Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Кафедра ИС

Отчет

По дисциплине: “Алгоритмизация и программирование”

Лабораторная работа №6

“Программирование нелинейных

структур данных на языке С/С++”

Выполнил:

ст.гр. ИС/б-12

Долженко И.А.

Проверил:

Сметанина Т.И.

Севастополь

2018

1 ЦЕЛЬ РАБОТЫ

Изучение нелинейных структур данных и приобретение навыков разработки и отладки программ, использующих древовидные структуры. Исследование особенностей работы с поисковыми бинарными деревьями на языке С/С++.

2 ПОСТАНОВКА ЗАДАЧИ

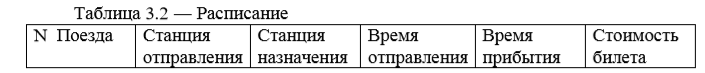
Представить приведенную таблицу в виде бинарного дерева. Написать функции создания и обхода дерева, а также одну из функций, приведенных ниже. Значения полей и количество записей в таблице студент выбирает самостоятельно. Программа должна сохранять дерево в файле и создавать его заново при её повторном запуске. 

Таблица 3.2. Функцию, которая определяет уровень, на котором находится элемент Е в дереве Т.

3 СТРУКТУРНАЯ СХЕМА АЛГОРИТМА

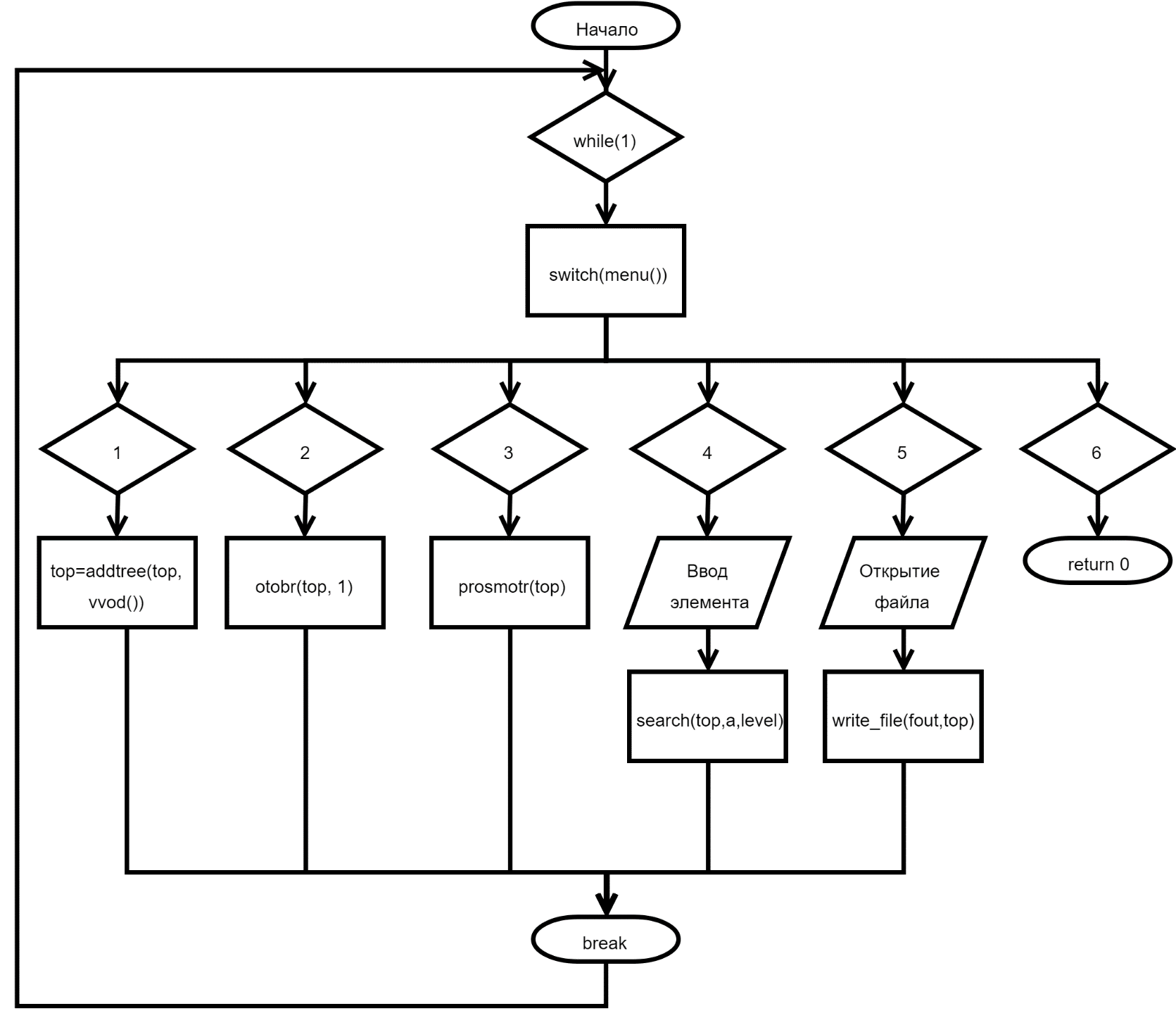


Рисунок 1 – Основная программа

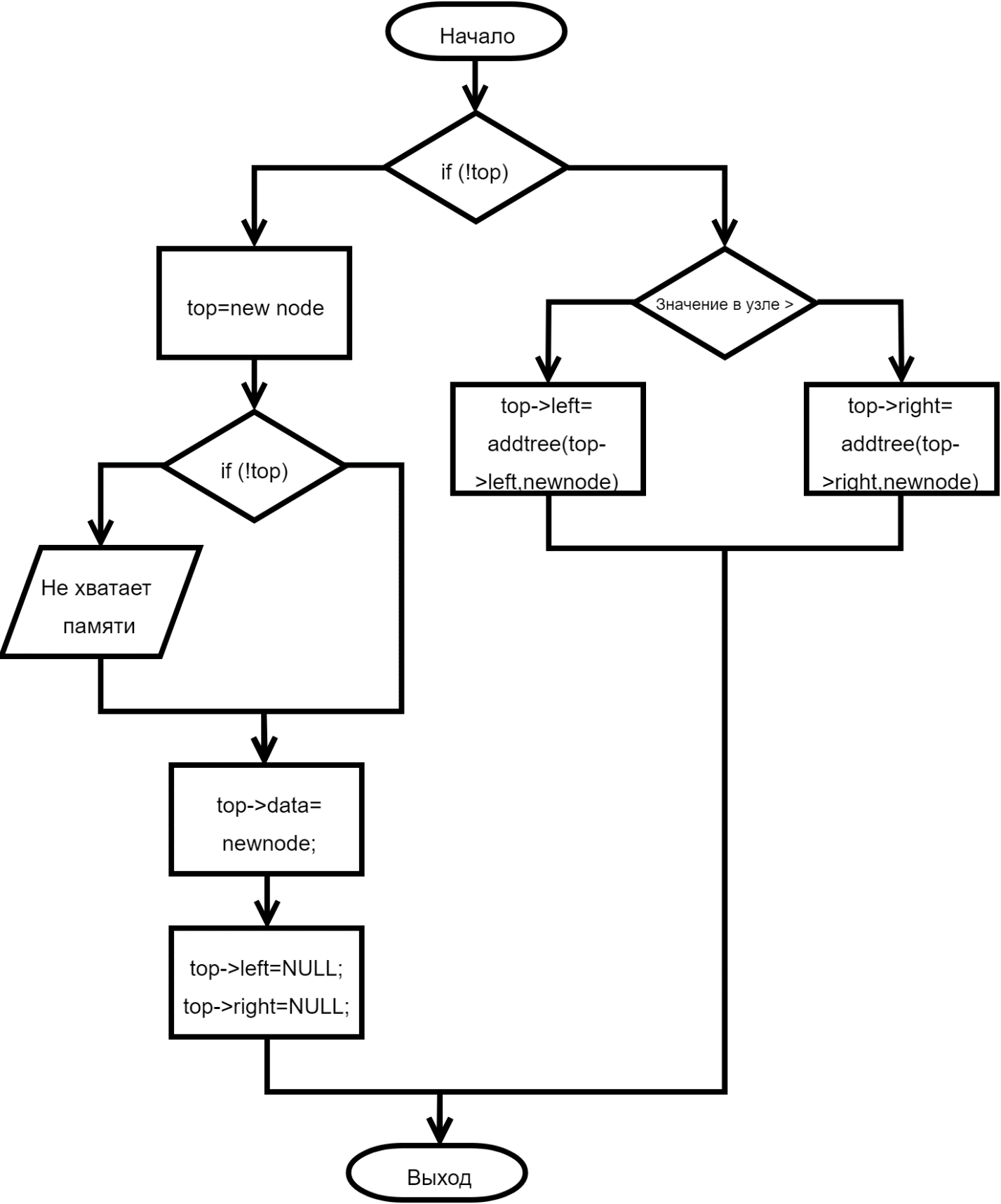
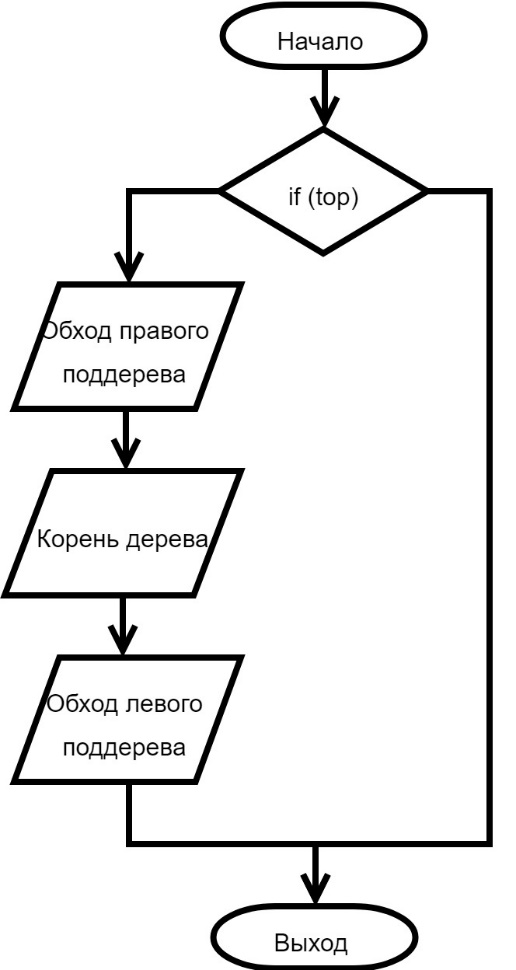
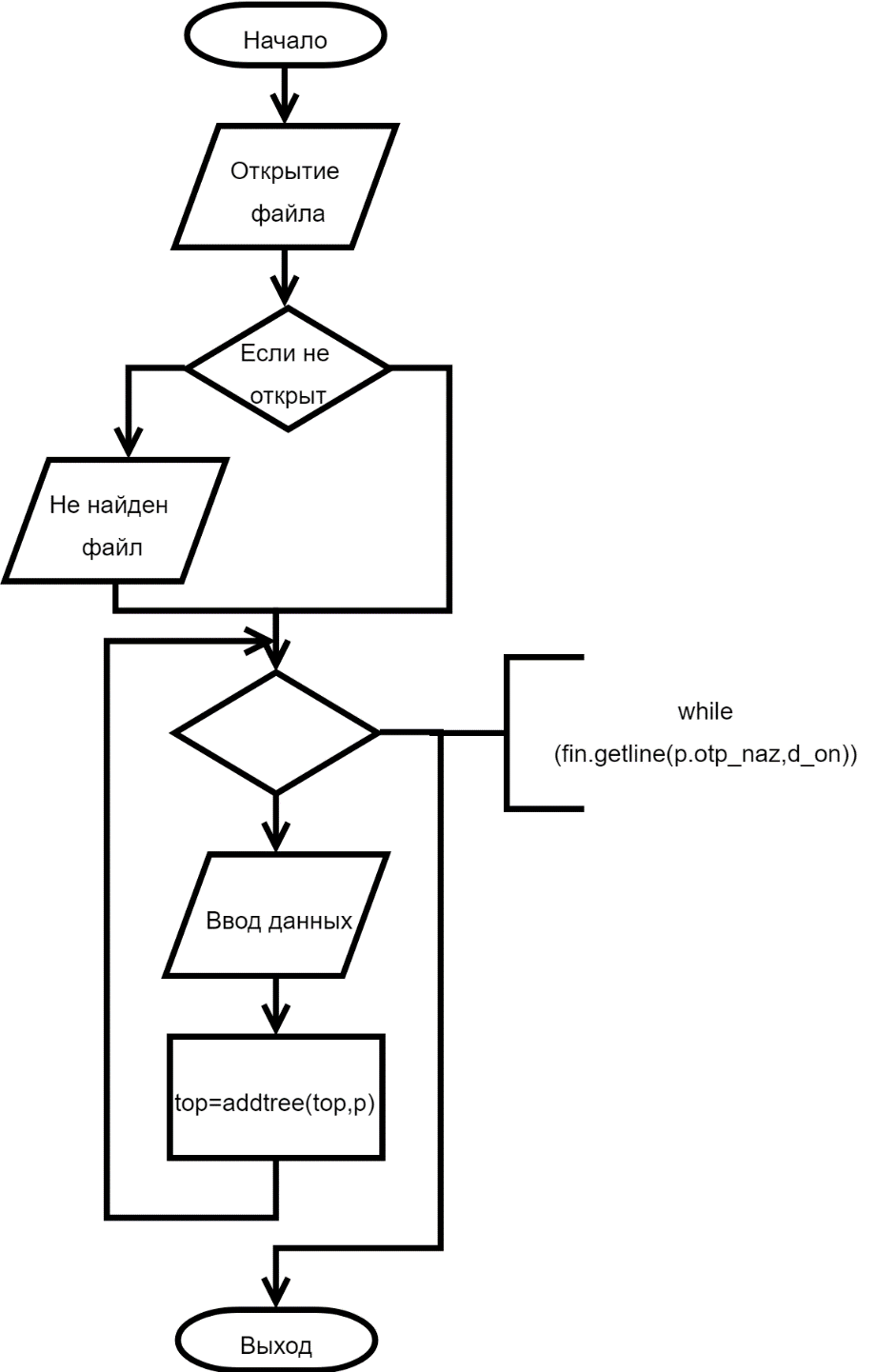


