Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение  
высшего образования   
«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчет защищен с оценкой \_\_\_\_\_

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Отчет

По лабораторной работе №2

**«(Реализация АТД на языке С)»**

по дисциплине «Программирование - 3семестр»

Студент группы ПИ-02 Гранкин А.А.

Преподаватель Троицкий В.С.

Барнаул 2021

Задание:

Модифицировать программу из ЛР № 2.   Заменить struct на class; Функции становятся методами класса; Поля становятся private, доступ к ним при необходимости осуществляется через методы public; Дополнительно в main продемонстрировать работу с динамическим массивом объектов класса и с массивом динамических объектов класса; применение операторов new, delete, delete[]; использование (\*a) и ->.

Решение:

Файл Teacher.h

#pragma once

#include <iostream>

#include <stdio.h>

#include <string>

#include <locale.h>

using namespace std;

// Структура Учитель

class Teacher

{

private:

string Fam = ""; // Фамилия

string Name = ""; // Имя

string Otch = ""; // Отчество

public:

// Конструктор

Teacher();

// Фун-ии получения данных из полей

string get\_t\_F();

string get\_t\_N();

string get\_t\_O();

// Деструктор

~Teacher();

// Инициализация структуры Учитель

void set\_t(string Fam, string Name, string Otch);

// Получение данных

Teacher get\_t();

// Ввод учителя

void inp\_Teach();

// Ввод учителя

void out\_Teach();

// Расширение динамического массива

Teacher\* Exten\_mas(Teacher\* &old\_mas, int len, int difference);

};

Файл Teacher.cpp

#include "Teacher.h"

// Конструктор

Teacher::Teacher()

{

Fam = "";

Name = "";

Otch = "";

}

// Фун-ии получения данных из полей

string Teacher::get\_t\_F()

{

return this->Fam;

}

string Teacher::get\_t\_N()

{

return this->Name;

}

string Teacher::get\_t\_O()

{

return this->Otch;

}

// Инициализация структуры Учитель

void Teacher::set\_t(string Fam\_s, string Name\_s, string Otch\_s)

{

Fam = Fam\_s;

Name = Name\_s;

Otch = Otch\_s;

}

// Получение данных

Teacher Teacher::get\_t()

{

return \*this;

}

// Ввод учителя

void Teacher::inp\_Teach()

{

cout << "Введите Фамилию учителя: ";

cin >> Fam;

cout << "Введите имя учителя: ";

cin >> Name;

cout << "Введите отчество учителя: ";

cin >> Otch;

}

// Ввод учителя

void Teacher::out\_Teach()

{

cout << "ФИО учителя: " << Fam << " " << Name << " " << Otch << endl;

}

// Расширение динамического массива

// Указатель на старый массив, длина старого массива, во сколько раз больше новый массив

Teacher\* Teacher::Exten\_mas(Teacher\* &old\_mas, int len, int difference)

{

Teacher\* point;

if (len > 0)

{

if (difference > 0)

{

Teacher\* new\_mas = new Teacher[len + difference];

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

{

\*new\_mas = \*old\_mas;

}

delete[] old\_mas;

old\_mas = new\_mas;

point = old\_mas;

}

else

point = old\_mas;

}

else

point = old\_mas;

return point;

}

// Деструктор

Teacher::~Teacher()

{

;

}

Файл Lesson.h

#pragma once

#include "Teacher.h"

// Структура Уроки

class Lessons

{

private:

string Name = ""; // Название предмета

Teacher Teach; // Учитель

public:

// Конструктор

Lessons();

// Фун-ии получения данных из полей

string get\_l\_N();

Teacher get\_l\_T();

// Деструктор

~Lessons();

// Инициализация структуры Уроки

void set\_l(string Name\_Less, Teacher t);

// Получение данных

Lessons get\_l();

// Обнуление полей

void null\_l();

// Ввод названия урока

void inp\_Less(Teacher t);

// Фун-ия расширения динамического массива

Lessons\* Extern\_Less(Lessons\* &old\_mas, int len, int dif);

// Вывод уроков

void out\_Less();

};

Файл Lesson.cpp

#include "Lesson.h"

// Конструктор

Lessons::Lessons()

