Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и кибербезопасности

Высшая школа компьютерных технологий и информационных систем

**Лабораторная работа № 7**

по дисциплине «Алгоритмы и структуры данных»

Выполнил

студент гр. 5130901/20003 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Вагнер А.А.

(подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Гаранин В.А.

(подпись)

 «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Санкт-Петербург   
2024

Цель работы

* освоить методы поиска.
* спроектировать и использовать созданное дерево поиска

Задание

Создать дерево поиска (в соответствии с индивидуальным заданием) и заполнить дерево данными из линейного массива данных. Осуществить поиск данных по дереву. Провести вывод на экран (файл по выбору) значений дерева разными способами (прямой, обратный, центральный) В случае невозможности (из условий индивидуального задания) поиска бинарным методом, провести поиск бинарным методом одной буквы (цифры). Придумать тестовые примеры, для которых были бы эффективными каждый из методов.

Вариант

Дан список фамилий. Необходимо заполнить дерево в соответствии с последними буквами фамилии. Произвести поиск введенной фамилии. Бинарное дерево поиска.

Код программы

#include <iostream>

#include <string>

#include <vector>

struct Node {

std::string lastName;

Node\* left;

Node\* right;

Node(std::string name) : lastName(name), left(nullptr), right(nullptr) {}

};

Node\* insert(Node\* root, std::string lastName) {

if (root == nullptr) {

return new Node(lastName);

}

if (lastName.back() < root->lastName.back()) {

root->left = insert(root->left, lastName);

}

else {

root->right = insert(root->right, lastName);

}

return root;

}

bool search(Node\* root, std::string lastName) {

if (root == nullptr) {

std::cout << lastName << " is not found D:" << std::endl;

return false;

}

if (root->lastName == lastName) {

std::cout << lastName << " is found :D" << std::endl;

return true;

}

else if (lastName.back() < root->lastName.back()) {

return search(root->left, lastName);

}

else {

return search(root->right, lastName);

}

}

int main() {

std::cout << "Hello World!";

Node\* root = nullptr;

// Пример заполнения дерева

std::vector<std::string> lastNames = { "Nazar", "Cherepovetz", "Yurga", "Yugra", "Bipki"};

for (const auto& name : lastNames) {

root = insert(root, name);

}

search(root, "Nazar");

search(root, "Kvazar");

return 0;

}

Элементы дерева

"Nazar", "Cherepovetz", "Yurga", "Yugra", "Bipki"

Результат работы программы

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

Вывод

В ходе данной работы были получены навыки написания алгоритмов поиска в бинарном дереве на языке С++.