**Лабораторная работа №3**

*ЛИНЕЙНЫЕ СПИСКИ*

*Вариант 10*

*Выполнил:* Чекулаев В. Ю.

ПО(аб)-81

***Задание:*** В лабораторной работе требуется написать программу, обеспечивающую хранение и обработку информации в соответствии с вариантом (см. таб. 1). Для хранения данных использовать указанную структуру данных. Обеспечить выполнение дополнительных и основных операций при работе со списками:

1. добавление в список;
2. удаление из списка;
3. определение длины списка;
4. поиск элемента по значению;
5. изменение элемента списка;
6. вывод списка на экран.

Диалог с пользователем программы должен осуществляться с помощью меню.

***Задание в соответствии с вариантом:***

Реализовать ведение списков студентов нескольких учебных групп и списка студентов, находящихся в академическом отпуске. Сведения о студенте включают: номер зачетной книжки, ФИО, возраст. Для находящихся в академическом отпуске указать ФИО, дату начала и окончания отпуска.

Дополнительные операции:

1. Перевод студента из одной группы в другую.

2. Вывод для указанного студента длительность отпуска в месяцах

1) ***Код программы:***

#include<iostream>

#include<fstream>

#include<string>

using namespace std;

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

const unsigned short int stud = 20, academ\_num = 5;

struct DATE

{

string day;

string month;

string year;

};

struct LIST

{

bool academ = false;

short int num;

string book\_number;

string fio;

string age;

DATE academ\_start;

DATE academ\_finish;

int ptr;

};

int grForm(LIST\* group, int len); // Функция формирования групп из файла

void grOut(LIST\* group, int gr\_head); // Функция вывода списка группы на экран

void addStud(LIST\* group, int\* gr\_free, int\* gr\_head); // Функция добавления студента в группу

bool removeStud(LIST\* group, int\* gr\_free, int\* gr\_head, string fio); // Функция удаления студента из группы

int studAmount(LIST\* group, int gr\_head); // Функция подсчета количества студентов в группе

bool searchStud(LIST\* group, int gr\_head, int param); // Функция поиска студента по параметру

void showStud(LIST\* group, int head); // Функция вывода информации о студенте

bool changeStud(LIST\* group,int gr\_head, string fio); // Функция изменения информации о студенте

int vacation(LIST\* group, int gr\_head, string fio); // Функция подсчета длительности академического отпуска студента

bool transfer(LIST\* from, LIST\* to, int\* from\_head, int\* from\_free, int\* to\_head, int\* to\_free, string fio); //Функция перевода студента из одной группы в другую

bool inAcadem(LIST\* from, LIST\* to, int\* from\_head, int\* from\_free, int\* to\_head, int\* to\_free, string fio); //Функция отправки студента из группы в академический отпуск

bool fromAcadem(LIST\* academ, LIST\* gr1, LIST\* gr2, int\* academ\_head, int\* academ\_free, int\* gr1\_head, int\* gr1\_free, int\* gr2\_head, int\* gr2\_free, string fio); // Функция возврата студента из академического отпуска

unsigned short int mainMenu();

void clearDisplay();

int main()

{

unsigned short int ans;

clearDisplay();

/\*

LIST PO[stud];

int po\_head = 0;

PO[po\_head].ptr = -1;

int po\_free = 1;

for(int i = 1; i < stud-1; i++)

{

PO[i].ptr = i+1;

}

PO[stud-1].ptr = -1;

LIST MMI[stud];

int mmi\_head = 0;

MMI[mmi\_head].ptr = -1;

int mmi\_free = 1;

for(int i = 1; i < stud-1; i++)

{

MMI[i].ptr = i+1;

}

MMI[stud-1].ptr = -1;\*/

LIST ACADEM[stud];

int academ\_head = 0;

ACADEM[academ\_head].ptr = -1;

int academ\_free = 1;

for(int i = 1; i < stud-1; i++)