{

this->Name = "";

this->Teach.set\_t("", "", "");

}

// Инициализация структуры Уроки

void Lessons::set\_l(string Name\_Less, Teacher t)

{

Name = Name\_Less;

Teach = t;

}

// Обнуление полей

void Lessons::null\_l()

{

this->Name = "";

this->Teach.set\_t("", "", "");

}

// Получение данных

Lessons Lessons::get\_l()

{

return \*this;

}

// Фун-ии получения данных из полей

string Lessons::get\_l\_N()

{

return this->Name;

}

Teacher Lessons::get\_l\_T()

{

return this->Teach;

}

// Ввод названия урока

void Lessons::inp\_Less(Teacher t)

{

cout << "Введите название предмета: ";

cin >> Name;

cout << endl;

Teach = t;

}

// Вывод уроков, передаем адрес, где будет храниться информация

void Lessons::out\_Less()

{

cout << "Название предмета: " << Name << endl;

Teach.out\_Teach();

}

// Расширение динамического массива

// Указатель на старый массив, длина старого массива, во сколько раз больше новый массив

Lessons\* Lessons::Extern\_Less(Lessons\* &old\_mas, int len, int dif)

{

Lessons\* point;

if (len > 0)

{

if (dif > 0)

{

Lessons\* new\_mas = new Lessons[len + dif];

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

{

\*new\_mas = \*old\_mas;

}

delete[] old\_mas;

old\_mas = new\_mas;

point = old\_mas;

}

else

point = old\_mas;

}

else

point = old\_mas;

return point;

}

// Деструктор

Lessons::~Lessons()

{

;

}

Файл Mark.h

#pragma once

#include "Lesson.h"

// Структура Оценка

class Mark

{

private:

int Value = 0; // Оценка

public:

// Конструктор

Mark();

// Деструктор

~Mark();

// Инициализация структуры Оценка

void set\_m(int v);

// Получение оценки

int get\_m();

// Ввод данных в структуру Оценка

void inp\_Mark();

// Фун-ия расширения динамического массива

Mark\* Extern\_Mark(Mark\* &old\_mas, int len, int dif);

// Вывод данных из структуры

void out\_Mark();

};

Файл Mark.cpp

#include "Mark.h"

// Конструктор

Mark::Mark()

{

Value = 0;

}

// Инициализация структуры Оценка

void Mark::set\_m(int v)

{

Value = v;

}

// Получение оценки

int Mark::get\_m()

{

return Value;

}

// Ввод данных в структуру Оценка

void Mark::inp\_Mark()

{

string mark;

cout << "Введите оценку: ";

cin >> mark;

Value = stoi(mark);

cout << endl;

}

// Вывод данных из структуры

void Mark::out\_Mark()

{

cout << "Оценка: " << Value << endl;

}

// Фун-ия расширения динамического массива

Mark\* Mark::Extern\_Mark(Mark\* &old\_mas, int len, int dif)

{

Mark\* ls;

if (len > 0)

{

if (dif > 0)

{

Mark\* new\_mas = new Mark[len + dif];

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

{

\*new\_mas = \*old\_mas;

}

delete[] old\_mas;

old\_mas = new\_mas;

ls = old\_mas;

}

else

ls = old\_mas;

}

else

ls = old\_mas;

return ls;

}

// Деструктор

Mark::~Mark()

{

;

}

Файл Student.h

#pragma once

#include "Mark.h"

// Структура Ученик

class Student

{

private:

string Fam = ""; // Фамилия

string Name = ""; // Имя

string Otch = ""; // Отчество

Mark mas\_Marks[20]; // Массив оценок

Lessons mas\_Less[20]; // Массив уроков

int i = 0; // Текущее кол-во уроков

public:

// Конструктор

Student();

// Деструктор

~Student();

// Функция получения

Student get\_s();

// Функции получения значений полей

string get\_s\_F();

string get\_s\_N();

string get\_s\_O();

Mark\* get\_s\_M();

Lessons\* get\_s\_L();

// Очищение всех полей структуры Ученик (Student)

void null\_s();

// Инициализация структуры Ученик

void set\_s(string Fam, string Name, string Otch, Mark mas\_m[20], Lessons mas\_l[20]);

