Министерство образования и науки Российской Федерации

Севастопольский Государственный Университет

Кафедра ИС

Отчёт

По учебной практике

за 1 курс

Вариант 3

Выполнил:

ст. гр. ИС/б-12-о

Волобуев Ю.С.

Проверил:

Забаштанский А.К.

Севастополь

2018

1.Тема: Программирование операций над строками и файлами

1.1 ЗАДАЧА

Написать программу, которая считывает текст из файла и определяет количество строк, начинающихся с пробела.

1.2 РЕШЕНИЕ ЗАДАЧИ

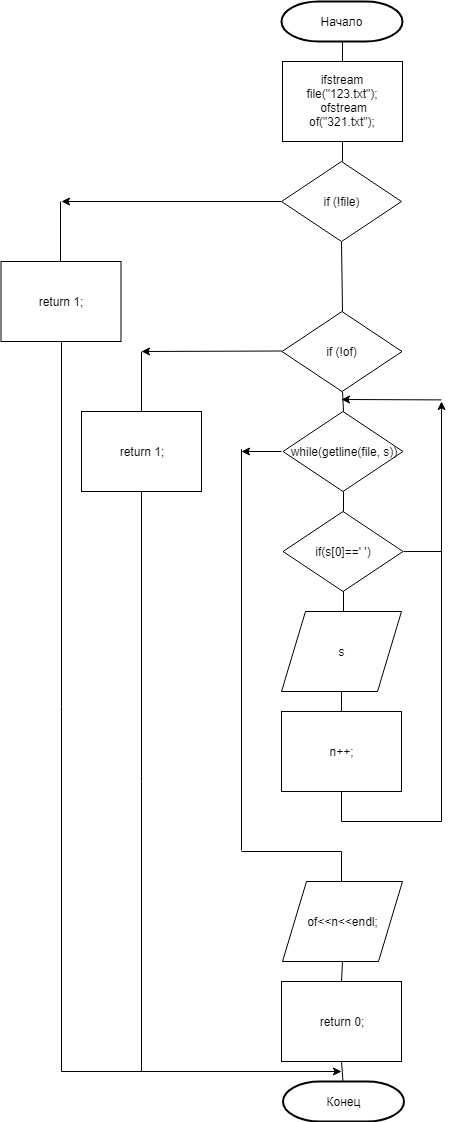


Рисунок 1.1 – Структурная схема алгоритма основной функции

1.3 ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <cstdio>

#include <cstring>

#include <conio.h>

#include <iostream>

#include <fstream>

#include <windows.h>

#include <locale.h>

using namespace std;

int main()

{

string s;

int n=0;

ifstream file("123.txt");

ofstream of("321.txt");

if (!file)

{

puts("Can't open inpfile\n");

return 1;

}

if (!of)

{

puts("Can't open outpfile\n");

return 1;

}

while(getline(file, s))

{

if(s[0]==' ')

{

cout<<s<<endl;

n++;

}

}

of<<n<<endl;

cout<<n;

file.close();

of.close();

return 0;

}

1.4 ТЕСТИРОВАНИЕ

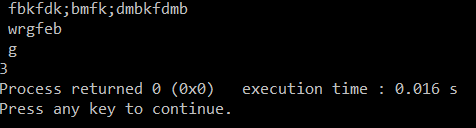
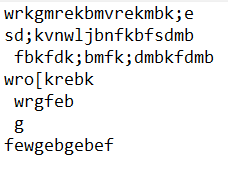


Рисунок 1.2 – Результат обработки программой тестового файла

2.Тема: Программирование операций над строками и файлами

2.1 ЗАДАЧА

Дана структура Дата, состоит из трех полей целого типа (День, Месяц и Год). Необходимо написать функцию, прибавляющую к дате один день.

