Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет "Высшая школа экономики"»

Московский институт электроники и математики им. А.Н. Тихонова
НИУ ВШЭ
Департамент компьютерной инженерии

Курс: Алгоритмизация и программирование

Раздел	Max оценка	Итог. оценка
Работа программы	1	
Тесты	1	
Правильность алгоритма	3	
Ответы на вопросы	2	
Дополнительное задание	3	
Итого	10	

ОТЧЁТ
по лабораторной работе №10

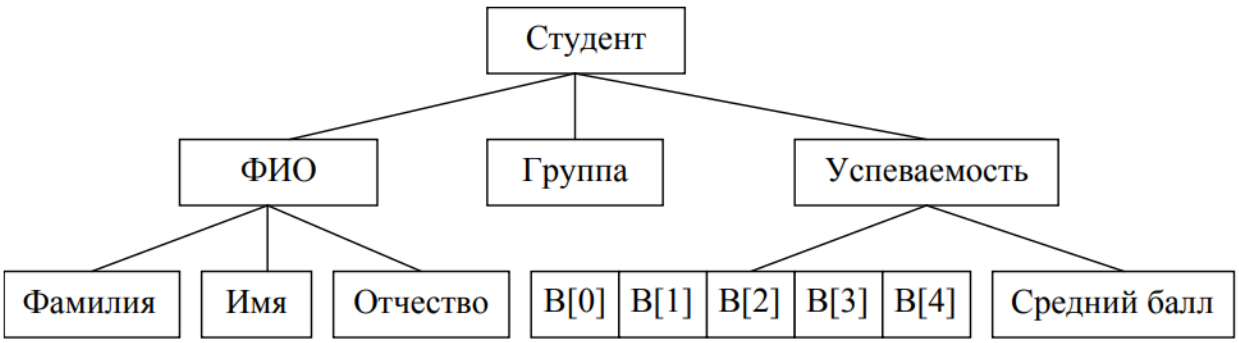
Студент: Боев Андрей Олегович
Группа: БИВ238
Вариант: №412 (1, 1, 7)
Руководитель: Литвиненко Алексей Михайлович
Оценка:
Дата сдачи:

МОСКВА 2024

Оглавление

Задание

Написать программу, которая вводит из файла структуры вида:



Имя файла задает пользователь. Массив баллов B[0:4] содержит данные о результатах сдачи экзаменов, по 10-балльной шкале. Каждое поле структуры занимает в файле одну строку, а массив оценок размещается на отдельной строке. Средний балл не записан в файле, а вычисляется в процессе чтения данных. Структуры размещаются в **стеке**, реализованном с помощью линейного списка.

Для полученного списка программа выполняет следующие действия: вставить новую запись, задаваемую пользователем, перед каждой записью с данными о студенте из указанной группы.

Листинг

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;

struct student {
    struct student* next = NULL;
    struct { string first, last, patronymic; } name;
    string group;
    struct { int B[5] = { 0 }; float average; } grades;
};

struct student* read_fl();
void input_student(student* student);
bool insert_student(struct student** lst, struct student* new_student, string
group);
void stack_to_file(struct student* lst);
void print_stack(struct student* lst);
string read_str();
void free_memory(student* lst);


int main()
{
    struct student** lst = new student*;
    *lst = read_fl();
    print_stack(*lst);

    string group;
    cout << "Input the group number: " << endl;
    group = read_str();
    student* new_student = new student;
    input_student(new_student);
    if (insert_student(lst, new_student, group))
        print_stack(*lst);
    else
        cout << "No students with the " << group << " group were found." << endl;
    stack_to_file(*lst);
    free_memory(*lst);
    return 0;
}

struct student* read_fl() {
    int i = 1;
    struct student* lst = NULL, * curr = new student;
    string str;

    string filename = ".txt";
    cout << "Input name of the file with data: " << endl;
    filename = read_str();
    cout << endl;

    fstream fl;
    fl.open(filename);
    if (!fl) {
        cout << "File not found." << endl;
        exit(1);
    }
    while (!fl.eof()) {
        getline(fl, str);
        switch (i)
        {
```

```
            case 1:
                curr->name.last = str;
                break;
            case 2:
                curr->name.first = str;
                break;
            case 3:
                curr->name.patronymic = str;
                break;
            case 4:
                curr->group = str;
                break;
            case 5:
                int j = 0;
                for (int k = 0; k < str.length(); k++) {
                    if (str[k] == ',')
                        continue;
                    if (str[k] == ' ')
                        j++;
                    else
                        curr->grades.B[j] = curr->grades.B[j] * 10 + (str[k] - 48);
                }
                break;
            }
            if (i == 5) {
                i = 1;
                curr->grades.average = (float)(curr->grades.B[0] + curr->grades.B[1] +
                    curr->grades.B[2] + curr->grades.B[3] + curr->grades.B[4]) / 5;
                curr->next = lst;
                lst = curr;
                curr = new student;
            }
            else i++;
        }
        fl.close();
        return lst;
}

void input_student(student* student) {
        cout << "Last name:     ";
        student->name.last = read_str();
        cout << "First name:    ";
        student->name.first = read_str();
        cout << "Patronymic:    ";
        student->name.patronymic = read_str();
        cout << "Group:         ";
        student->group = read_str();
        cout << "Grades:        ";
        string grades = read_str();
        int j = 0;
        for (int k = 0; k < grades.length(); k++) {
            if (grades[k] == ',')
                continue;
            if (grades[k] == ' ') {
                j++;
            }
            else {
                student->grades.B[j] = student->grades.B[j] * 10 + (grades[k] - 48);
            }
        }
        student->grades.average = (float)(student->grades.B[0] + student->grades.B[1] +
            student->grades.B[2] + student->grades.B[3] + student->grades.B[4]) / 5;
}
```