// Ввод ФИО ученика в структуру

void inp\_FIO\_Stud();

// Добавление к ученику урока

int add\_Less\_to\_Stud(Lessons ls);

// Добавление к ученику урока и оценки

int add\_LM\_to\_Stud(Lessons ls, Mark m);

// Добавление к ученику оценки к конкретному предмету

int add\_Mark\_to\_Stud(Lessons ls, Mark m);

// Вывод ученика

// Передаем ученика и флаг для различного вывода уроков и оценок

// fl\_out (0 или 1)

void out\_Stud(int fl\_out);

// Расширение динамического массива

Student\* Exten\_Stud(Student\* &old\_mas, int len, int difference);

};

Файл Student.cpp

#include "Student.h"

// Конструктор

Student::Student()

{

Fam = "";

Name = "";

Otch = "";

Lessons L;

Mark M;

L.null\_l();

M.set\_m(0);

i = 0;

for (int i = 0; i < 20; i++)

{

mas\_Less[i] = L;

}

for (int i = 0; i < 20; i++)

{

mas\_Marks[i] = M;

}

}

// Очищение всех полей структуры Ученик (Student)

void Student::null\_s()

{

Fam = "";

Name = "";

Otch = "";

Lessons L;

Mark M;

L.null\_l();

M.set\_m(0);

this->i = 0;

for (int i = 0; i < 20; i++)

{

mas\_Less[i] = L;

}

for (int i = 0; i < 20; i++)

{

mas\_Marks[i] = M;

}

}

// Функция получения

Student Student::get\_s()

{

return \*this;

}

// Функции получения значений полей

string Student::get\_s\_F()

{

return this->Fam;

}

string Student::get\_s\_N()

{

return this->Name;

}

string Student::get\_s\_O()

{

return this->Otch;

}

Mark\* Student::get\_s\_M()

{

return this->mas\_Marks;

}

Lessons\* Student::get\_s\_L()

{

return this->mas\_Less;

}

// Инициализация структуры Ученик

void Student::set\_s(string Fam, string Name, string Otch, Mark mas\_m[20], Lessons mas\_l[20])

{

Fam = Fam;

Name = Name;

Otch = Otch;

this->i = 0;

int i = 0;

while (mas\_Less[i].get\_l\_N() != "")

{

mas\_Less[i] = mas\_l[i];

i++;

this->i++;

}

for (int i = 0; i < this->i; i++)

{

mas\_Marks[i] = mas\_m[i];

}

}

// Ввод ФИО ученика в структуру

void Student::inp\_FIO\_Stud()

{

cout << "Введите Фамилию ученика: ";

cin >> Fam;

cout << "Введите Имя ученика: ";

cin >> Name;

cout << "Введите Отчество ученика: ";

cin >> Otch;

}

// Добавление к ученику урока

// 0 - не удачно, 1 - удачно

int Student::add\_Less\_to\_Stud(Lessons ls)

{

int fl = 0;

for (int i = 0; (i < 20) && (fl == 0); i++)

{

if (mas\_Less[i].get\_l\_N() == "")

{

mas\_Less[i] = ls;

fl = 1;

}

}

return fl;

}

// Добавление к ученику урока и оценки

// 0 - не удачно, 1 - удачно

int Student::add\_LM\_to\_Stud(Lessons ls, Mark m)

{

int fl = 0;

for (int i = 0; (i < 20) && (fl == 0); i++)

{

if (mas\_Less[i].get\_l\_N() == "")

{

mas\_Less[i] = ls;

mas\_Marks[i] = m;

fl = 1;

}

}

return fl;

}

// Добавление к ученику оценки к конкретному предмету

// 0 - не удачно, 1 - удачно

int Student::add\_Mark\_to\_Stud(Lessons ls, Mark m)

{

int fl = 0;

for (int i = 0; (i < 20) && (fl == 0); i++)

{

if (mas\_Less[i].get\_l\_N() == ls.get\_l\_N())

{

if (mas\_Less[i].get\_l\_T().get\_t\_F() == ls.get\_l\_T().get\_t\_F())

