Федеральное государственное автономное

образовательное учреждение

высшего профессионального образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ЛАБОРАТОРНАЯ РАБОТА №9**

# Тема: “Алгоритмы обработки двоичных деревьев”

# Вариант 2

Преподаватель

Студент КИ

Красноярск 2013

**Цель работы:**

приобретение навыков работы с нелинейными списками.

**Задача:**

Из входной последовательности чисел построить двоичное дерево и определить длину пути (число ветвей) от корня до вершины с минимальным элементом.

**Порядок выполнения работы:**  
  
Согласно условию задачи разработать блок-схему алгоритма, составить программу и отладить её, подготовить отчет и защитить лабораторную работу преподавателю.

#include <iostream>

using namespace std;

struct node{

int num;

int c;

node \*left;

node \*right;

};

node \*tree(node \*root, int num){

if(root == NULL)

{

root = new node;

root->num = num;

root->c = 1;

root->right = NULL;

root->left = NULL;

}

else

{

if(num == root->num)

{

root->c++;

}

else

{

if(num < root->num)

{

root->left = tree(root->left, num);

}

else

{

root->right = tree(root->right, num);

}

}

}

return root;

}

int getMinimal(node \*tree)

{

if(tree->left == NULL)

{

return tree->num;

}

return getMinimal(tree->left);

}

int getPathLen(node \*tree, int num)

{

if(num == tree->num)

{

return 0;

}

else

{

if( num > tree->num)

{

return getPathLen(tree->right, num) + 1;

}

else

{

return getPathLen(tree->left, num) +1;

}

}

}

int main()

{

node \*root = NULL;

int n, t, min, b;

puts("vvedite colichestvo elementov");

cin>>n;

puts("vvedite elements");

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

cin>>t;

root = tree(root, t);

}

cin>>b;

min = getMinimal(root);

cout<<min<<' '<<getPathLen(root, min)<<'\n';

system("pause");

return 0;

}

