МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



**ОТЧЁТ**

**по лабораторной работе № 6**

**«Деревья*»***

по дисциплине:

**«Программирование»**

|  |  |
| --- | --- |
| Выполнил(а):  Дашкин С.М. Группа АТ-34 | Проверил:  Кухто А.В. |

Новосибирск

2024

**ЦЕЛЬ РАБОТЫ**

|  |  |
| --- | --- |
| Вариант   6 | |
| Тип дерева | Дерево типа А |
| Исходные данные | Строка (макс. длина 8 символов) |
| Порядок обхода дерева | В ширину |

*Дерево типа А* – “Бинарное дерево”

Каждая вершина дерева содержит:

- 2 указателя (на каждый альтернативный путь поиска на поддерево);

- данные – указатель на объект, содержащий данные, принадлежащие этой вершине.

Примечание: по Вашему усмотрению можете добавить вспомогательные поля;

*Описание:*

Каждая вершина содержит 3 поля: поле данных и два указателя на другие вершины. Левый указатель служит для ссылки на вершину нижнего уровня, а правый – на соседнюю вершину ого же уровня. Указателям, не ссылающимся на другие вершины, присваивается значение NULL.

**ХОД РАБОТЫ:**

Код реализует структуру дерева, где каждая вершина имеет значение типа char и два указателя на левого и правого потомков.

Функция *insert* вставляет новую вершину в дерево в соответствии с порядком значений. Если значение новой вершины меньше значения текущей вершины, то новая вершина добавляется влево, иначе вправо.

Функция *breadthFirstTraversal* осуществляет обход дерева в ширину, начиная с указанной корневой вершины и ищет заданную вершину с помощью указанного значения destination. Выводит каждую вершину, которую проходит, пока не найдет заданное значение.

Функция *createNode* создает новую вершину с заданным значением данных.

В функции *main* пользователь вводит значение корневой вершины и количество вершин для вставки, затем вводит значения каждой вершины и добавляет их в дерево с помощью функции insert. Пользователь также вводит вершину, к которой он хочет пройти обход в ширину с помощью функции breadthFirstTraversal.

Наконец, код выводит результаты обхода в ширину до выбранной вершины и до максимально-заданной вершины с последующим очисткой памяти и завершением программы.

**ЛИСТИНГ:**

#define \_CRT\_SECURE\_NO\_DEPRECATE

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale.h>

struct TreeNode {

char data[9];

struct TreeNode\* left;

struct TreeNode\* right;

};

void insert(struct TreeNode\*\* root, struct TreeNode\* newNode) {

if (\*root == NULL) {

\*root = newNode;

}

else {

if (strcmp(newNode->data, (\*root)->data) < 0) {

if ((\*root)->left == NULL) {

(\*root)->left = newNode;

}

else {

insert(&((\*root)->left), newNode);

}

}

else {

if ((\*root)->right == NULL) {

(\*root)->right = newNode;

}

else {

insert(&((\*root)->right), newNode);

}

}

}

}

void breadthFirstTraversal(struct TreeNode\* root, char\* destination) {

if (root == NULL) {

return;

}

struct TreeNode\* queue[100];

int front = 0, rear = 0;

queue[rear++] = root;

while (front < rear) {

struct TreeNode\* current = queue[front++];

if (strcmp(current->data, destination) == 0) {

printf("%s ", current->data);

break;

}

printf("%s ", current->data);

if (current->left != NULL) {

queue[rear++] = current->left;

}

if (current->right != NULL) {

queue[rear++] = current->right;

}

}

}

struct TreeNode\* createNode(char\* data) {

struct TreeNode\* newNode = (struct TreeNode\*)malloc(sizeof(struct TreeNode));

strncpy(newNode->data, data, 8);

newNode->data[8] = '\0';

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

int main() {

struct TreeNode\* root = NULL;

setlocale(LC\_ALL, "Russian");

char input[9];

printf("Введите значение корневой вершины: ");

scanf("%8s", input);

root = createNode(input);

int numNodes;

printf("Введите количество вершин для вставки: ");

scanf("%d", &numNodes);

for (int i = 0; i < numNodes; i++) {

printf("Введите значение вершины %d: ", i + 1);

scanf("%8s", input);

struct TreeNode\* newNode = createNode(input);

insert(&root, newNode);