{

if (mas\_Less[i].get\_l\_T().get\_t\_N() == ls.get\_l\_T().get\_t\_N())

{

if (mas\_Less[i].get\_l\_T().get\_t\_O() == ls.get\_l\_T().get\_t\_O())

{

mas\_Marks[i] = m;

fl = 1;

}

}

}

}

}

return fl;

}

// Вывод ученика

// Передаем ученика и флаг для различного вывода уроков и оценок

// fl\_out (0 или 1)

void Student::out\_Stud(int fl\_out)

{

cout << "ФИО ученика: " << Fam << " " << Name << " " << Otch << endl;

if (fl\_out == 0)

{

cout << "Уроки:" << endl;

for (int i = 0; i < 20; i++)

{

mas\_Less[i].out\_Less();

}

cout << "Оценки:" << endl;

for (int i = 0; i < 20; i++)

{

mas\_Marks[i];

}

}

else

{

cout << "Уроки:" << endl;

int i = 0;

while (mas\_Less[i].get\_l\_N() != "")

{

cout << " \* " << mas\_Less[i].get\_l\_N() << " \* " << mas\_Marks[i].get\_m() << " \* " << endl;

i++;

}

}

}

// Расширение динамического массива

// Указатель на старый массив, длина старого массива, во сколько раз больше новый массив

Student\* Student::Exten\_Stud(Student\* &old\_mas, int len, int difference)

{

Student\* point;

if (len > 0)

{

if (difference > 0)

{

Student\* new\_mas = new Student[len + difference];

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

{

\*new\_mas = \*old\_mas;

}

delete[] old\_mas;

old\_mas = new\_mas;

point = old\_mas;

}

else

point = old\_mas;

}

else

point = old\_mas;

return point;

}

// Деструктор

Student::~Student()

{

;

}

Файл Class.h

#pragma once

#include "Student.h"

// Структура Класс

struct Class\_

{

private:

string Name = ""; // Название класса

string year = ""; // Год обучения

int i = 0; // Текущее кол-во учеников

Student mas\_Stud[32]; // Массив учеников

public:

// Конструктор

Class\_();

// Деструктор

~Class\_();

// Очищение полей структуры Класс

void null\_c();

// Инициализация структуры Класс

void set\_c(string Name, string Year, Student\* mas[32]);

// Получение данных

Class\_ get\_c();

// Функции получения значений полей

string get\_c\_n();

string get\_c\_y();

Student\* get\_c\_s();

// Ввод данных в структуру Класс

void inp\_Class\_();

// Добавление к классу ученика

int add\_St\_to\_Cl(Student st);

// Выод данных из структуры Класс

// fl\_out (0 или 1)

void out\_Class\_(int fl\_out);

// Функция по нахождению лучших учеников в классе

int Best\_Stud\_inClass();

// Функция по нахождению худших учеников в классе

int Bed\_Stud\_inClass();

// Расширение динамического массива

Class\_\* Exten\_Stud(Class\_\* &old\_mas, int len, int difference);

};

Файл Class.cpp

#include "Class.h"

// Конструктор

Class\_::Class\_()

{

Name = "";

year = "";

i = 0;

Student S;

S.null\_s();

for (int i = 0; i < 32; i++)

{

mas\_Stud[i] = S;

}

}

// Очищение полей структуры Класс

void Class\_::null\_c()

{

Name = "";

year = "";

i = 0;

Student S;

S.null\_s();

for (int i = 0; i < 32; i++)

{

mas\_Stud[i] = S;

}

}

// Инициализация структуры Класс

void Class\_::set\_c(string Name, string Year, Student\* mas[32])

{

Name = Name;

year = Year;

i = 0;

Student S;

S.null\_s();

for (int i = 0; i < 32; i++)

{

mas\_Stud[i] = \*mas[i];

}

}

// Получение данных

Class\_ Class\_::get\_c()

{

return \*this;

}

// Функции получения значений полей

string Class\_::get\_c\_n()

{

return Name;

}

string Class\_::get\_c\_y()

{

return year;

}

Student\* Class\_::get\_c\_s()

{

return mas\_Stud;

}

// Ввод данных в структуру Класс

void Class\_::inp\_Class\_()