```
bool insert_student(struct student** lst, struct student* new_student, string group)
{
    bool flag = false;
    struct student* curr, * pred;
    curr = pred = *lst;
    while (curr)
    {
        if (curr->group == group)
        {
            struct student* temp = new student;
            temp->name = new_student->name;
            temp->group = new_student->group;
            temp->grades = new_student->grades;
            if (curr == pred) {
                temp->next = *lst;
                *lst = temp;
            }
            else {
                temp->next = curr;
                pred->next = temp;
            }
            flag = true;
        }
        pred = curr;
        curr = curr->next;
    }
    return flag;
}

void stack_to_file(struct student* lst) {
    struct student* p = lst;
    ofstream fo;
    string filename_out;
    cout << "Input name of the file with results: " << endl;
    filename_out = read_str();
    cout << endl;
    if (filename_out.substr(filename_out.length() - 4) != ".txt") {
        cout << "Wrong filename. " << endl;
        exit(1);
    }
    else {
        fo.open(filename_out);
        while (p) {
            fo << p->name.last << endl;
            fo << p->name.first << endl;
            fo << p->name.patronymic << endl;
            fo << p->group << endl;
            for (int i = 0; i < 4; ++i)
                fo << p->grades.B[i] << " ";
            fo << endl << p->grades.average << endl << endl;
            p = p->next;
        }
    }
}

string read_str()
{
    string s = "";
    do
    {
        getline(cin, s);
    } while (!s.size());

    return s;
```

```
}

void print_stack(struct student* lst) {
    cout << endl;
    struct student* p = lst;
    while (p) {
        cout << "Last name:      " << p->name.last << endl;
        cout << "First name:     " << p->name.first << endl;
        cout << "Patronymic:     " << p->name.patronymic << endl;
        cout << "Group:          " << p->group << endl;
        cout << "Grades:         ";
        for (int i = 0; i < 5; ++i)
            cout << p->grades.B[i] << " ";
        cout << endl;
        cout << "GPA:            " << p->grades.average << endl << endl;
        p = p->next;
    }
}

void free_memory(student* lst) {
    student* p = lst, * next = lst;
    while (p) {
        next = p->next;
        delete p;
        p = next;
    }
    cout << "The memory is cleared." << endl;
}
```

Тесты

№	Исходные данные	Результаты
1	Input name of the file with data: .txt	File not found.

2	Input name of the file with data:
	test.txt
	
	Last name: Ivanov
	First name: Alexander
	Patronymic: Maksimovich
	Group: BIV249
	Grades: 10 3 9 10 5
	GPA: 7.4
	
	Last name: Romanov
	First name: Maksim
	Patronymic: Vladimirovich
	Group: BIV249
	Grades: 8 5 3 9 5
	GPA: 6
	
	Last name: Boev
	First name: Andrey
	Patronymic: Olegovich
	Group: BIV238
	Grades: 1 2 3 4 5
	GPA: 3
	
	Input the group number:
	BIV256
	Last name: Belov
	First name: Oleg
	Patronymic: Vladimirovich
	Group: BIV234
	Grades: 10, 6, 7, 8, 10
	
	Input name of the file with results:
	out.txt
	
	The memory is cleared.
		No students with the BIV256 group were found.
		
		out.txt {
		Ivanov
		Alexander
		Maksimovich
		BIV249
		10 3 9 10
		7.4
		
		Romanov
		Maksim
		Vladimirovich
		BIV249
		8 5 3 9
		6
		
		Boev
		Andrey
		Olegovich
		BIV238
		1 2 3 4
		3
		}

3	Input name of the file with data: test.txt Last name: Ivanov First name: Alexander Patronymic: Maksimovich Group: BIV249 Grades: 10 3 9 10 5 GPA: 7.4	Belob Oleg Vladimirovich BIV256 5 6 7 8 7.2 Ivanov Alexander Maksimovich BIV249

	Last name: Romanov First name: Maksim Patronymic: Vladimirovich Group: BIV249 Grades: 8 5 3 9 5 GPA: 6 Last name: Boev First name: Andrey Patronymic: Olegovich Group: BIV238 Grades: 1 2 3 4 5 GPA: 3 Input the group number: BIV249 Last name: Belob First name: Oleg Patronymic: Vladimirovich Group: BIV256 Grades: 5, 6, 7, 8, 10 Last name: Belob First name: Oleg Patronymic: Vladimirovich Group: BIV256 Grades: 5 6 7 8 10 GPA: 7.2 Last name: Ivanov First name: Alexander Patronymic: Maksimovich Group: BIV249 Grades: 10 3 9 10 5 GPA: 7.4 Last name: Belob First name: Oleg Patronymic: Vladimirovich Group: BIV256 Grades: 5 6 7 8 10 GPA: 7.2 Last name: Romanov First name: Maksim Patronymic: Vladimirovich Group: BIV249 Grades: 8 5 3 9 5 GPA: 6 Last name: Boev First name: Andrey	10 3 9 10 7.4 Belob Oleg Vladimirovich BIV256 5 6 7 8 7.2 Romanov Maksim Vladimirovich BIV249 8 5 3 9 6 Boev Andrey Olegovich BIV238 1 2 3 4 3

	Patronymic: Olegovich Group: BIV238 Grades: 1 2 3 4 5 GPA: 3 Input name of the file with results: out.txt The memory is cleared.	