2.2 РЕШЕНИЕ ЗАДАЧИ

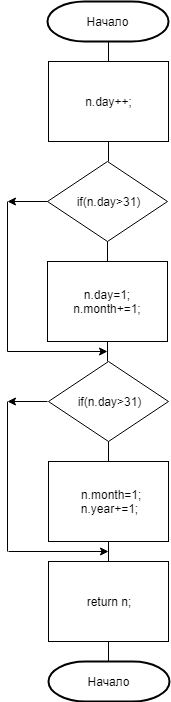


Рисунок 2.1 – Структурная схема алгоритма основной функции

2.3 ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <iostream>

#include <fstream>

#include <cstdlib>

#include <cassert>>

using namespace std;

struct date

{

int day, month, year;

};

using namespace std;

date ret(date n);

int main()

{

date n;

cin>>n.day>>n.month>>n.year;

n=ret(n);

cout<<n.day<<"."<<n.month<<"."<<n.year;

return 0;

}

date ret(date n)

{

n.day++;

if(n.day>31)

{

n.day=1;

n.month+=1;

}

if(n.month>12)

{

n.month=1;

n.year+=1;

}

return n;

}

2.4 ТЕСТИРОВАНИЕ

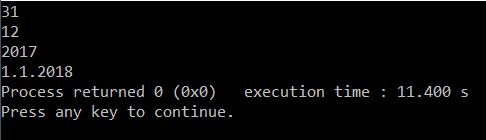


Рисунок 2.2 – Результат выполнения программы на тестовых данных

3.Тема: Программирование линейных списков на языке C/C++.

3.1 ЗАДАЧА

Представить таблицу в виде линейного списка, элементами которого являются строки таблицы.

Таблица 3.1 – Список студентов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Фамилия студента | Оценки | | |
| Физика | История | Математика |
|  |  |  |  |  |

Написать программу, содержащую следующие функции:

* создание списка (ввод данных с клавиатуры)
* просмотр списка (вывод данных в табличном виде)
* добавление нового элемента в список
* сохранение данных из списка в файл
* загрузка данных из файла в список
* удаление всех элементов списка

Оформить работу программу в виде меню. Каждый пункт меню отвечает одной из функций. Дополнить программу функцией в соответствии с вариантом задания. Предусмотреть все проверки.

Дополнительная функция: Написать функцию поиска по НЕ ключевому полю.

3.2 РЕШЕНИЕ ЗАДАЧИ

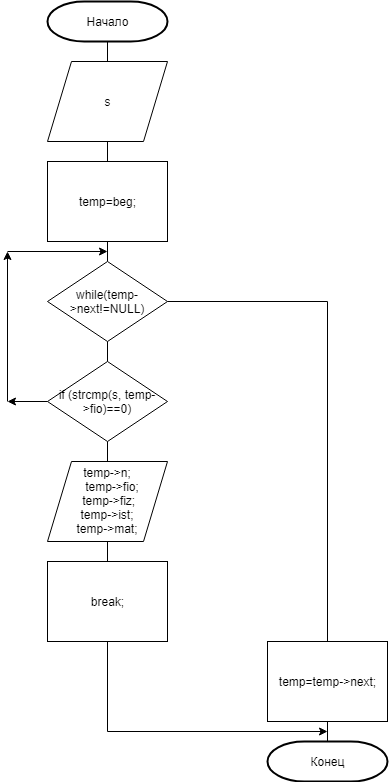


Рисунок 3.1 – Структурная схема алгоритма основной функции

3.3 ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <fstream>

#include <cstdlib>

#include <cstring>

#include <iomanip>

#include <conio.h>

#include <iostream>

#include <windows.h>

using namespace std;

const int lenfio=20, len=5;

struct stud

{

char fio[lenfio], n[len];

char fiz[len], ist[len], mat[len];

stud\* next;

} ;

const int size\_p=sizeof(stud);

stud\* add(stud\* end, const stud& z);

stud\* addf(const stud& z);

stud\* poisk(stud \*beg);

stud\* addsort(stud \*beg, const stud& z);

stud\* del(stud\* beg);

stud inpstruct();

int menu();

void print(const stud& z);

void view(stud\* beg);

int sorting(char\* file);

int readfile(char\* file, stud\*\* beg, stud\*\* end);

int writefile(char\* file, stud\* temp);

int main()

{

setlocale(0,"Russian");

stud \*beg=0, \*end=0;

char \*file="123.txt";

while (1)

{

switch (menu())

{

case 1: if (beg)

{

end=add(end,inpstruct());

}

else {

beg=addf(inpstruct());

end=beg;

}

break;

case 2: beg=del(beg);

cin.get();

break;

case 3: addsort(beg, inpstruct());

break;

case 4: view(beg); break;

case 5: writefile(file,beg); break;

case 6: readfile(file,&beg,&end); break;

case 7: poisk(beg); break;

case 8: return 0;

default: cout<<"Вам следует ввести номер от 1 до 8"<< endl;

cin.get();

break;