Рисунок 2 – Добавление узла в дерево



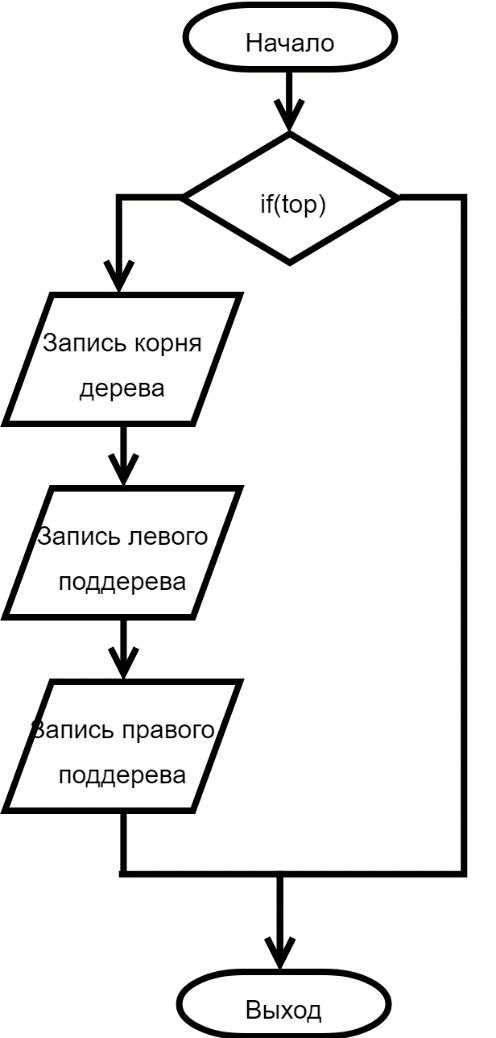
Рисунок 3 – Отображение структуры дерева