{

cout << "Введите название класса: ";

cin >> Name;

cout << endl;

cout << "Введите год обучения: ";

cin >> year;

cout << endl;

}

// Добавление к классу ченика

int Class\_::add\_St\_to\_Cl(Student st)

{

int fl = 0;

if (i < 20)

{

mas\_Stud[i] = st;

i++;

fl = 1;

}

return fl;

}

// Вывод данных из структуры Класс

// fl\_out (0 или 1)

void Class\_::out\_Class\_(int fl\_out)

{

cout << "Название класса: " << Name << endl;;

cout << "Год обучения: " << year << endl;

cout << endl;

cout << "Ученики:" << endl;

if (fl\_out == 0)

{

int i = 0;

while (mas\_Stud[i].get\_s\_F() != "")

{

mas\_Stud[i].out\_Stud(0);

i++;

}

}

else

{

int i = 0;

while (mas\_Stud[i].get\_s\_F() != "")

{

mas\_Stud[i].out\_Stud(1);

i++;

}

}

}

// Функция по нахождению лучших учеников в классе

int Class\_::Best\_Stud\_inClass()

{

double mas\_m[32];

int exit = 0;

for (int i = 0; i < 32; i++)

{

mas\_m[i] = 0;

}

int i = 0;

int k = 0;

int sum = 0;

while (mas\_Stud[i].get\_s\_F() != "")

{

int j = 0;

k = 0;

sum = 0;

while (mas\_Stud[i].get\_s\_L()[j].get\_l\_N() != "")

{

sum += mas\_Stud[i].get\_s\_M()[j].get\_m();

k++;

j++;

}

mas\_m[i] = double(sum) / k;

i++;

}

if (i == 0)

exit = -1;

else

{

double maxM = 0;

maxM = mas\_m[0];

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

{

if (maxM < mas\_m[j])

maxM = mas\_m[j];

}

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

{

if (mas\_m[j] == maxM)

mas\_Stud[j].out\_Stud(1);

}

}

return exit;

}

// Функция по нахождению худших учеников в классе

int Class\_::Bed\_Stud\_inClass()

{

double mas\_m[32];

int exit = 0;

for (int i = 0; i < 32; i++)

{

mas\_m[i] = 0;

}

int i = 0;

int k = 0;

int sum = 0;

while (mas\_Stud[i].get\_s\_F() != "")

{

int j = 0;

k = 0;

sum = 0;

while (mas\_Stud[i].get\_s\_L()[j].get\_l\_N() != "")

{

sum += mas\_Stud[i].get\_s\_M()[j].get\_m();

k++;

j++;

}

mas\_m[i] = double(sum) / k;

i++;

}

if (i == 0)

exit = -1;

else

{

double minM = 0;

minM = mas\_m[0];

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

{

if (minM > mas\_m[j])

minM = mas\_m[j];

}

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

{

if (mas\_m[j] == minM)

mas\_Stud[j].out\_Stud(1);

}

}

return exit;

}

// Расширение динамического массива

Class\_\* Class\_::Exten\_Stud(Class\_\* &old\_mas, int len, int difference)

{

Class\_\* point;

if (len > 0)

{

if (difference > 0)

{

Class\_\* new\_mas = new Class\_[len + difference];

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

{

\*new\_mas = \*old\_mas;

}

delete[] old\_mas;

old\_mas = new\_mas;

point = old\_mas;

}

else

point = old\_mas;

}

else

point = old\_mas;

return point;

}

// Деструктор

Class\_::~Class\_()

{

;

}

Файл lab2\_t1.cpp

#include "Class.h"

int main()

{

setlocale(LC\_ALL, "Russian");

system("chcp 1251");

system("cls");

cout << "========================Тест структуры Учитель===========================" << endl;

Teacher t\_Chem;

Teacher\* t\_Biology = new Teacher;

t\_Chem.set\_t("", "", "");

t\_Biology->set\_t("", "", "");

cout << "Введите ФИО 2-ух учителей:" << endl;

cout << endl;

t\_Chem.inp\_Teach();

t\_Chem.out\_Teach();

cout << endl;

t\_Biology->inp\_Teach();

(\*t\_Biology).out\_Teach();

