Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

им. В.И. Ульянова (Ленина)»

кафедра Апу

**ОТЧЕТ**

**по лабораторной работе №4**

**«Бинарное дерево поиска»**

Выполнил: Панфёров М. А.

Группа №3371

Преподаватель: Ряскова Е. Б.

Санкт-Петербург

2024

Постановка задачи

Требуется создать программу, в которой:

* Реализован диалог с пользователем для ввода элементов множества, состоящего из целых чисел, и числа элементов этого множества.
* Реализован алгоритм построения бинарного дерева поиска на основе заданного множества.
* Реализован вывод на экран полученного дерева с использованием прямого, симметричного и обратного вариантов обхода.
* Для взаимодействия с пользователем реализуйте в программе меню, содержащее следующие разделы:
  + Ввод элементов множества
  + Построение и вывод на экран бинарного дерева
  + Выход из программы

Листинг программы

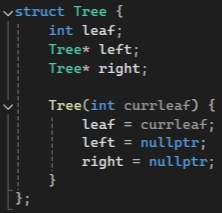


Рис. 1 – структура данных дерева.

На рисунке 1 показана структура данных дерева. Она в себе содержит данные, левую и правую ветви, а также конструктор, где все значения пустые.

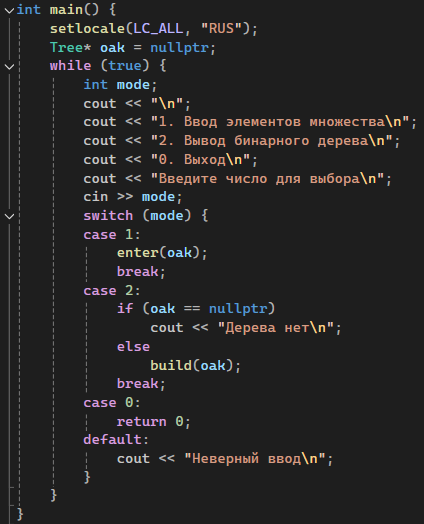


Рис. 2 – главная функция.

На рисунке 2 показана глафная функция. Тут создаётся наше дерево и оно пустое. После идёт меню с выбором действия.

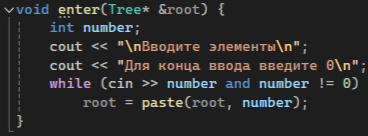


Рис. 3 – функция добавления элемента в дерево

На рисунке 3 показана функция добавления элемента в дерево. На вход подаётся дерево. В функции происходит ввод числа (цикл while работает, пока идёт ввод чисел и пока не будет введён 0). Рассмотрение функции paste ниже.

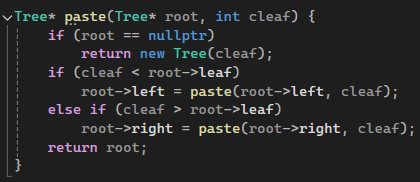


Рис. 4 – функция добавления числа в ветви дерева

На рисунке 4 показана функция добавления числа ветви дерева. На входе также подаётся дерево и число. В функции идёт распределение, в какую ветвь пойдёт число.

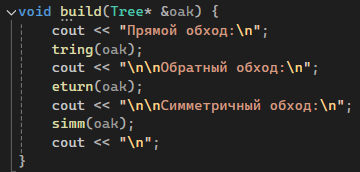


Рис. 5 – функция вывода разных методов обхода

На рисунке показана функция, где реализованы 3 метода обхода: прямой, обратный и симметричный. Функции tring, etrun и simm рассмотрены ниже.

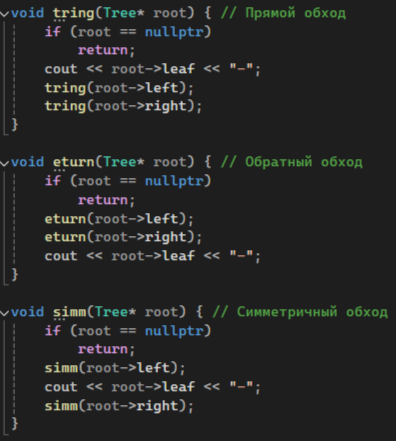


Рис. 7 – функции вывода обходов

На рисунке 7 показаны функции вывода обходов. Во всех обходах, если дерево пустое, то будет выход из функции.

Код и пример работы программы

AiSD\_Laba4.cpp:

#include "iostream"

using namespace std;

struct Tree {

int leaf;

Tree\* left;

Tree\* right;

Tree(int currleaf) {

leaf = currleaf;

left = nullptr;

right = nullptr;

}

};

Tree\* paste(Tree\* root, int cleaf) {

if (root == nullptr)

return new Tree(cleaf);

if (cleaf < root->leaf)

root->left = paste(root->left, cleaf);

else if (cleaf > root->leaf)

root->right = paste(root->right, cleaf);

return root;

}

void tring(Tree\* root) { // Прямой обход

if (root == nullptr)

return;

cout << root->leaf << "-";

tring(root->left);

tring(root->right);

}

void eturn(Tree\* root) { // Обратный обход

if (root == nullptr)

return;

eturn(root->left);

eturn(root->right);

cout << root->leaf << "-";

}

void simm(Tree\* root) { // Симметричный обход

if (root == nullptr)

return;

simm(root->left);

cout << root->leaf << "-";

simm(root->right);

}

void enter(Tree\* &root) {

int number;

cout << "\nВводите элементы\n";

cout << "Для конца ввода введите 0\n";

while (cin >> number and number != 0)

root = paste(root, number);

}

void build(Tree\* &oak) {

cout << "Прямой обход:\n";

tring(oak);

cout << "\n\nОбратный обход:\n";

eturn(oak);

cout << "\n\nСимметричный обход:\n";

simm(oak);

cout << "\n";

}

int main() {

setlocale(LC\_ALL, "RUS");

Tree\* oak = nullptr;

while (true) {

int mode;

cout << "\n";

cout << "1. Ввод элементов множества\n";

cout << "2. Вывод бинарного дерева\n";

cout << "0. Выход\n";

cout << "Введите число для выбора\n";

cin >> mode;

switch (mode) {

case 1:

enter(oak);

break;

case 2:

if (oak == nullptr)

cout << "Дерева нет\n";

else

build(oak);

break;

case 0:

return 0;

default:

cout << "Неверный ввод\n";

}

}

}

Пример работы программы:

1. Ввод элементов множества

2. Вывод бинарного дерева

0. Выход

Введите число для выбора

1

Вводите элементы

Для конца ввода введите 0

1

23

4

54

75

3

7

8

8

3

9

0

1. Ввод элементов множества

2. Вывод бинарного дерева

0. Выход

Введите число для выбора

2

Прямой обход:

1-23-4-3-7-8-9-54-75-

Обратный обход:

3-9-8-7-4-75-54-23-1-

Симметричный обход:

1-3-4-7-8-9-23-54-75-

1. Ввод элементов множества

2. Вывод бинарного дерева

0. Выход

Введите число для выбора

0

Вывод: я реализовал программу, которая строит деревья. Я узнал принцип трёх видов обхода и могу применять в своих проектах.