{

ACADEM[i].ptr = i+1;

}

ACADEM[stud-1].ptr = -1;

///\*

LIST PO[stud]; PO[0].num = 1;

int po\_head = 0;

int po\_free = grForm(PO, stud);

LIST MMI[stud]; MMI[0].num = 2;

int mmi\_head = 0;

int mmi\_free = grForm(MMI, stud);

//LIST ACADEM[stud]; ACADEM[0].num = academ\_num;

//int academ\_head = 0;

//int academ\_free = grForm(ACADEM, stud);

//\*/

string fio;

bool flag;

unsigned short int ans1;

main\_menu:

clearDisplay();

ans = mainMenu();

clearDisplay();

switch(ans){ // Отрисовка меню

case 0:

case 1:

case 2:

case 3:

case 4:

case 5:

case 6:

case 7:

case 8:

case 9:

case 10:

default:

}

exit:

clearDisplay();

return 0;

}

unsigned short int mainMenu(){

int ans;

cout << "--------------------------------------INFO--------------------------------------";

cout << " Формат записи даты: дд.мм.гггг\n";

cout << " Существующие списки групп: ПО, ММИ, АКАДЕМ(изначально пуст)\n";

cout << " Максимальное число cтудентов в группе: " << stud << "\n";

cout << "--------------------------------------------------------------------------------";

cout << "Выберите операцию: \n";

cout << " 1. Добавить студента в группу\n";

cout << " 2. Удалить студента из группы\n";

cout << " 3. Вывести на экран количество студентов в группе\n";

cout << " 4. Поиск студента в группе по параметру\n";

cout << " 5. Редактировать данные о студенте\n";

cout << " 6. Вывести список группы на экран\n";

cout << " 7. Перевести студента из одной группы в другую\n";

cout << " 8. Вывести для указанного студента длительность отпуска в месяцах\n";

cout << " 9. Отправить студента в академический отпуск\n";

cout << " 10. Вернуть студента из академического отпуска\n\n";

cout << " 0. Выйти из программы\n\n >";

cin >> ans;

return ans;

}

void clearDisplay(){

cout << "\033[2J\033[1;1H";

}

bool inAcadem(LIST\* from, LIST\* to, int\* from\_head, int\* from\_free, int\* to\_head, int\* to\_free, string fio){

int head = \*from\_head, buf, buf1 = to[\*to\_free].ptr, k = 0;

bool flag = true;

string date\_start, date\_finish, b\_n, age;

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

if(head != -1)

{

if(from[head].fio == fio)

{

to[\*to\_free].fio = from[head].fio;

cout << "Введите дату начала академического отпуска: ";

cin >> date\_start;

cout << "Введите дату конца академическгог отпуска: ";

cin >> date\_finish;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_start.day += date\_start[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_start.month += date\_start[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

to[\*to\_free].academ\_start.year += date\_start[k];

k++;

}

k = 0;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_finish.day += date\_finish[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_finish.month += date\_finish[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

to[\*to\_free].academ\_finish.year += date\_finish[k];

k++;

}

from[head].academ = true;

to[\*to\_free].ptr = \*to\_head;

\*to\_head = \*to\_free;

\*to\_free = buf1;

break;

}

buf = head;

head = from[head].ptr;

} else {

flag = false;

return flag;

}

}

return flag;

}

bool fromAcadem(LIST\* academ, LIST\* gr1, LIST\* gr2, int\* academ\_head, int\* academ\_free, int\* gr1\_head, int\* gr1\_free, int\* gr2\_head, int\* gr2\_free, string fio){

bool flag;

int head1 = \*gr1\_head, head2 = \*gr2\_head, buf;

flag = removeStud(academ, academ\_free, academ\_head, fio);

if(!flag){

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

if(gr1[head1].fio == fio){

gr1[head1].academ = false;

return flag;

} else{

buf = head1;

head1 = gr1[head1].ptr;

