Министерство науки и высшего образования Российской федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №4

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Бинарное дерево поиска»

Выполнили ст. группы 22ВВВ1:

Кирилин М. С.

Николаев А. А.

Приняли:

К.э.н., доцент Акифьев И. В.

К.т.н., доцент Юрова О. В.

Пенза 2023

**Цель работы:**

Цель данной лабораторной работы заключается в изучении и практическом применении бинарных деревьев в программировании.

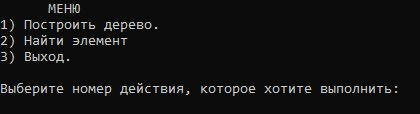
**Лабораторное задание:**

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.

**Ход работы:**

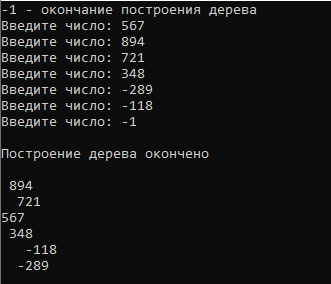
Объявляем структуру, которая представляет узел дерева. Создаем функции для построения, создания, вывода дерева на экран и поиска искомого значения. В главной функции вызываем заданные функции для заполнения, вывода дерева, поиска искомого значения и вывода количества вхождений определенного значения.

**Реализация меню:**

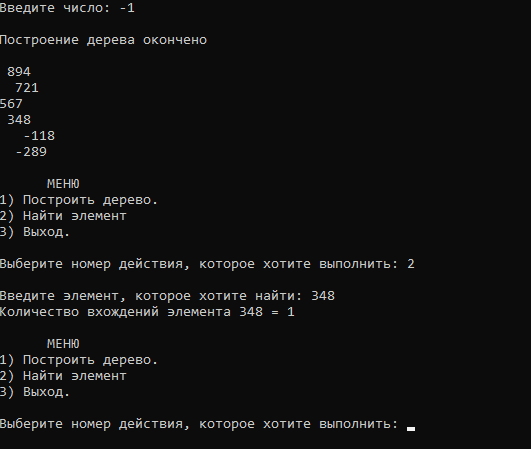
****

**Результаты работы программы:**

Основное задание:



Дополнительно задание:



**Вывод:**

В ходе лабораторной работы нами было изучено бинарное дерево поиска, которое является структурой данных, используемой для хранения и поиска элементов. Реализовали такие функции как: построение, создание, выводдерева на экран и поиск искомого значения. Оценили сложность кода, она равна O(log(n)).

**Листинг:**

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <Windows.h>

typedef struct Tree {

int data;

struct Tree\* left;

struct Tree\* right;

} Tree;

int k=-2;

Tree\* CreateTree(Tree\* root, int data)

{

if (root == NULL)

{

root = (Tree\*)malloc(sizeof(Tree));

if (root == NULL)

{

printf("Ошибка выделения памяти");

exit(0);

}

root->left = NULL;

root->right = NULL;

root->data = data;

return root;

}

if (data < root->data)

{

root->left = CreateTree(root->left, data);

}

else if (data > root->data)

{

root->right = CreateTree(root->right, data);

}

else

{

printf("Нельзя добавить уже существующий элемент!\n");

free(CreateTree(0, data));

return root;

}

return root;

}

//Функция для подсчета вхождений элемента в дерево

int counter(Tree\* root, int data) {

if (root == NULL) {

return 0;

}

if (root->data == data) {

k++;

return 1 + counter(root->left, data) + counter(root->right, data);

}

else if (data < root->data) {

k++;

return counter(root->left, data);

}

else {

k++;

return counter(root->right, data);

}

}

// Функция вывода дерева на экран

void print\_tree(Tree\* root, int level)

{

if (root == NULL)

{

return;

}

print\_tree(root->right, level + 1);

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

{

printf("\t");

}

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

print\_tree(root->left, level + 1);

}

int main()

{

SetConsoleCP(1251);//Установка языка

SetConsoleOutputCP(1251);

int data, start = 1;

Tree\* root = NULL;

int f = 0, element,input;

do

{

printf(" МЕНЮ\n");

printf("1) Построить дерево.\n");

printf("2) Найти элемент\n");

printf("3) Выход.\n");

printf("\nВыберите номер действия, которое хотите выполнить: ");

scanf("%d", &f);

switch (f)

{

case 1:

{

system("cls");

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

while (start)

{

printf("Введите число: ");

scanf("%d", &data);

if (data == -1)

{

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

start = 0;

}

else root = CreateTree(root, data );

}

print\_tree(root, 0);

printf("\n");

break;

}

case 2:

{

printf("\nВведите элемент, которое хотите найти: ");

scanf("%d", &element);

input = counter(root, element);

printf("Количество вхождений элемента %d = %d\n", element, input);

printf("Уровень элемента %d", k);

printf("\n");

break;

}

case 3:

{

system("cls");

printf("ВЫХОД\n");

return 0;

default:

printf("Неверный выбор\n");

return 0;

}

}

} while (f != 4);

return 0;

}