Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №4

по теме «Бинарное дерево поиска.»

Выполнили:

Студенты группы 21ВВ2

Кожняков Д.М.

Кабачек Н.А.

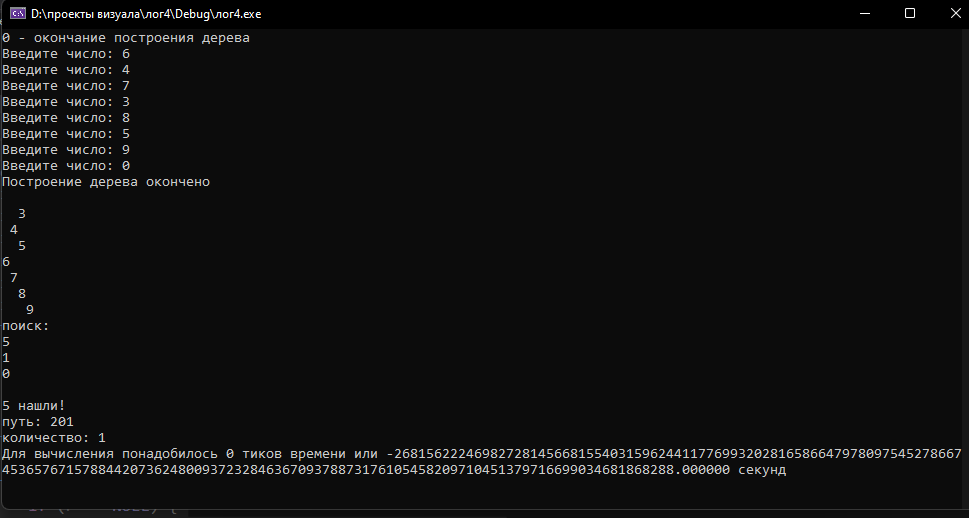
Принял:

Митрохин М.А.

Юрова О.В.

Пенза 2021

**Результат работы программы:**



**Задание 1:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

#include <malloc.h>

#include <string.h>

#include <ctime>

struct Node {

int data;

struct Node\* left;

struct Node\* right;

};

struct Node\* root;

struct Node\* CreateTree(struct Node\* root, struct Node\* r, int data, int d) {

if (r == NULL) {

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

if (r == NULL)

{

printf("Ошибка выделения памяти");

exit(0);

}

r->left = NULL;

r->right = NULL;

r->data = data;

if (root == NULL) return r;

if (data > root->data) root->left = r;

else root->right = r;

return r;

}

if (d == 1 && data == r->data)

return root;

if (data > r->data)

CreateTree(r, r->left, data, d);

else

CreateTree(r, r->right, data, d);

return root;

}

void print\_tree(struct Node\* r, int l)

{

if (r == NULL)

{

return;

}

print\_tree(r->right, l + 1);

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

{

printf(" ");

}

printf("%d\n", r->data);

print\_tree(r->left, l + 1);

}

int schet(struct Node\* r, int l) {

if (r == NULL) {

return 0;

}

if (r->data == l) {

return 1 + schet(r->right, l);

}

if (r->data > l) {

return schet(r->right, l);

}

if (r->data < l) {

return schet(r->left, l);

}

}

int poisk(struct Node\* r, int l) {

if (r == NULL) {

return 3;

}

if (r->data == l) {

printf("\n%d нашли!\n", r->data);

return 2;

}

if (r->data > l) {

printf("1\n");

return 10 \* poisk(r->right, l) + 1;

}

if (r->data < l) {

printf("0\n");

return 10 \* poisk(r->left, l);

}

}

int main() {

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int D, start = 1, ddd = 1;//повтороение запрещенно

root = NULL;

printf("0 - окончание построения дерева\n");

while (start) {

printf("Введите число: ");

scanf\_s("%d", &D);

if (D == 0) {

printf("Построение дерева окончено\n\n");

start = 0;

}

else

root = CreateTree(root, root, D, ddd);

}

print\_tree(root,0);

printf("поиск:\n");

scanf\_s("%d", &D);

time\_t start1 = clock();

printf("путь: %d\n", poisk(root, D));

time\_t stop = clock();

double time = (stop - start1);

printf("количество: %d\n", schet(root, D));

float sec = time / 1000;

printf("Для вычисления понадобилось %d тиков времени или %f секунд \n", time, sec);

scanf\_s("%d", &D);

return 0;

}

**Вывод:** Мы изучили принцип работы поиска в бинарных деревьях