Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №4

по курсу “Логика и основа алгоритмизации в ИЗ”

на тему “ Бинарное дерево поиска”

Выполнили студенты группы 21ВВ3:

Тюкалов В.Е.

Чинов Д.Д.

Приняли:

Митрохин М.А., Юрова О.В.

Пенза 2022

**Название**

Бинарное дерево поиска.

**Цель работы:** выполнить лабораторные указания 1-4 реализуя бинарное дерево поиска.

**Лабораторное задание:**

1. Реализовать алгоритм поиска вводимого с клавиатуры значения в уже созданном дереве.
2. Реализовать функцию подсчета 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.

**Листинг**

//Выролнили студенты группы 21ВВ3: Тюкалов В.Е. и Чинов Д.Д.

#include<iostream>

struct Node {

int data;

struct Node\* left;

struct Node\* right;

};

struct Node\* root;

int count = 0;

struct Node\* CreateTreeR(struct Node\* root, struct Node\* r, int data)

{

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 (data > r->data && data != r->data)

CreateTreeR(r, r->left, data);

else if (data < r->data && data != r->data)

CreateTreeR(r, r->right, data);

else

{

printf("Такой элемент уже есть\n");

}

return root;

}

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

{

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 (data > r->data)

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

else

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

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);

}

void counter(struct Node\* r, int dat)

{

if (r == NULL) return;

if (dat == r->data)

{

count++;

}

counter(r->right, dat);

counter(r->left, dat);

}

void find(struct Node\* r, int dat)

{

if (r == NULL) return;

if (dat == r->data)

{

printf("Найдено: %d", r->data);

return;

}

else

{

find(r->right, dat);

find(r->left, dat);

}

}

int main()

{

setlocale(LC\_ALL, "");

int D, start = 1, dat = 0, var = 0;

printf("Задание 3: \n");

printf("Довалять одинаковые элементы(0 - да, иначе - нет): ");

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

root = NULL;

printf("-1 - окончание построения дерева\n");

while (start)

{

printf("Введите число: ");

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

if (D == -1)

{

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

start = 0;

}

else

if(var == 0)

root = CreateTree(root, root, D);

else

root = CreateTreeR(root, root, D);

}

print\_tree(root, 0);

printf("\nЗадание 1: \n");

printf("Введите искомое число: ");

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

find(root, dat);

printf("\n\nЗадание 2: ");

counter(root, dat);

printf("\nКол-во %d: %d\n", dat, count);

return 0;

}

**Результаты работы программы:**

Результаты работы программы показан на рисунках 1 и 2.

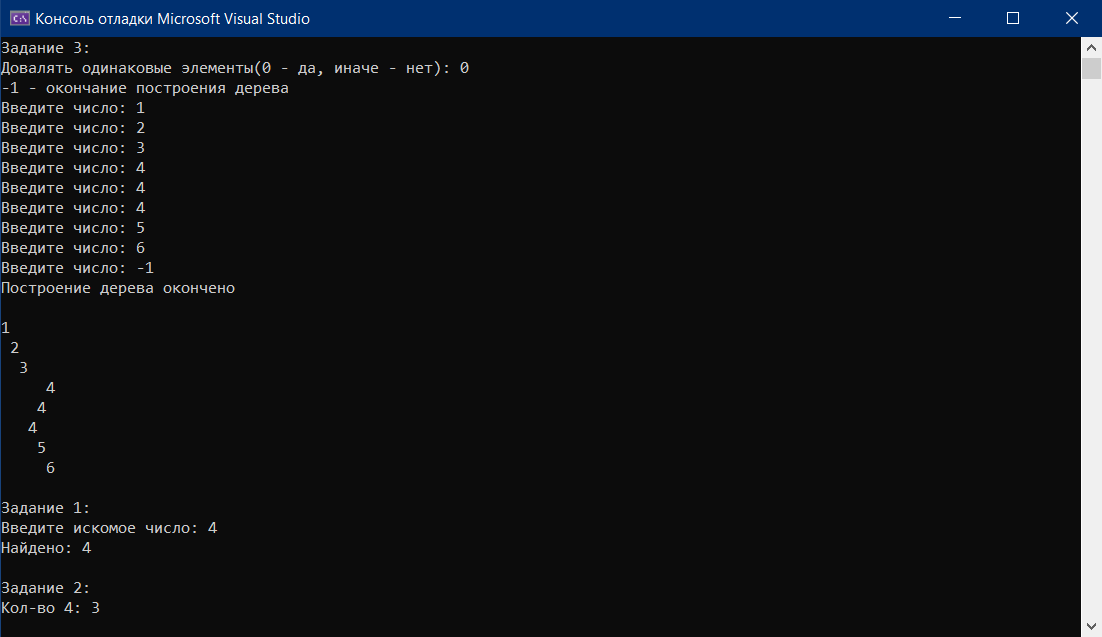


Рис. 1

**Задание 1:**

Реализован алгоритм поиска вводимого с клавиатуры значения в уже созданном дереве.

**Задание 2:**

Реализована функция подсчета числа вхождений заданного элемента в дерево.

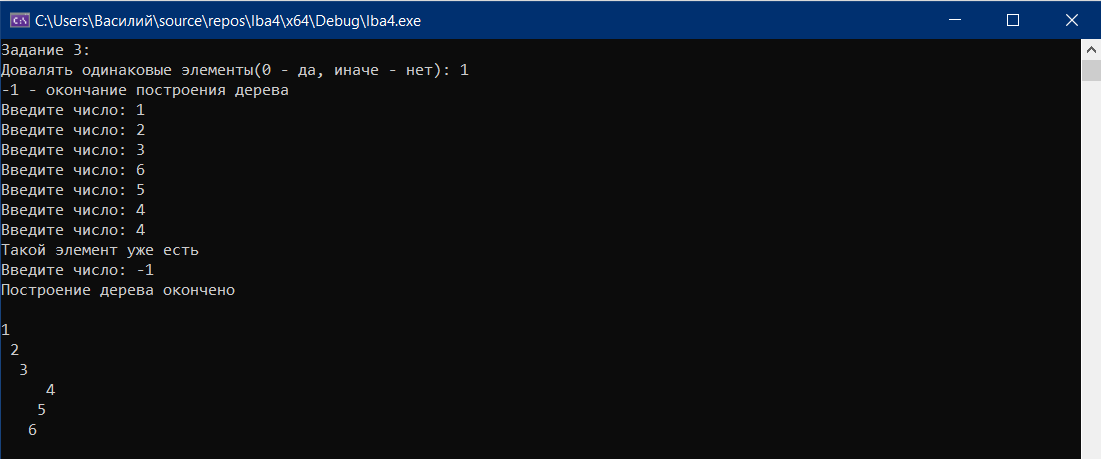


Рис. 2

**Задание 3:**

Изменена функция добавления элементов для исключения добавления одинаковых символов.

**Задание 4:**

Оценив процедуры поиска по значению в бинарном дереве, мы рассчитали сложность, которая ровна n\*Log n.

**Вывод:** Выполнив лабораторные задания 1-4 мы научились реализовывать структуру данных, бинарное дерево и реализовывать функцию работы с деревом.