Рисунок 4 – Чтение файла

Рисунок 5 – Запись в файл

4 ТЕКСТ ПРОГРАММЫ

#include <fstream>

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <time.h>

#include <iomanip>

#include <conio.h>

#include <iostream>

using namespace std;

//----------------------константы-----------------------------------------

const int d\_on=80;

//---------------------структуры-------------------------------------------

struct zakaz

{

int nomer;

char otp\_naz[d\_on];

float price, time\_o, time\_p;

};

struct node

{ zakaz data;

node\* left;

node\* right;

};

//-------------прототипы функций--------------------------------------

node\* addtree(node \*top,const zakaz& newnode);

int menu();

int search(node \*top, char a,int &level);

void otobr(node \*top, int otstup);

void prosmotr(node \*top);

int write\_file(ofstream &f, node\* top);

int read\_file(char\* filename, node\* &top);

zakaz vvod();

//-------------------------основная функция------------------------------

int main()

{

setlocale(0,"RUS");

node \*top=0;

char \*filename="aero.txt";

ofstream fout;

read\_file(filename,top);

while (1)

{

switch (menu())

{

case 1:

top=addtree(top,vvod());

break;

case 2:

otobr(top,1);

cout<<"Нажмите любую клавишу "<<endl; cin.get(); break;

case 3:

prosmotr(top);

cout<<"Нажмите любую клавишу "<<endl; cin.get();

break;

case 4:

{

int level=1;

char a[d\_on];

cout<<"Введите элемент"<<endl;

cin>>a[d\_on];

cout<<"Элемент находится на уровне:"<<search(top,a,level)<<endl;

cout<<"Нажмите любую клавишу "<<endl; cin.get();

break;

}

case 5:

fout.open(filename);

if (!fout){cout<<"Ошибка открытия файла"<<endl; return 1;}

write\_file(fout,top);

cout<<"Данные сохранены в файле: "<<filename<<endl;

cout<<"=============================="<<endl;

fout.close();

cout<<"Нажмите любую клавишу "<<endl; cin.get();

break;

case 6:

return 0;

default:

cout<<"Вам следует ввести число от 1 до 6"<< endl; cin.get();

break;

}

}

}

//-----------------------добавление узла в дерево-------------------------

node\* addtree(node \*top,const zakaz& newnode)

{

if (!top)

{

top=new node;

if (!top)

{

cout<<"Не хватает памяти"<<endl;

return NULL;

}

top->data=newnode;

top->left=NULL;

top->right=NULL;

}

else

if (strcmp(top->data.otp\_naz,newnode.otp\_naz)>0)

top->left=addtree(top->left,newnode);

else top->right=addtree(top->right,newnode);

return top;

}

// ----------------отображение структуры дерева---------------------------

void otobr(node \*top, int otstup)

{

if (top)

{

otstup+=3;

otobr(top->right,otstup);

cout<<setw(otstup)<<'\*'<<top->data.otp\_naz<<endl;

otobr(top->left,otstup);

}

}

// ----------------просмотр дерева----------------------------------------

void prosmotr(node \*top)

{

if (top)

{

cout<<top->data.otp\_naz<<endl;

cout<<top->data.nomer<<endl;

cout<<top->data.price<<endl;

cout<<top->data.time\_o<<endl;

cout<<top->data.time\_p<<endl;

prosmotr(top->left);

prosmotr(top->right);