}

}

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

if(gr2[head2].fio == fio){

gr2[head2].academ = false;

return flag;

} else{

buf = head2;

head2 = gr2[head2].ptr;

}

}

}

return flag;

}

bool transfer(LIST\* from, LIST\* to, int\* from\_head, int\* from\_free, int\* to\_head, int\* to\_free, string fio){

int head = \*from\_head, buf, buf1 = to[\*to\_free].ptr, k = 0;

bool flag = true;

unsigned short int ans;

string date\_start, date\_finish, b\_n, age;

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

if(head != -1)

{

if(from[head].fio == fio)

{

if(from[head].academ\_start.day == "\0")

{

if(to[\*to\_head].academ\_start.day == "\0")

{

to[\*to\_free].book\_number = from[head].book\_number;

to[\*to\_free].age = from[head].age;

to[\*to\_free].fio = from[head].fio;

from[head].fio.clear();

from[head].book\_number.clear();

from[head].age.clear();

flag = false;

if(i == 0){

\*from\_head = from[head].ptr;

from[head].ptr = \*from\_free;

\*from\_free = head;

break;

}

from[buf].ptr = from[head].ptr;

from[head].ptr = \*from\_free;

\*from\_free = head;

to[\*to\_free].ptr = \*to\_head;

\*to\_head = \*to\_free;

\*to\_free = buf1;

break;

} else {

cout << "При переводе студента " << fio << " сотрутся его номер зачетной книжки и возраст. Продолжить? (1-да, 2-нет): ";

cin >> ans;

if(ans == 1){

to[\*to\_free].fio = from[head].fio;

from[head].fio.clear();

from[head].book\_number.clear();

from[head].age.clear();

cout << "Введите дату начала академического отпуска: ";

cin >> date\_start;

cout << "Введите дату конца академическгог отпуска: ";

cin >> date\_finish;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_start.day += date\_start[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_start.month += date\_start[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

to[\*to\_free].academ\_start.year += date\_start[k];

k++;

}

k = 0;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_finish.day += date\_finish[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

to[\*to\_free].academ\_finish.month += date\_finish[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

to[\*to\_free].academ\_finish.year += date\_finish[k];

k++;

}

if(i == 0){

\*from\_head = from[head].ptr;

from[head].ptr = \*from\_free;

\*from\_free = head;

break;

}

from[buf].ptr = from[head].ptr;

from[head].ptr = \*from\_free;

\*from\_free = head;

to[\*to\_free].ptr = \*to\_head;

\*to\_head = \*to\_free;

\*to\_free = buf1;

break;

} else{

flag = false;

break;

}

}

} else {

cout << "При переводе студента " << fio << " сотрутся даты начала и конца его академического отпуска. Продолжить? (1-да, 2-нет): ";

cin >> ans;

if(ans == 1){

to[\*to\_free].fio = from[head].fio;

from[head].fio.clear();

from[head].academ\_start.day.clear();

from[head].academ\_start.month.clear();

from[head].academ\_start.year.clear();

from[head].academ\_finish.day.clear();

from[head].academ\_finish.month.clear();

from[head].academ\_finish.year.clear();

flag = false;

cout << "Введите номер зачетной книжки: ";

cin >> b\_n;

cout << "Введите возраст: ";

cin >> age;

to[\*to\_free].book\_number = b\_n;

to[\*to\_free].age = age;

if(i == 0){

\*from\_head = from[head].ptr;

from[head].ptr = \*from\_free;

\*from\_free = head;

break;

}

from[buf].ptr = from[head].ptr;

from[head].ptr = \*from\_free;

\*from\_free = head;

to[\*to\_free].ptr = \*to\_head;

\*to\_head = \*to\_free;

\*to\_free = buf1;

break;

}

}

flag = false;

}

buf = head;

head = from[head].ptr;

} else{

break;

}

}

return flag;