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

cout << endl;

cout << endl;

cout << "========================Тест структуры Уроки===========================" << endl;

Lessons Chem;

Lessons\* Biology = new Lessons;

Chem.null\_l();

(\*Biology).null\_l();

cout << "Введите 2 урока:" << endl;

cout << endl;

cout << "Для учителя - ";

t\_Chem.out\_Teach();

Chem.inp\_Less(t\_Chem);

cout << "Вывод:" << endl;

Chem.out\_Less();

cout << endl;

cout << "Для учителя - ";

(\*t\_Biology).out\_Teach();

Biology->inp\_Less(\*t\_Biology);

cout << "Вывод:" << endl;

Biology->out\_Less();

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

cout << endl;

cout << endl;

cout << "=======================Тест структуры Ученик и Оценки============================" << endl;

Student st1;

Student\* st2 = new Student;

Student st3;

st1.null\_s();

(\*st2).null\_s();

st3.null\_s();

cout << "Введите 3 ученика:" << endl;

cout << endl;

cout << "Первый ученик:" << endl;

st1.inp\_FIO\_Stud();

cout << endl;

cout << "Выберите, какие уроки у первого ученика:" << endl;

cout << endl;

Chem.out\_Less();

string str = "";

cout << "Добавить этот предмет ученику? (д - добавить, другой набор символов - нет): ";

cin >> str;

if (str == "д")

{

if (st1.add\_Less\_to\_Stud(Chem))

{

cout << "Урок добавлен!" << endl;

cout << endl;

cout << "Добавить оценку (1,2,3,4,5 - добавить, другой набор символов - нет):" << endl;

str = "";

cin >> str;

Mark mr;

mr.set\_m(0);

mr.set\_m(stoi(str));

if ((mr.get\_m() >= 1) || (mr.get\_m() <= 5))

{

if (st1.add\_Mark\_to\_Stud(Chem, mr))

cout << "Оценка добавлена!" << endl;

else

cout << "Оценка не добавлена!" << endl;

}

}

else

cout << "Урок не добавлен!" << endl;

}

else

cout << "Урок не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод:" << endl;

st1.out\_Stud(1);

cout << endl;

cout << "Выберите, какие уроки у первого ученика:" << endl;

cout << endl;

Biology->out\_Less();

str = "";

cout << "Добавить этот предмет ученику? (д - добавить, другой набор символов - нет): ";

cin >> str;

if (str == "д")

{

if (st1.add\_Less\_to\_Stud(\*Biology))

{

cout << "Урок добавлен!" << endl;

cout << endl;

cout << "Добавить оценку (1,2,3,4,5 - добавить, другой набор символов - нет):" << endl;

str = "";

cin >> str;

Mark mr;

mr.set\_m(0);

mr.set\_m(stoi(str));

if ((mr.get\_m() >= 1) || (mr.get\_m() <= 5))

{

if (st1.add\_Mark\_to\_Stud(\*Biology, mr))

cout << "Оценка добавлена!" << endl;

else

cout << "Оценка не добавлена!" << endl;

}

}

else

cout << "Урок не добавлен!" << endl;

}

else

cout << "Урок не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод:" << endl;

st1.out\_Stud(1);

cout << endl;

cout << "Второй ученик:" << endl;

st2->inp\_FIO\_Stud();

cout << endl;

cout << "Выберите, какие уроки у первого ученика:" << endl;

Chem.out\_Less();

str = "";

cout << "Добавить этот предмет ученику? (д - добавить, другой набор символов - нет): ";

cin >> str;

if (str == "д")

{

if (st2->add\_Less\_to\_Stud(Chem))

{

cout << "Урок добавлен!" << endl;

cout << endl;

cout << "Добавить оценку (1,2,3,4,5 - добавить, другой набор символов - нет):" << endl;

str = "";

cin >> str;

Mark mr;

mr.set\_m(0);

mr.set\_m(stoi(str));

if ((mr.get\_m() >= 1) || (mr.get\_m() <= 5))

{

if ((\*st2).add\_Mark\_to\_Stud(Chem, mr))

cout << "Оценка добавлена!" << endl;

else

cout << "Оценка не добавлена!" << endl;