}

char destination[9];

printf("Введите вершину, в которую хотите пойти: ");

scanf("%8s", destination);

printf("Обход в ширину до выбранной вершины: ");

breadthFirstTraversal(root, destination);

struct TreeNode\* current = root;

printf("\nОбход в ширину до максимально-заданной вершины: ");

breadthFirstTraversal(root, current->data + 1);

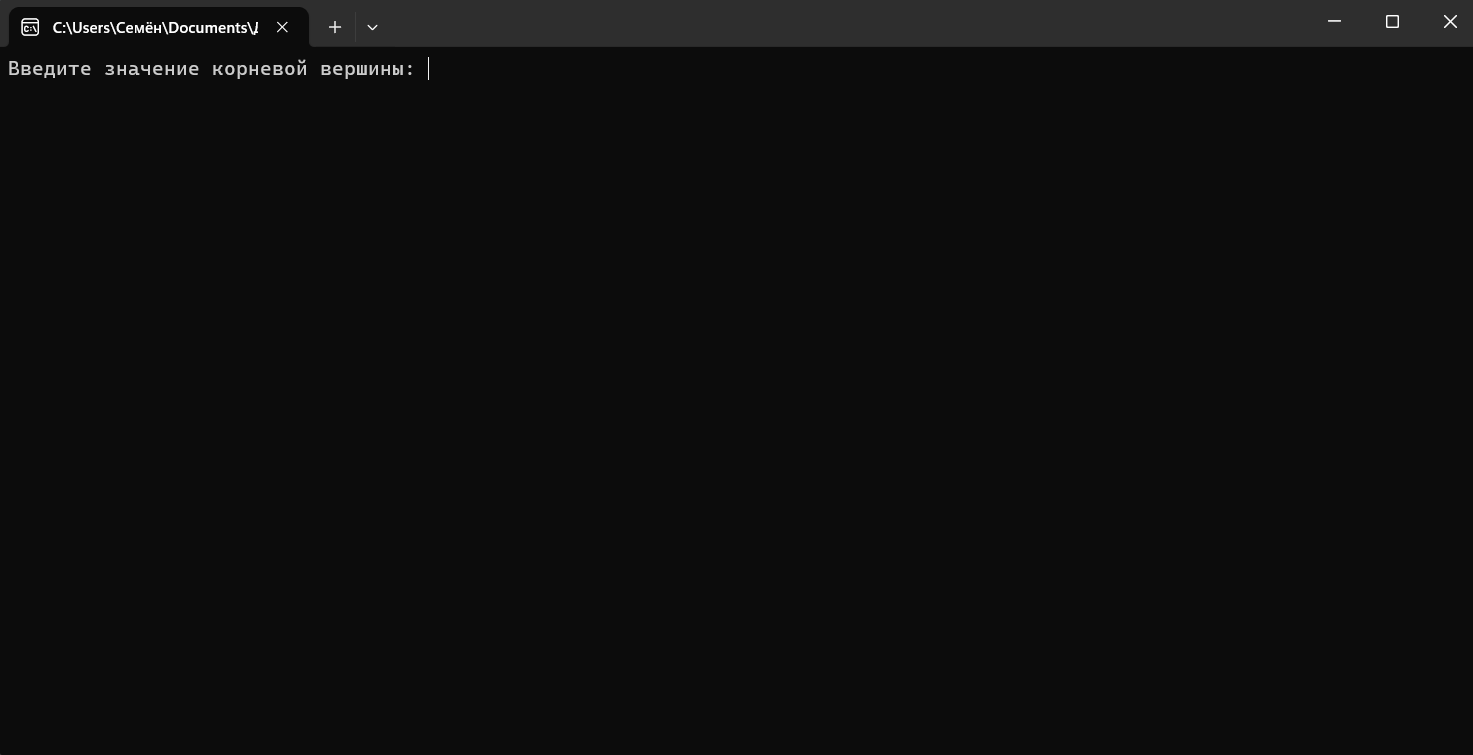
printf("\n");

return 0;