}

int vacation(LIST\* group, int gr\_head, string fio){

bool flag = true;

int head = gr\_head, result = 0, tmp;

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

{

if(head != -1){

if(group[head].fio == fio){

result = stoi(group[head].academ\_finish.month) - stoi(group[head].academ\_start.month);

tmp = stoi(group[head].academ\_finish.year) - stoi(group[head].academ\_start.year);

if((result < 0 && tmp == 1) || (result > 0 && tmp == 1)){

result += 12;

} else if((result == 0 && tmp == 2)){

result += 24;

}

return result;

flag = false;

break;

}

head = group[head].ptr;

} else{

break;

}

}

return 0;

}

bool changeStud(LIST\* group,int gr\_head, string fio){

int head = gr\_head;

int ans, k = 0;

string tmp, day, month, year;

bool flag = true;

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

if(group[head].fio == fio)

{

cout << "\n";

showStud(group, head);

cout << "\n";

cout << "Изменить ФИО студента? (1-да; 2-нет): ";

cin >> ans;

if(ans == 1){

cout << "Введите ФИО: ";

cin.ignore();

getline(cin,tmp);

group[head].fio = tmp;

cout << "\n";

showStud(group, head);

cout << "\n";

tmp.clear();

}

if(group[head].academ\_start.day == "\0")

{

cout << "Изменить номер зачетной книжки студента? (1-да; 2-нет): ";

cin >> ans;

if(ans == 1){

cout << "Введите номер зачетной книжки: ";

cin >> tmp;

group[head].book\_number = tmp;

cout << "\n";

showStud(group, head);

cout << "\n";

tmp.clear();

}

cout << "Изменить возраст студента? (1-да; 2-нет): ";

cin >> ans;

if(ans == 1){

cout << "Введите возраст: ";

cin >> tmp;

group[head].age = tmp;

cout << "\n";

showStud(group, head);

cout << "\n";

tmp.clear();

}

} else {

cout << "Изменить дату начала академического отпуска студента? (1-да; 2-нет): ";

cin >> ans;

if(ans == 1){

cout << "Введите дату начала академического отпуска: ";

cin >> tmp;

for(int i = 0; i < 2; i++){

day += tmp[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

month += tmp[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

year += tmp[k];

k++;

}

k = 0;

group[head].academ\_start.day = day;

group[head].academ\_start.month = month;

group[head].academ\_start.year = year;

day.clear(); month.clear(); year.clear();tmp.clear();

cout << "\n";

showStud(group, head);

cout << "\n";

}

cout << "Изменить дату конца академического отпуска студента? (1-да; 2-нет): ";

cin >> ans;

if(ans == 1){

cout << "Введите дату конца академического отпуска: ";

cin >> tmp;

for(int i = 0; i < 2; i++){

day += tmp[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

month += tmp[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

year += tmp[k];

k++;

}

group[head].academ\_finish.day = day;

group[head].academ\_finish.month = month;

group[head].academ\_finish.year = year;

cout << "\n";

showStud(group, head);

cout << "\n";

}

}

flag = false;

break;

} else {

head = group[head].ptr;

if(head == -1){

break;

}

}

}

return flag;

}

void showStud(LIST\* group, int head){

cout << " " << group[head].book\_number << " " << group[head].fio << " " <<

group[head].age << " " << " ";

if(group[head].academ\_start.day != "\0"){

cout << group[head].academ\_start.day << "." << group[head].academ\_start.month << "." <<

group[head].academ\_start.year << "-" << group[head].academ\_finish.day << "." <<

group[head].academ\_finish.month << "." << group[head].academ\_finish.year;

}

if(group[head].academ){

cout << "(Студент в отпуске)";

}

cout << "\n";

}

bool searchStud(LIST\* group, int gr\_head, int param){

int head = gr\_head, k = 0;

bool flag = true;

string tmp, tmp1, day, month, year;

switch (param)

{

case 1:

cout << "Введите ФИО студента: ";

cin.ignore();

getline(cin,tmp);