}

}

else

cout << "Урок не добавлен!" << endl;

}

else

cout << "Урок не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод:" << endl;

(\*st2).out\_Stud(1);

cout << endl;

cout << "Выберите, какие уроки у первого ученика:" << endl;

cout << endl;

(\*Biology).out\_Less();

str = "";

cout << "Добавить этот предмет ученику? (д - добавить, другой набор символов - нет): ";

cin >> str;

if (str == "д")

{

if (st2->add\_Less\_to\_Stud(\*Biology))

{

cout << "Урок добавлен!" << endl;

cout << endl;

cout << "Добавить оценку (1,2,3,4,5 - добавить, другой набор символов - нет):" << endl;

str = "";

cin >> str;

Mark mr;

mr.set\_m(0);

mr.set\_m(stoi(str));

if ((mr.get\_m() >= 1) || (mr.get\_m() <= 5))

{

if (st2->add\_Mark\_to\_Stud(\*Biology, mr))

cout << "Оценка добавлена!" << endl;

else

cout << "Оценка не добавлена!" << endl;

}

}

else

cout << "Урок не добавлен!" << endl;

}

else

cout << "Урок не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод:" << endl;

st2->out\_Stud(1);

cout << endl;

cout << "Третий ученик:" << endl;

st3.inp\_FIO\_Stud();

cout << endl;

cout << "Выберите, какие уроки у первого ученика:" << endl;

Chem.out\_Less();

str = "";

cout << "Добавить этот предмет ученику? (д - добавить, другой набор символов - нет): ";

cin >> str;

if (str == "д")

{

if (st3.add\_Less\_to\_Stud(Chem))

{

cout << "Урок добавлен!" << endl;

cout << endl;

cout << "Добавить оценку (1,2,3,4,5 - добавить, другой набор символов - нет):" << endl;

str = "";

cin >> str;

Mark mr;

mr.set\_m(0);

mr.set\_m(stoi(str));

if ((mr.get\_m() >= 1) || (mr.get\_m() <= 5))

{

if (st3.add\_Mark\_to\_Stud(Chem, mr))

cout << "Оценка добавлена!" << endl;

else

cout << "Оценка не добавлена!" << endl;

}

}

else

cout << "Урок не добавлен!" << endl;

}

else

cout << "Урок не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод:" << endl;

st3.out\_Stud(1);

cout << endl;

cout << "Выберите, какие уроки у первого ученика:" << endl;

cout << endl;

Biology->out\_Less();

str = "";

cout << "Добавить этот предмет ученику? (д - добавить, другой набор символов - нет): ";

cin >> str;

if (str == "д")

{

if (st3.add\_Less\_to\_Stud(\*Biology))

{

cout << "Урок добавлен!" << endl;

cout << endl;

cout << "Добавить оценку (1,2,3,4,5 - добавить, другой набор символов - нет):" << endl;

str = "";

cin >> str;

Mark mr;

mr.set\_m(0);

mr.set\_m(stoi(str));

if ((mr.get\_m() >= 1) || (mr.get\_m() <= 5))

{

if (st3.add\_Mark\_to\_Stud(\*Biology, mr))

cout << "Оценка добавлена!" << endl;

else

cout << "Оценка не добавлена!" << endl;

}

}

else

cout << "Урок не добавлен!" << endl;

}

else

cout << "Урок не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод:" << endl;

st3.out\_Stud(1);

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

cout << endl;

cout << endl;

cout << "=======================Тест структуры Класс============================" << endl;

cout << "Создайте 2 класса учеников:" << endl;

cout << endl;

Class\_ cl1;

Class\_\* cl2 = new Class\_;

cl1.null\_c();

(\*cl2).null\_c();

cout << "Введите данные для первого класса" << endl;

cl1.inp\_Class\_();

cout << endl;

st1.out\_Stud(1);

cout << "Добавить данного ученика в класс? (д - добавить, другой набор символов - нет): ";

str = "";

cin >> str;

if (str == "д")

{

if (cl1.add\_St\_to\_Cl(st1))

{

cout << "Ученик добавлен!" << endl;

cout << endl;

}

else

cout << "Ученик не добавлен!" << endl;