}

}

//-------------------ввод данных------------------------------------------

zakaz vvod()

{

zakaz p;

cout<<"Введите пункт отправления-пункт назначения"<<endl;

cin.getline(p.otp\_naz,d\_on);

cout<<"Введите номер рейса"<<endl;

cin>>p.nomer;

cout<<"Введите время отправления"<<endl;

cin>>p.time\_o;

cout<<"Введите время прибытия"<<endl;

cin>>p.time\_p;

cout<<"Введите цену"<<endl;

cin>>p.price;

return p;

}

//---------------отображение и ввод пунктов меню-----------------

int menu()

{

char buf[10];

int item;

do

{

system("CLS");

cout<<"1- Добавить элемент в дерево"<<endl;

cout<<"2- Отобразить структуру дерева"<<endl;

cout<<"3- Просмотр дерева"<<endl;

cout<<"4- Поиск уровня на котором находится элемент"<<endl;

cout<<"5- Запись данных в файл"<<endl;

cout<<"6- Выход"<<endl;

cout<<"============================"<<endl;

cout<<"Введите номер пункта меню:"<<endl;

cin>>buf;

cin.get();

item=atoi(buf);

if (!item)

{

cout<<"Вам следует ввести число от 1 до 6"<<endl;

cin.get();

}

}

while (!item);

return item;

}

//--------------------определить уровень элемента-------------------------

int search(node \*top, char a, int &level)

{

if (strcmp(a,top->data.otp\_naz)>0) level=search(top->right, a, level+1);

else if (strcmp(a,top->data.otp\_naz)<0) level=search(top->left, a, level+1);

else return level;

}

// -----------------------------чтение файла------------------------------

int read\_file(char\* filename, node\* &top ) {

ifstream fin(filename,ios::in);

if (!fin) {cout<<"Не найден файл"<<filename<<endl; return 1;}

zakaz p;

top = 0;

while (fin.getline(p.otp\_naz,d\_on))

{

fin>>p.nomer;

fin>>p.time\_o;

fin>>p.time\_p;

fin>>p.price;

top=addtree(top,p);

}

return 0;

}

// --------------------запись данных в файл-------------------------------

int write\_file(ofstream &f, node\* top) {

if (top) {

f<<top->data.otp\_naz<<endl;

f<<top->data.nomer<<endl;

f<<top->data.time\_o<<endl;

f<<top->data.time\_p<<endl;

f<<top->data.price<<endl;

write\_file(f,top->left);

write\_file(f,top->right);

}

return 0; }

5 ТЕСТИРОВАНИЕ

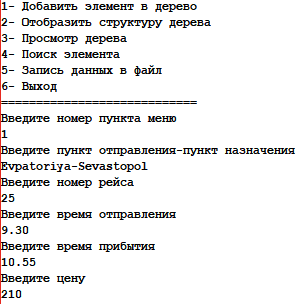


Рисунок 6 – Ввод нового элемента в дерево

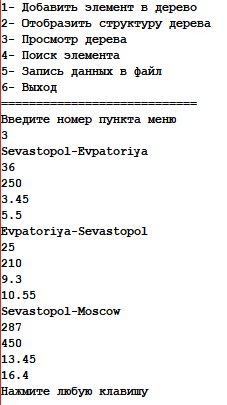


Рисунок 7 – Просмотр дерева

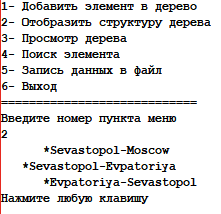
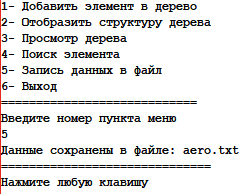


Рисунок 8 – Структура дерева

  
Рисунок 9 – Запись данных в файл

ВЫВОД

В ходе выполнения лабораторной работы были получены навыки работы с бинарными деревьями. Изучены основные операции над деревьями и их алгоритмы. Была написана программа, содержащая в себе функции создания и обхода бинарного дерева, а также функцию определения уровня, на котором находится элемент.