Министерство образования и науки Российской Федерации  
Федеральное‌ ‌государственное‌ ‌бюджетное‌ ‌образовательное‌ ‌учреждение‌

высшего‌ ‌образования‌

**«Пермский национальный исследовательский  
политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЁТ**

**по лабораторной работе №19**

Дисциплина: «Основы теории алгоритмов и структуры данных»

Тема: Реализация бинарных деревьев

Вариант 1

Выполнил:

Студент группы РИС-20-1б

Азмагулов Артём Вадимович

Проверила:

Доцент кафедры ИТАС

Полякова О. А.

**Пермь, 2021**

**Цель работы**

Ознакомление и реализация бинарных деревьев на языке программирования с++

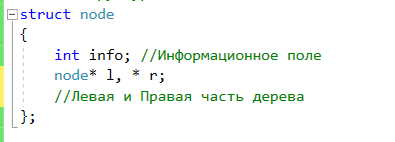
**Постановка задачи**

1. Сформировать идеально сбалансированное бинарное дерево, тип информационного поля указан в варианте.  
  2. Распечатать полученное дерево.  
  3. Выполнить обработку дерева в соответствии с заданием, вывести полученный результат.  
  4. Преобразовать идеально сбалансированное дерево в дерево поиска.  
  5. Распечатать полученное дерево.  
  
2. Варианты заданий  
  1. Тип информационного поля char. Найти количество элементов с заданным ключом.

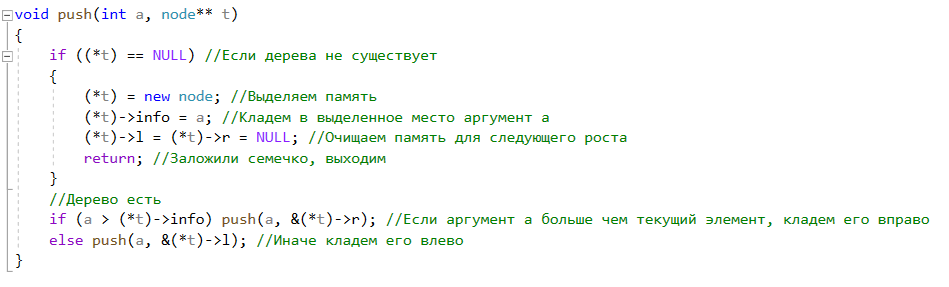
**Анализ задачи**

**1.** Для решения задачи необходимо:

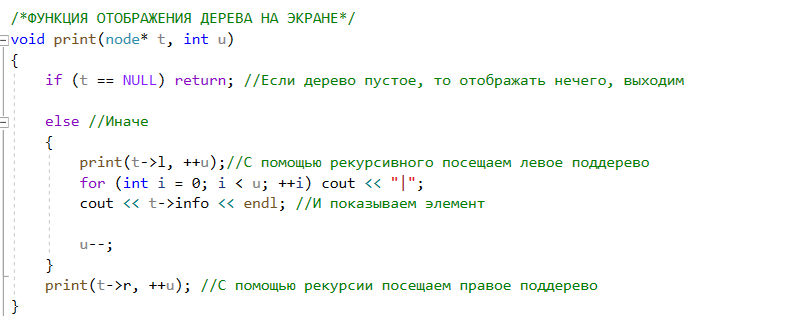
**1.1.** Реализовать структуру для хранения всех элементов дерева



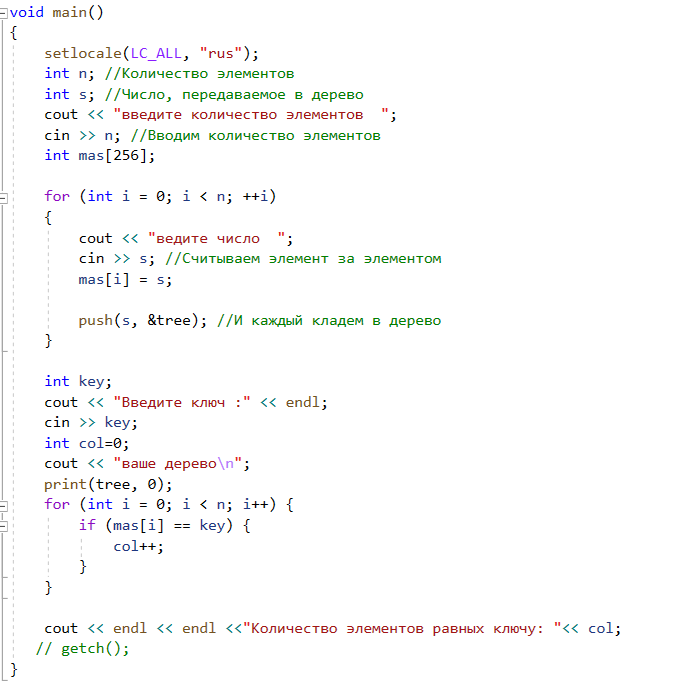
**1.2.** Реализовать функцию по записи элементов в дерево



1.3.Разработать функцию вывода дерева:



**1.4.** Реализовать главную функцию по приему и выводу всех параметров в функции:



**2.**В ходе работы были использованы следующие типы данных:

**2.1.** Все данные дерева принимались как элементы целочисленного типа:



**3.**Для решения задачи данные были представлены в следующем виде:

**3.1.**Данные введенные пользователем принимают целочисленное значение

**4.**Для операций ввода и вывода использовались следующие операторы и функции:

**4.1.**Ввод данных реализован при помощи функции cin:

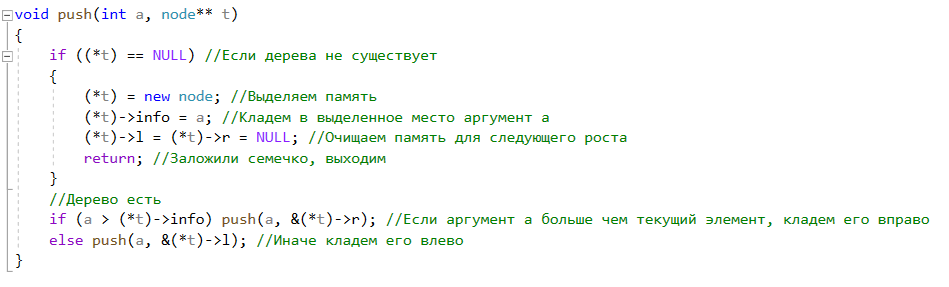


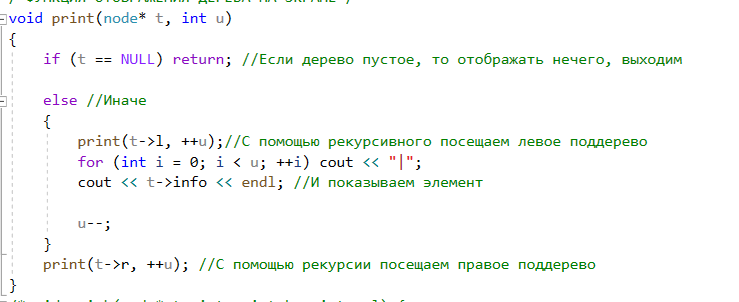
**4.2.** Вывод данных на консоль реализован с помощью оператора cout.



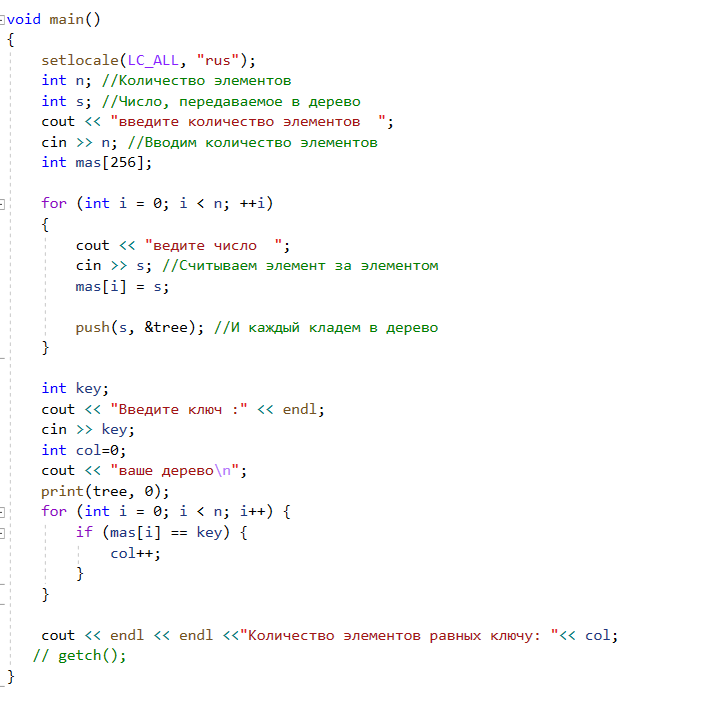
**5.**Поставленные задачи будут решены следующими действиями:

**5.1.** Для добавления нового элемента стека используется указатель на элемент , через который и происходит добавление нового элемента a в зависимости от сравнения с прошлым элементом.

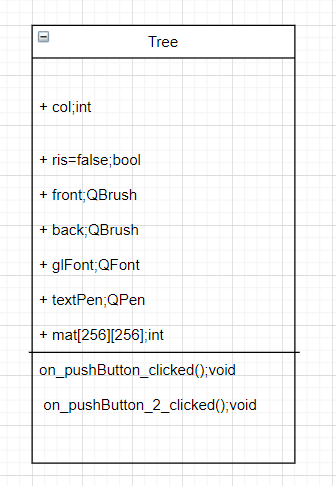


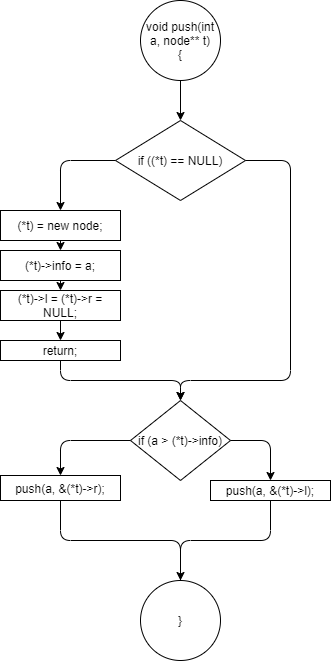
**5.2.**Вывод элементов дерева происходит рекурсивно с последующим обращением к соответствующему элементу дерева.

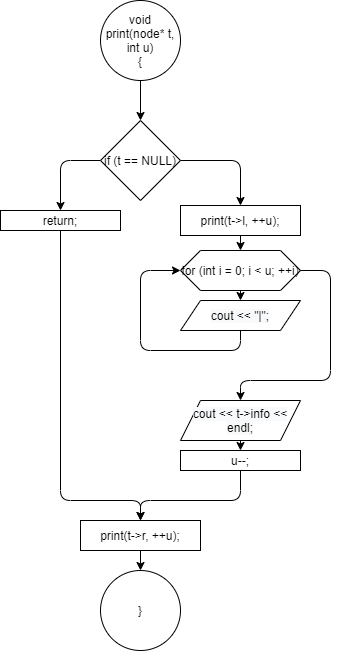
**5.3. .**В главной функции происходит объявление необходимых элементов с последующим вызовом функций.

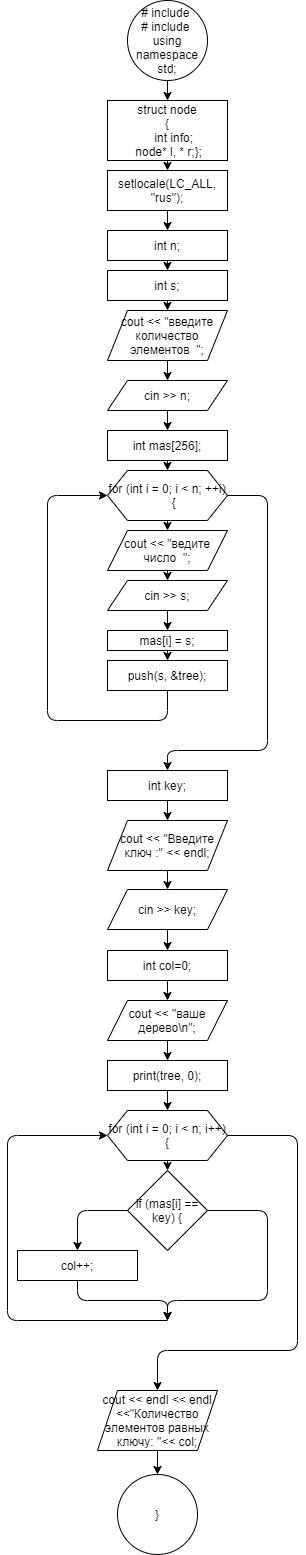


**Блок-схема программы**









**Решение**

# include <iostream>

# include <conio.h>

using namespace std;

//Наша структура

struct node

{

int info; //Информационное поле

node\* l, \* r;

//Левая и Правая часть дерева

};

node\* tree = NULL; //Объявляем переменную, тип которой структура Дерево

/\*ФУНКЦИЯ ЗАПИСИ ЭЛЕМЕНТА В БИНАРНОЕ ДЕРЕВО\*/

void push(int a, node\*\* t)

{

if ((\*t) == NULL) //Если дерева не существует

{

(\*t) = new node; //Выделяем память

(\*t)->info = a; //Кладем в выделенное место аргумент a

(\*t)->l = (\*t)->r = NULL; //Очищаем память для следующего роста

return; //Заложили семечко, выходим

}

//Дерево есть

if (a > (\*t)->info) push(a, &(\*t)->r); //Если аргумент а больше чем текущий элемент, кладем его вправо

else push(a, &(\*t)->l); //Иначе кладем его влево

}

/\*ФУНКЦИЯ ОТОБРАЖЕНИЯ ДЕРЕВА НА ЭКРАНЕ\*/

void print(node\* t, int u)

{

if (t == NULL) return; //Если дерево пустое, то отображать нечего, выходим

else //Иначе

{

print(t->l, ++u);//С помощью рекурсивного посещаем левое поддерево

for (int i = 0; i < u; ++i) cout << "|";

cout << t->info << endl; //И показываем элемент

u--;

}

print(t->r, ++u); //С помощью рекурсии посещаем правое поддерево

}

void main()

{

setlocale(LC\_ALL, "rus");

int n; //Количество элементов

int s; //Число, передаваемое в дерево

cout << "введите количество элементов ";

cin >> n; //Вводим количество элементов

int mas[256];

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

{

cout << "ведите число ";

cin >> s; //Считываем элемент за элементом

mas[i] = s;

push(s, &tree); //И каждый кладем в дерево

}

int key;

cout << "Введите ключ :" << endl;

cin >> key;

int col=0;

cout << "ваше дерево\n";

print(tree, 0);

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

if (mas[i] == key) {

col++;

}

}

cout << endl << endl <<"Количество элементов равных ключу: "<< col;

// getch();

}}

**Скриншоты результатов работы программы**