}

else

cout << "Ученик не добавлен!" << endl;

cout << endl;

cout << endl;

st2->out\_Stud(1);

cout << "Добавить данного ученика в класс? (д - добавить, другой набор символов - нет): ";

str = "";

cin >> str;

if (str == "д")

{

if (cl1.add\_St\_to\_Cl(\*st2))

{

cout << "Ученик добавлен!" << endl;

cout << endl;

}

else

cout << "Ученик не добавлен!" << endl;

}

else

cout << "Ученик не добавлен!" << endl;

cout << endl;

cout << endl;

st3.out\_Stud(1);

cout << "Добавить данного ученика в класс? (д - добавить, другой набор символов - нет): ";

str = "";

cin >> str;

if (str == "д")

{

if (cl1.add\_St\_to\_Cl(st3))

{

cout << "Ученик добавлен!" << endl;

cout << endl;

}

else

cout << "Ученик не добавлен!" << endl;

}

else

cout << "Ученик не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод класаа №1" << endl;

cl1.out\_Class\_(1);

cout << endl;

cout << endl;

cout << endl;

cout << "Введите данные для второго класса" << endl;

cl2->inp\_Class\_();

cout << endl;

st1.out\_Stud(1);

cout << "Добавить данного ученика в класс? (д - добавить, другой набор символов - нет): ";

str = "";

cin >> str;

if (str == "д")

{

if (cl2->add\_St\_to\_Cl(st1))

{

cout << "Ученик добавлен!" << endl;

cout << endl;

}

else

cout << "Ученик не добавлен!" << endl;

}

else

cout << "Ученик не добавлен!" << endl;

cout << endl;

cout << endl;

st2->out\_Stud(1);

cout << "Добавить данного ученика в класс? (д - добавить, другой набор символов - нет): ";

str = "";

cin >> str;

if (str == "д")

{

if ((\*cl2).add\_St\_to\_Cl(\*st2))

{

cout << "Ученик добавлен!" << endl;

cout << endl;

}

else

cout << "Ученик не добавлен!" << endl;

}

else

cout << "Ученик не добавлен!" << endl;

cout << endl;

cout << endl;

st3.out\_Stud(1);

cout << "Добавить данного ученика в класс? (д - добавить, другой набор символов - нет): ";

str = "";

cin >> str;

if (str == "д")

{

if ((\*cl2).add\_St\_to\_Cl(st3))

{

cout << "Ученик добавлен!" << endl;

cout << endl;

}

else

cout << "Ученик не добавлен!" << endl;

}

else

cout << "Ученик не добавлен!" << endl;

cout << endl;

cout << endl;

cout << "Вывод класаа №2" << endl;

cout << endl;

(\*cl2).out\_Class\_(1);

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

cout << endl;

cout << endl;

cout << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Функция поиска лучших учеников в классе\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Лучшие ученики класса №1:" << endl;

cl1.Best\_Stud\_inClass();

cout << endl;

cout << endl;

cout << "Лучшие ученики класса №2:" << endl;

(\*cl2).Best\_Stud\_inClass();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

cout << endl;

cout << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Функция поиска худших учеников в классе\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Худшие ученики класса №1:" << endl;

cl1.Bed\_Stud\_inClass();

cout << endl;

cout << endl;

cout << "Худшие ученики класса №2:" << endl;

cl2->Bed\_Stud\_inClass();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << endl;

cout << endl;

cout << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Динамический массив указателей класса Ученик\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

Student\* mas\_s = new Student[3];

mas\_s[0] = st1;

mas\_s[1] = \*st2;

mas\_s[2] = st3;

for (int i = 0; i < 3; i++)

mas\_s[i].out\_Stud(1);

Mark m1;

m1.set\_m(5);

mas\_s->Exten\_Stud(mas\_s, 3, 1);

mas\_s[3].inp\_FIO\_Stud();

mas\_s[3].add\_Less\_to\_Stud(Chem);

mas\_s[3].add\_Mark\_to\_Stud(Chem, m1);

mas\_s[3].out\_Stud(1);

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

system("Pause");

delete t\_Biology;

delete Biology;

delete st2;

delete cl2;

delete[] mas\_s;

}

Тесты программы: