Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №4

по теме «Бинарное дерево поиска»

Выполнили:

Студенты группы 21ВВ2

Козлова К.С.

Кривенкова В.С.

Принял:

Митрохин М.А.

Юрова О.В.

Пенза 2021

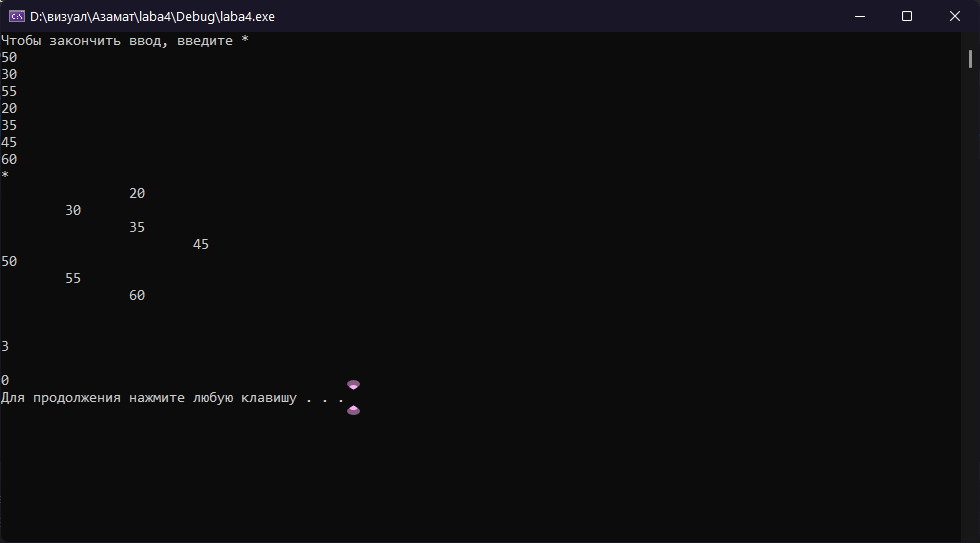
**Цель работы:**

Научиться реализовывать алгоритм поиска вводимого с клавиатуры значения в уже созданном дереве и вести подсчёт повторяющихся элементов.

**Лабораторные работы:**

1. Реализовать алгоритм поиска вводимого с клавиатуры значения в уже созданном дереве.
2. Реализовать функцию подсчёта числа вхождений заданного элемента в дерево.
3. \* Изменить функцию добавления элементов для исключения добавления одинаковых символов.
4. \* Оценить сложность процедуры поиска по значению в бинарном дереве.

**Результат работы программы:**



**Листинг:**

#include <iostream>

#include <stdlib.h>

#include <stdio.h>

using namespace std;

struct tree {

int info;

struct tree\* left;

struct tree\* right;

};

struct tree\* root;

struct tree\* stree(struct tree\* root, struct tree\* r, int info)

{

if (!r) {

r = (struct tree\*)malloc(sizeof(struct tree));

if (!r) {

printf("Не хватает памяти\n");

exit(0);

}

r->left = NULL;

r->right = NULL;

r->info = info;

if (!root) return r; /\* первый вход \*/

if (info < root->info) root->left = r;

else root->right = r;

return r;

}

//if (r->info == info)

// return root;

if (info < r->info)

stree(r, r->left, info);

else

stree(r, r->right, info);

return root;

}

void inorder(struct tree\* root, int l)

{

if (!root) return;

inorder(root->left, l + 1);

for (int i = 0; i < l; i++)

{

printf("\t");

}

printf("%d \n", root->info);

inorder(root->right, l + 1);

}

int search\_tree(struct tree\* root, int data)

{

int n = 0;

while (root != NULL) {

if ((data) == root->info)

return n;

if ((data) > (root->info))

root = root->right;

else

root = root->left;

n++;

}

return n;

}

// сложность O(n+e)

int Counter(struct tree\* root, int data)

{

int n = 0;

while (root != NULL) {

if ((data) == root->info)

n++;

if ((data) >= (root->info))

root = root->right;

else

root = root->left;

}

return n;

}

int main()

{

setlocale(0, "");

root = NULL; /\* инициализация корня дерева \*/

cout << "Чтобы закончить ввод, введите \*" << endl;

while (true)

{

string ss;

cin >> ss;

if (ss == "\*")

break;

root = stree(root, root, atoi(ss.c\_str()));

}

inorder(root, 0);

cout << endl << endl;

cout << search\_tree(root, 10);

cout << endl << endl;

cout << Counter(root, 10) << endl;

system("pause");

return 0;

}

**Вывод:** в ходе данной лабораторной работы мы научились реализовывать алгоритм поиска вводимого с клавиатуры значения в уже созданном дереве и вести подсчёт повторяющихся элементов.