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

if(head != -1){

if(group[head].fio == tmp){

showStud(group, head);

flag = false;

}

head = group[head].ptr;

} else{

break;

}

}

break;

case 2:

cout << "Введите номер зачетной книжки студента: ";

cin >> tmp;

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

if(head != -1){

if(group[head].book\_number == tmp){

showStud(group, head);

flag = false;

}

head = group[head].ptr;

} else{

break;

}

}

break;

case 3:

cout << "Введите возраст студента: ";

cin >> tmp;

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

if(head != -1){

if(group[head].age == tmp){

showStud(group, head);

flag = false;

}

head = group[head].ptr;

} else{

break;

}

}

break;

case 4:

cout << "Введите дату начала академического отпуска студента: ";

cin >> tmp;

for(int i = 0; i < 2; i++){

day += tmp[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

month += tmp[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

year += tmp[k];

k++;

}

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

if(head != -1){

if(group[head].academ\_start.day == day && group[head].academ\_start.month == month && group[head].academ\_start.year == year){

showStud(group, head);

flag = false;

}

head = group[head].ptr;

} else{

break;

}

}

break;

case 5:

cout << "Введите дату конца академического отпуска студента: ";

cin >> tmp;

for(int i = 0; i < 2; i++){

day += tmp[k];

k++;

}

k++;

for(int i = 0; i < 2; i++){

month += tmp[k];

k++;

}

k++;

for(int i = 0; i < 4; i++){

year += tmp[k];

k++;

}

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

if(head != -1){

if(group[head].academ\_finish.day == day && group[head].academ\_finish.month == month && group[head].academ\_finish.year == year){

showStud(group, head);

flag = false;

}

head = group[head].ptr;

} else{

break;

}

}

break;

default:

break;

}

return flag;

}

int studAmount(LIST\* group, int gr\_head){

int head = gr\_head, counter = 0;

if(group[head].ptr != -1){

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

if(head == -1){

break;

}

counter += 1;

head = group[head].ptr;

}

} else return 0;

return counter;

}

bool removeStud(LIST\* group, int\* gr\_free, int\* gr\_head, string fio){

int head = \*gr\_head, buf;

bool flag = true;

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

if(group[head].fio == fio){

flag = false;

if(group[head].academ){

cout << "Студент находится в академическом отпуске. Верните его из отпуска и повторите операцию.";

return flag;

}

group[head].book\_number.clear();

group[head].fio.clear();

group[head].age.clear();

group[head].academ\_finish.day.clear();

group[head].academ\_finish.month.clear();

group[head].academ\_finish.year.clear();

group[head].academ\_start.day.clear();

group[head].academ\_start.month.clear();

group[head].academ\_start.year.clear();

if(i == 0){

\*gr\_head = group[head].ptr;

group[head].ptr = \*gr\_free;

\*gr\_free = head;

break;

}

group[buf].ptr = group[head].ptr;

group[head].ptr = \*gr\_free;

\*gr\_free = head;

break;

} else{

buf = head;

head = group[head].ptr;

}

}

return flag;

}

void grOut(LIST\* group, int gr\_head){

int head = gr\_head;

if(group[head].ptr != -1)

{

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

if(head == -1){

break;

}

showStud(group, head);

head = group[head].ptr;

}

} else cout << "Список пуст!\n";

}

void addStud(LIST\* group, int\* gr\_free, int\* gr\_head){

string b\_n, fio, age, date\_s, date\_f;

unsigned short int flag = 0;

int buf = group[\*gr\_free].ptr;

if(\*gr\_head != -1)

{

cout << "Введите данные о студенте \n ФИО: ";

cin.ignore();

getline(cin, fio);

group[\*gr\_free].fio = fio;

if(group[0].num != academ\_num)

{

cout << " Номер зачетной книжки: ";

getline(cin, b\_n);

cout << " Возраст: ";

getline(cin, age);

group[\*gr\_free].book\_number = b\_n;

group[\*gr\_free].age = age;

group[\*gr\_free].ptr = \*gr\_head;

\*gr\_head = \*gr\_free;

\*gr\_free = buf;

} else {

cout << " Дату начала академического отпуска: ";

getline(cin, date\_s);

for(int i = 0; i < 2; i++){

group[\*gr\_free].academ\_start.day += date\_s[flag];

flag++;

}

flag++;

for(int i = 0; i < 2; i++){

group[\*gr\_free].academ\_start.month += date\_s[flag];

flag++;

}

flag++;

for(int i = 0; i < 4; i++){

group[\*gr\_free].academ\_start.year += date\_s[flag];

flag++;

}

flag = 0;

cout << " Дату окончания академического отпуска: ";

getline(cin, date\_f);

for(int i = 0; i < 2; i++){

group[\*gr\_free].academ\_finish.day += date\_f[flag];

flag++;

}

flag++;

for(int i = 0; i < 2; i++){

group[\*gr\_free].academ\_finish.month += date\_f[flag];

flag++;

}

flag++;

for(int i = 0; i < 4; i++){

group[\*gr\_free].academ\_finish.year += date\_f[flag];

flag++;

}

group[\*gr\_free].ptr = \*gr\_head;

\*gr\_head = \*gr\_free;

\*gr\_free = buf;

}

} else cout << "Нет места для записи!\n";

}

int grForm(LIST\* group, int len){

ifstream stream("groups.txt");

string buf = "0";

unsigned int flag, amount = 0;

while(getline(stream, buf))

{

if((buf[0] - '0') == group[0].num)

{

for(int i = 0; i < buf.length(); i++)

{

if(buf[i] == ';')

{

amount++;

}

};

amount++;

if(amount > stud){

cout << "Недостаточно места в массиве для группы №" << group[0].num << "! Выход из программы!\n";

exit(0);

}

for(int i = 2; i < buf.length(); i++)

{

if(buf[i] == '(')

{

flag = i;

break;

}

};

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

{

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].book\_number += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].fio += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].age += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].academ\_start.day += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].academ\_start.month += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].academ\_start.year += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].academ\_finish.day += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ',')

{

flag = i;

break;

}

group[j].academ\_finish.month += buf[i];

};

for(int i = flag+1; i < buf.length(); i++)

{

if(buf[i] == ';')

{

group[j].ptr = j + 1;

flag = i;

break;

} else if(buf[i] == ')'){

group[j].ptr = -1;

flag = i;

break;

}

group[j].academ\_finish.year += buf[i];

};

}

for(int i = amount; i < stud-1; i++)

{

group[i].ptr = i+1;

}

group[stud-1].ptr = -1;

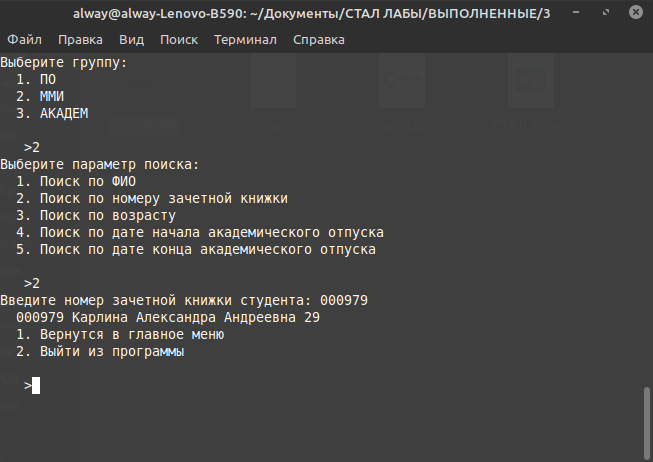
}

}

stream.close();

return amount+1;

}

2) ***Пример работы программы:***