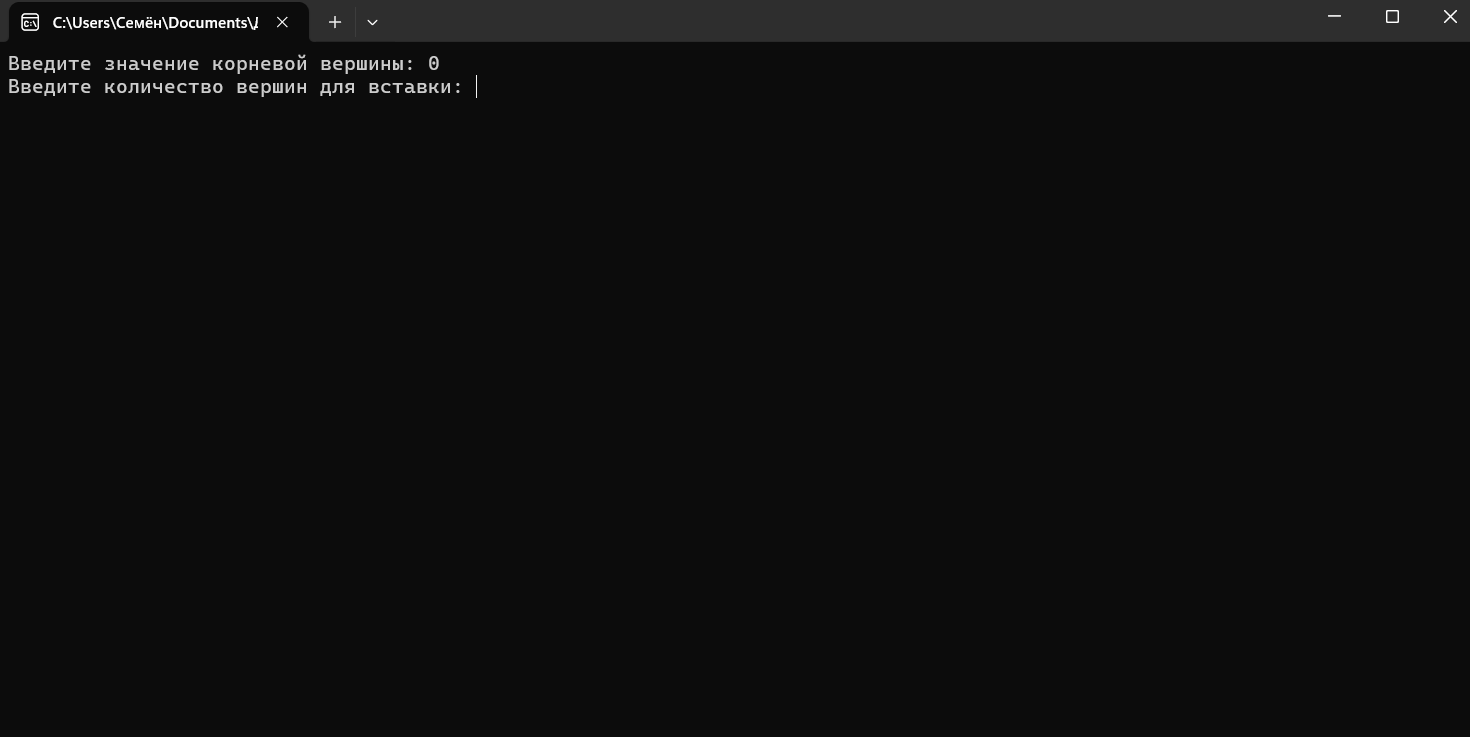
}

**ТЕСТИРОВАНИЕ:**

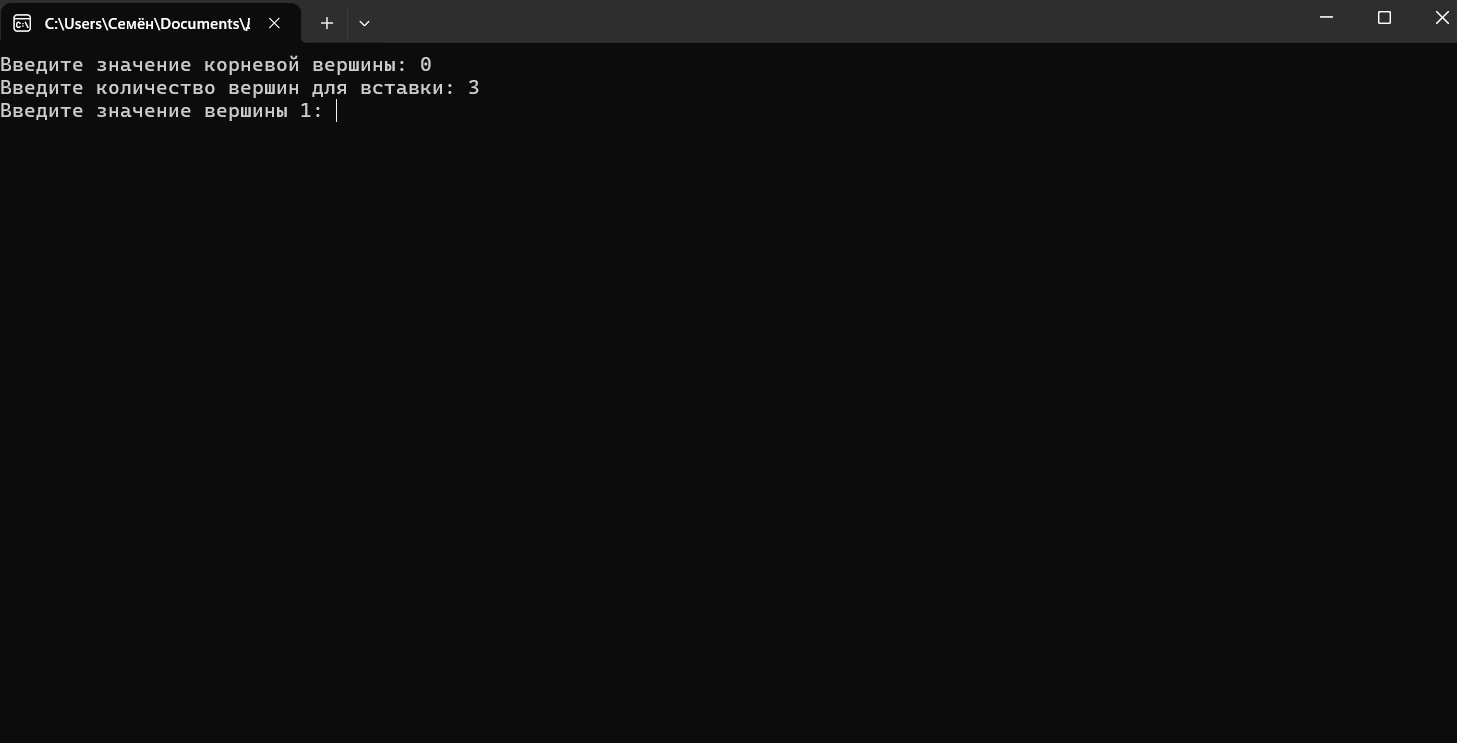
Запуск программы:



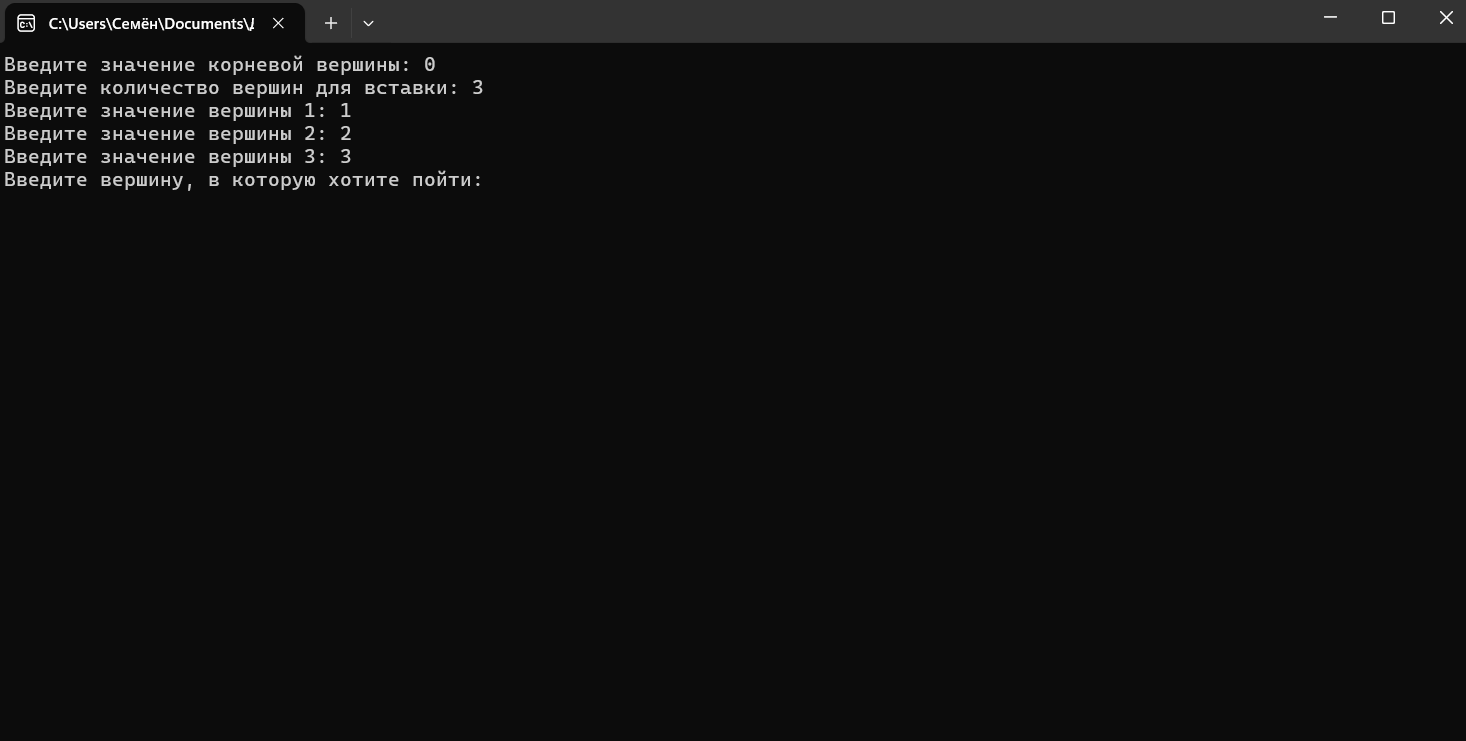
Введём любое значение, например 0.

****

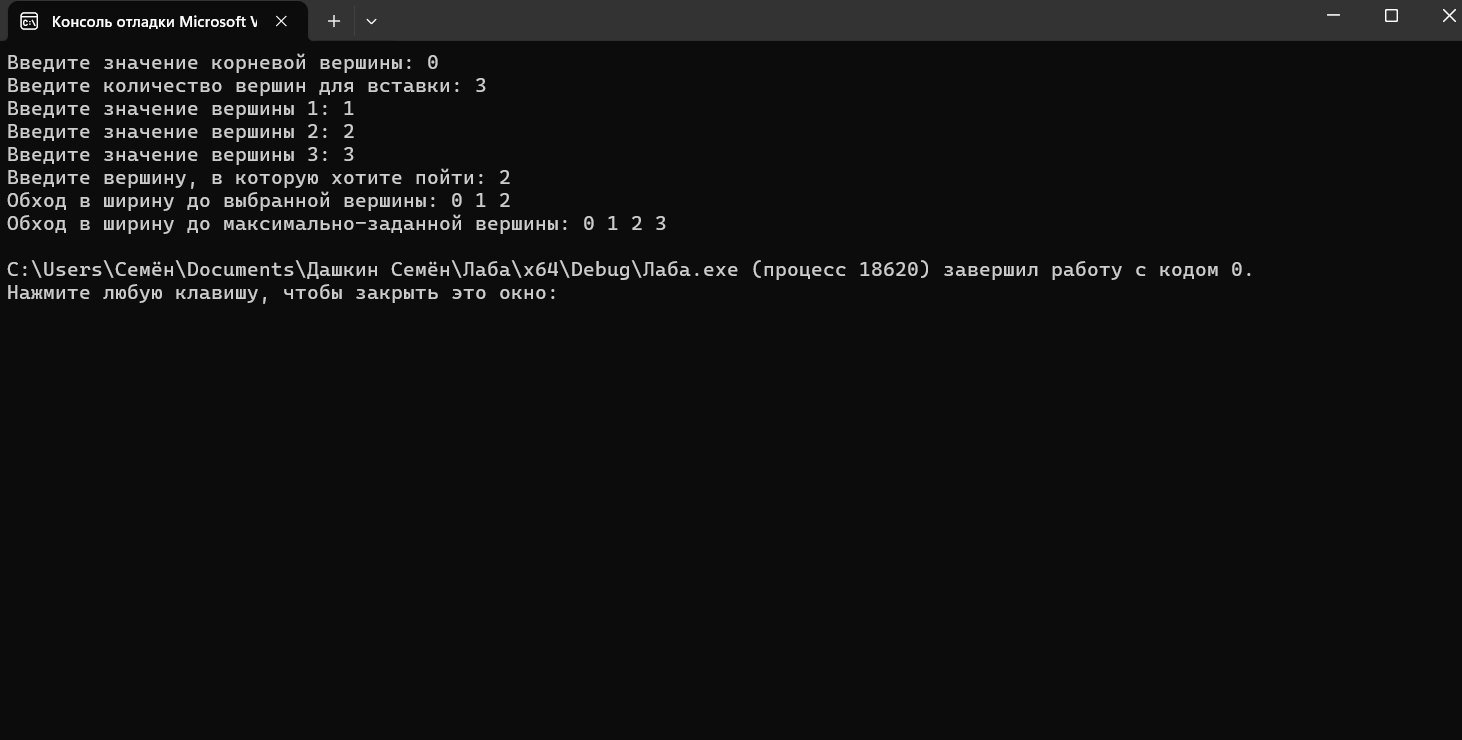
Теперь введём количество вершин, выходящие из корневой, например 3:



Далее зададим значения этим трём вершинам, например 1,2,3:

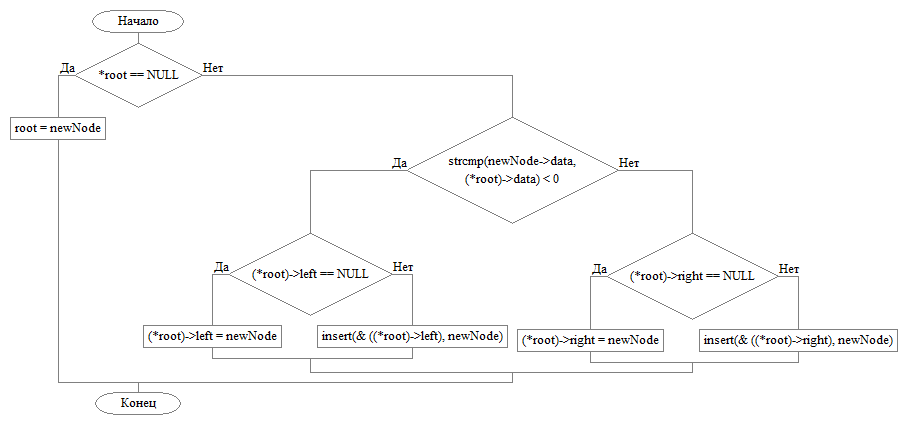


При вводе 2 нам выдаст обход до выбранной вершины и до максимально возможной заданной:

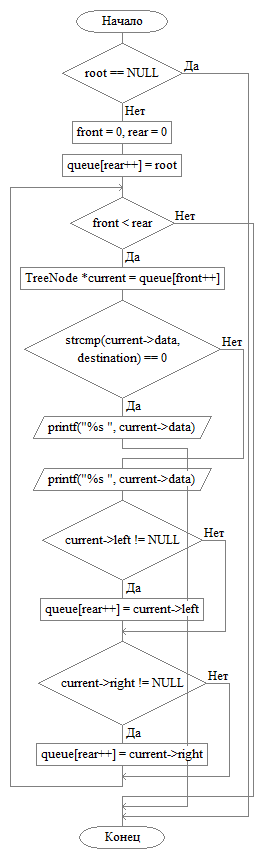


**БЛОК СХЕМЫ:**

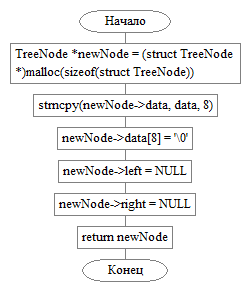
Блок-схема функции insert:



Блок-схема функции breadthFirstTraversal:



Блок-схема функции createNode:



Блок-схема функции main:

