
	bool isNull(int);
	char RootBT(int);
	int Left(int);
	int Right(int);
	int freeND();
	int ConsBT(const char &x, int lt, int rt);

}
```

binTree.cpp

```
#include <iostream>
#include <cstdlib>
#include "binTree.h"
using namespace std;

namespace binTree_modul
{
	node* Mem = new node;
	bool isNull(int b){
		return (b == 0);
	}
	char RootBT(int b){
		if (b == 0) {
			cerr << "Error: RootBT(null) \n";
			system("pause > nul");
		}
		else return Mem[b].info;
	}
	int Left(int b){
		if (b == 0) {
			cerr << "Error: Left(null) \n";
			system("pause > nul");
		}
```

```
        else return Mem[b].lt;
    }
    int Right(int b){
        if (b == 0) {
            cerr << "Error: Right(null) \n";
            system("pause > nul");
        }
        else return Mem[b].rt;
    }

    int freeND(){
        for(int i=1; i<100; i++){
            if(Mem[i].info == 0) return i;
        }
        return -1;
    }
    int ConsBT(const char &x, int lt, int rt){
        int k = freeND();
        if (k > 0) {
            Mem[k].info = x;
            Mem[k].lt = lt;
            Mem[k].rt = rt;
            return k;
        }
        else { cerr << "Memory not enough\n"; exit(1); }
    }
}
```

Source.cpp

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include "binTree.h"
#include <string>
using namespace std;
using namespace binTree_modul;
void displayBT(int b, int n)
{    // n  — уровень узла
    if (b != 0) {
        cout << ' ' << RootBT(b);
        if (!isNull(Right(b))) { displayBT(Right(b), n + 1); }
        else cout << endl; // вниз
        if (!isNull(Left(b))) {
            for (int i = 1; i <= n; i++) cout << "  "; // вправо
            displayBT(Left(b), n + 1);
        }
    }
    else {};
}
```

```
int build(string a, string b, int x) {
	int k;
	if (a == "") return 0;
	if (a.length() == 1) {
		k = ConsBT(a[0], 0, 0);
		return k;
	}
	if (x < 0) return 0;
	int flag = a.find(b[x]);
	if (flag < 0) {cout << "Error: некорректные данные!"<< endl;return 0;
}
	k = ConsBT(b[x], build(a.substr(0, flag), b.substr(0, flag), flag-1),
build(a.substr(flag + 1), b.substr(flag, x-1), x-1-flag));
	return k;
}

int main() {
	int ch;
	bool ex = true;
	string str1, str2;
	while(ex){
		cout << "Введите номер действия:"<< endl;
		cout << "0 - выход"<< endl;
		cout << "1 - ввод с клавиатуры"<< endl;
		cout << "2 - ввод из файла"<< endl;
		cin >> ch;
		switch(ch){
			case 0: {
				ex = false;
				break;
			}
			case 1:{
				cout<<"Введите ЛКП запись:"<< endl;
				cin >> str1;
				cout<<"Введите ЛПК запись:"<< endl;
				cin >> str2;
				if(str1.length()!= str2.length()){cout << "Error:
строки разной длины!"<< endl;}
				else{
					int MyTree = build(str1, str2, str2.length() -
1);
					displayBT(MyTree, 1);
				}
				break;
			}
			case 2:{
				ifstream fin("input.txt");
				if (!fin) { cout << "File not open for reading!\n";
return 1; }
				cout << "Считывается ЛКП запись..." << endl;
				getline(fin, str1, '\n');
				cout << "Считывается ЛПК запись..." << endl;
```

```
                        getline(fin, str2, '\n');
                        if(str1.length()!= str2.length()){cout << "Error:
строки разной длины!"<< endl;}
                        else{
                              int MyTree = build(str1, str2, str2.length() -
1);
                              displayBT(MyTree, 1);
                        }
                        break;
                  }
                  default:{
                        ex = false;
                        cout << "Error of input!";
                        break;
                  }
            }
      }
      return 0;
}
```

ПРИЛОЖЕНИЕ Б. Тестовые случаи

Таблица 1 - Результаты тестов.

input	output	True/false
DBZEAGFXC DZEBGXFCA	A C F X G B E Z D	True
ABC ACB	B C A	True
ABC ACBD	Error: “строки разной длины!”	True
DBZEAGFXC DJFHJSNJAA	Error: “некорректные данные!”	True