}

}

}

//-------------------------------------------------------------------

stud\* add(stud \*end,const stud& z)

{

stud \*elem=new stud;

\*elem=z;

elem->next=0;

end->next=elem;

end=elem;

return end;

}

stud\* addf(const stud& z)

{

stud \*beg=new stud;

\*beg=z;

beg->next=0;

return beg;

}

stud\* poisk(stud \*beg)

{

stud \*temp;

char s[lenfio];

cout<<"Введите ФИО студента: ";

cin>>s;

temp=beg;

while(temp->next!=NULL)

{

if (strcmp(s, temp->fio)==0)

{

cout<<setw(8)<<left<<temp->n;

cout<<setw(18)<<left<<temp->fio;

cout<<setw(9)<<left<<temp->fiz;

cout<<setw(10)<<left<<temp->ist;

cout<<setw(13)<<left<<temp->mat;

break;

}

temp=temp->next;

}

system("pause");

}

stud\* del(stud \*beg)

{

stud \*temp;

if (!beg) { cout<<"Очередь пустая"<<endl; return 0; }

while(temp->next!=NULL)

{

temp=beg;

beg=beg->next;

}

delete temp;

return beg;

}

stud\* addsort(stud \*beg, const stud& z)

{

stud\* elem=beg;

while(elem->next!=NULL)

{

if (strcmp(z.fio, elem->fio)>0)

{

stud \*newE=new stud;

\*newE=z;

newE->next=elem->next;

elem->next=newE;

}

elem=elem->next;

}

}

stud inpstruct()

{

char buf[10];

stud z;

cout<<"Введите номер"<<endl;

cin.getline(z.n,len);

cout<<"Введите Ф.И.О."<<endl;

cin.getline(z.fio,lenfio);

cout<<"Введите оценку по физике"<<endl;

cin.getline(z.fiz,len);

cout<<"Введите оценку по истории"<<endl;

cin.getline(z.ist,len);

cout<<"Введите оценку по математике"<<endl;

cin.getline(z.mat,len);

return z;

}

int menu()

{

char buf[10];

int item;

do

{

system("cls");

cout<<endl;

cout<<"1- Добавление элемента в очередь"<<endl;

cout<<"2- Удаление списка"<<endl;

cout<<"3- Добавление элемента с сохранением порядка"<<endl;

cout<<"4- Просмотр очереди"<<endl;

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

cout<<"6- Загрузка из файла"<<endl;

cout<<"7- Поиск студента"<<endl;

cout<<"8- Выход"<<endl;

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

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

cin>>buf;

cin.get();

item=atoi(buf);

if (!item)

{

cout<<"Вам следует вводить число от 1 до 8"<<endl;

cin.get();

}

}

while (!item);

return item;

}

void print(const stud& z)

{

cout<<setw(8)<<left<<z.n;

cout<<setw(18)<<left<<z.fio;

cout<<setw(9)<<left<<z.fiz;

cout<<setw(10)<<left<<z.ist;

cout<<setw(13)<<left<<z.mat;

}

void view(stud \*beg)

{

if (!beg)

{

cout<<"Очередь пустая"<<endl;

cin.get();

return;

}

stud \*temp=beg;

cout<<" Номер | Ф.И.О. | Физика | История | Математика "<<endl;

while (temp)

{

print(\*temp);

cout<<endl;

temp=temp->next;

}

cout<<"Нажмите любую клавишу"<<endl;

cin.get();

}

int readfile(char\* file,stud\*\* beg, stud\*\* end)

{

ifstream fin(file,ios::in);

if (!fin) {cout<<"Нет файла"<<file<<endl; return 1;}

stud z;

\*beg = 0;

while (fin.getline(z.n,len))

{

fin.getline(z.fio,lenfio);

fin.getline(z.fiz,len);

fin.getline(z.ist,len);

fin.getline(z.mat,len);

if (\*beg)

\*end=add(\*end,z);

else

{

\*beg=addf(z);

\*end=\*beg;

}

}

return 0;

}

int writefile(char\* file, stud\* temp)

{

ofstream fout(file);

if (!fout) {cout<<"Не могу открыть файл для записи"<<endl; return 1;}

while (temp)

{

fout<<temp->n<<endl;

fout<<temp->fio<<endl;

fout<<temp->fiz<<endl;

fout<<temp->ist<<endl;

fout<<temp->mat<<endl;

temp=temp->next;

}

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

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

cout<<"Нажмите любую клавишу"<<endl;

cin.get();

return 0;

}

3.4 ТЕСТИРОВАНИЕ

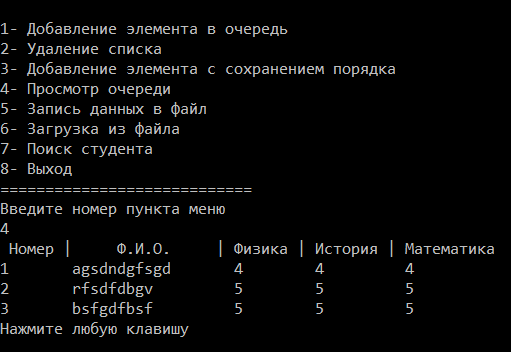


Рисунок 3.2 – Список студентов

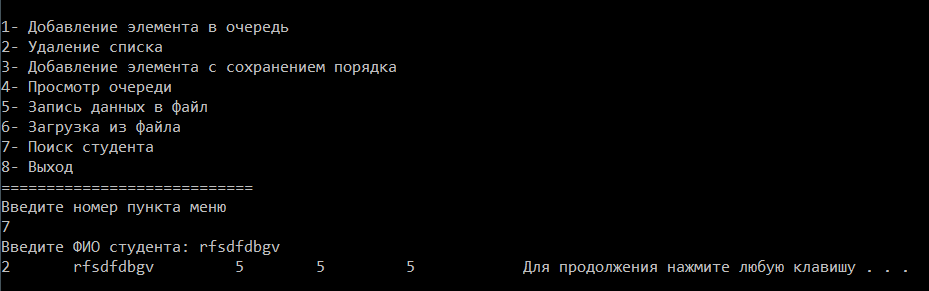


Рисунок 3.3 – Результат выполнения функции поиска

4.Тема: Программирование задач «Длинной арифметики»

4.1 ЗАДАЧА

Необходимо написать программу, выполняющую ввод и вывод длинных чисел через консоль. Вычислить произведение двух чисел 200 знаков каждое.

4.2 РЕШЕНИЕ ЗАДАЧИ

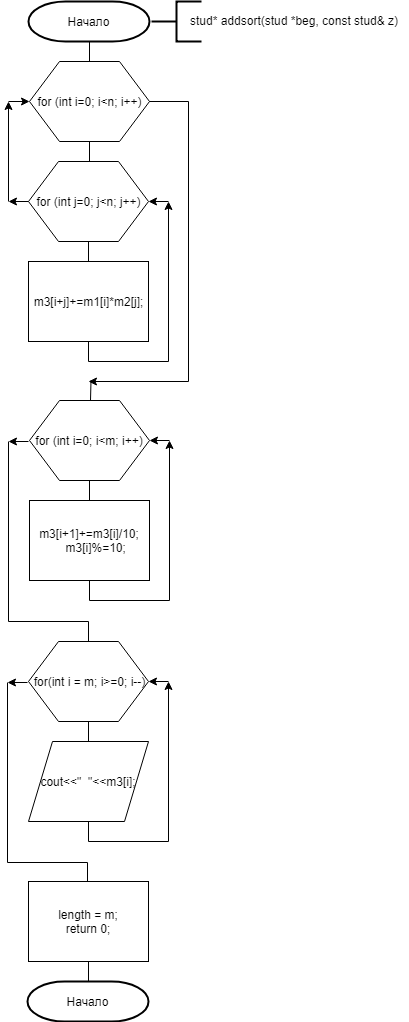


Рисунок 4.1 – Структурная схема алгоритма основной функции

4.3 ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <iomanip>

#include <conio.h>

#include <iostream>

#include <windows.h>

using namespace std;

const int n=2, m=5;

int m1[n] = {1,1};

int m2[n] = {1,1};

int m3[m], length;

char s1[n], s2[n];

int main()

{

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

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

m3[i+j]+=m1[i]\*m2[j];

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

{

m3[i+1]+=m3[i]/10;

m3[i]%=10;

}

for(int i = m; i>=0; i--)

cout<<" "<<m3[i];

length = m;

return 0;

4.4 ТЕСТИРОВАНИЕ

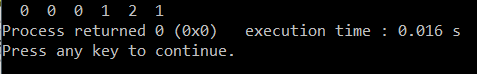


Рисунок 4.2 – Результат вычисления программы

Примечание: программа работает некорректно при вводе данных с клавиатуры, пробовал разные варианты ввода, в результате выходил мусор.

5. Тема: Программирование нелинейных динамических структур

5.1 ЗАДАЧА

Представить таблицу в виде бинарного дерева, элементами которого являются строки таблицы.

Таблица 5.1 – Список студентов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Фамилия студента | Оценки | | |
| Физика | История | Математика |
|  |  |  |  |  |

Написать программу, содержащую следующие функции:

* создание дерева (ввод данных с клавиатуры)
* просмотр дерева (вывод данных в табличном виде)
* отображение дерева (вывод структуры дерева)
* сохранение данных в файл
* загрузка данных из файла
* удаление всех элементов дерева

Оформить работу программу в виде меню. Каждый пункт меню отвечает одной из функций. Дополнить программу функцией в соответствии с вариантом задания. Предусмотреть все проверки.

Дополнительная функция: функция подсчета количества вершин, имеющих одного потомка.

5.2 РЕШЕНИЕ ЗАДАЧИ

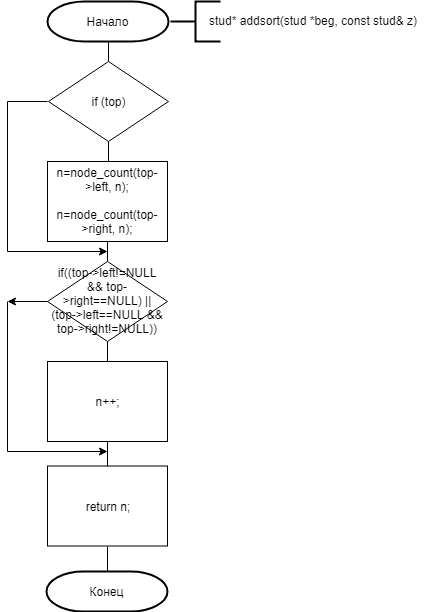


Рисунок 5.1 –Структурная схема алгоритма функции подсчета

5.3 ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <fstream>

#include <cstdlib>

#include <cstring>

#include <iomanip>

#include <conio.h>

#include <iostream>

#include <windows.h>

using namespace std;

const int lenfio=20, len=5;

struct stud

{

char fio[lenfio], n[len];

char fiz[len], ist[len], mat[len];

};

struct node

{

stud data;

node\* left;

node\* right;

};

const int size\_p=sizeof(stud);

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

int menu();

int node\_count(node \*top,int &n);

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

void prosmotr(node \*top);

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

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

void del(node \*top);

stud vvod();

int main()

{

setlocale(0,"Russian");

node \*top=0;

char \*file="stud.txt";

ofstream fout;

read\_file(file,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: {

cout<<" Номер | Ф.И.О. | Физика | История | Математика "<<endl;

prosmotr(top);

cout<<"Нажмите любую клавишу "<<endl;

cin.get();

break;

}

case 4: {

int level;

int n=0;

cout<<node\_count(top,n)<<endl;

cout<<"Нажмите любую клавишу "<<endl;

cin.get();

break;

}

case 5: {

ofstream fout(file);

if (!fout)

{

cout<<"Ошибка открытия файла"<<endl;

return 1;

}

write\_file(fout,top);

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

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

fout.close();

cout<<"Нажмите любую клавишу "<<endl;

cin.get();

break;

}

case 6: {

del(top);

break;

}

case 7: return 0;

default: {

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

cin.get();

break;

}

}

}

}

//-------------------------------------------------------------------

node\* addtree(node \*top,const stud& 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.n,newnode.n)>0)

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

else

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

return top;

}

void otobr(node \*top, int otstup)

{

if (top)

{

otstup+=3;

otobr(top->right,otstup);

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

otobr(top->left,otstup);

}

}

void prosmotr(node \*top)

{

if (top)

{

cout<<setw(8)<<left<<top->data.n;

cout<<setw(18)<<left<<top->data.fio;

cout<<setw(9)<<left<<top->data.fiz;

cout<<setw(10)<<left<<top->data.ist;

cout<<setw(13)<<left<<top->data.mat<<endl;

prosmotr(top->left);

prosmotr(top->right);

}

}

void del(node \*top)

{

if(top->left) del(top->left);

if(top->right) del(top->right);

delete top;

cout<<"Нажмите любую клавишу "<<endl;

cin.get();

}

stud vvod()

{

stud p;

cout<<"Введите номер"<<endl;

cin.getline(p.n,len);

cout<<"Введите ф.и.о."<<endl;

cin.getline(p.fio,lenfio);

cout<<"Введите оценку по физике"<<endl;

cin.getline(p.fiz,len);

cout<<"Введите оценку по истории"<<endl;

cin.getline(p.ist,len);

cout<<"Введите оценку по математике"<<endl;

cin.getline(p.mat,len);

return p;

}

int menu()

{

char key=0;

int point;

do

{

system("CLS");

if(key==72)

{

if(--point<0) point=7;

}

else if(key==80)

{

if(++point>7) point=0;

}

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

cout<<((point==0)?'>':' ')<<"1- Добавить элемент в дерево"<<endl;

cout<<((point==1)?'>':' ')<<"2- Отобразить структуру дерева"<<endl;

cout<<((point==2)?'>':' ')<<"3- Просмотр дерева"<<endl;

cout<<((point==3)?'>':' ')<<"4- Подсчет узлов с одним потомком"<<endl;

cout<<((point==4)?'>':' ')<<"5- Запись данных в файл"<<endl;

cout<<((point==5)?'>':' ')<<"6- Удаление дерева"<<endl;

cout<<((point==6)?'>':' ')<<"7- Выход"<<endl;

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

key=getch();

} while (key!=13 && key!=27);

system("CLS");

return key==13?point:6;

}

int node\_count(node \*top,int &n)

{

if (top)

{

n=node\_count(top->left, n);

n=node\_count(top->right, n);

if((top->left!=NULL && top->right==NULL) || (top->left==NULL && top->right!=NULL))

n++;

}

return n;

}

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

{

ifstream fin(file,ios::in);

if (!fin)

{

cout<<"Не найден файл"<<file<<endl;

return 1;

}

stud p;

top = 0;

while (fin.getline(p.n,len))

{

fin.getline(p.fio,lenfio);

fin.getline(p.fiz,len);

fin.getline(p.ist,len);

fin.getline(p.mat,len);

top=addtree(top,p);

}

return 0;

}

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

{

if (top)

{

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

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

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

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

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

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

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

}

return 0;

}

5.4 ТЕСТИРОВАНИЕ

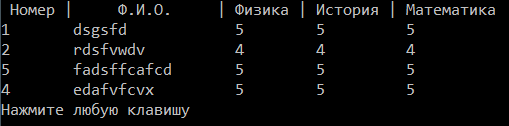


Рисунок 5.2 – Содержание дерева



Рисунок 5.3 – Результат работы функции подсчета

6.Тема: Реализация динамического интерфейса

6.1 ЗАДАЧА

Необходимо написать программу таким образом, чтобы интерфейс программы стал интерактивным, т.е. необходимый пункт меню выбирался с помощью управляющих клавиш (стрелочки) и запускался на выполнение нажатием клавиши Enter. Текущий пункт меню можно выделять подсветкой цветом или с помощью некоего “курсора” – вспомогательного символа.

6.2 ИСХОДНЫЙ КОД ПРОГРАММЫ

int menu()

{

char key=0;

int point;

do

{

system("CLS");

if(key==72)

{

if(--point<0) point=7;

}

else if(key==80)

{

if(++point>7) point=0;

}

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

cout<<((point==0)?'>':' ')<<"1- Добавить элемент в дерево"<<endl;

cout<<((point==1)?'>':' ')<<"2- Отобразить структуру дерева"<<endl;

cout<<((point==2)?'>':' ')<<"3- Просмотр дерева"<<endl;

cout<<((point==3)?'>':' ')<<"4- Подсчет узлов с одним потомком"<<endl;

cout<<((point==4)?'>':' ')<<"5- Запись данных в файл"<<endl;

cout<<((point==5)?'>':' ')<<"6- Удаление дерева"<<endl;

cout<<((point==6)?'>':' ')<<"7- Выход"<<endl;

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

key=getch();

} while (key!=13 && key!=27);

system("CLS");

return key==13?point:6;

}

6.4 ТЕСТИРОВАНИЕ

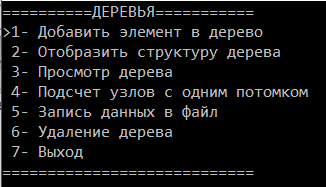


Рисунок 6.3 – Вид интерактивного меню программы