САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

Практическая работа №8

«Объявление и реализация класса. Реализация инкапсуляции. Конструкторы и деструкторы»

Выполнил: Мороз И.О.

Группа К3120

Проверил: Осипов Н.А.

Санкт-Петербург

2018 г.

# Упражнение 1.

1. Для демонстрации раоты с классами был создан новый пустой проект Student01. В нём был создан пустой файл Student01\_main.cpp
2. Был описан класс Student. Методы и функции класса получили public доступ, поля класса получили private доступ

class Student

{

public:

void set\_name(string student\_name)

{

name = student\_name;

}

string get\_name()

{

return name;

}

void set\_last\_name(string student\_last\_name)

{

last\_name = student\_last\_name;

}

string get\_last\_name()

{

return last\_name;

}

void set\_scores(int student\_scores[])

{

for (int i = 0; i < 5; ++i) {

scores[i] = student\_scores[i];

}

}

void set\_average\_score(double ball)

{

average\_score = ball;

}

double get\_average\_score()

{

return average\_score;

}

private:

int scores[5];

double average\_score;

string name;

string last\_name;

};

1. Был создан метод main. В нём была создана переменная student01 класса Student. Для неё был реализован ввод всех значений, а также вывод этих значений в консоль

Student student01;

string name;

string last\_name;

int scores[5];

cout << "Name: ";

getline(cin, name);

cout << "Last name: ";

getline(cin, last\_name);

int sum = 0;

for (int i = 0; i < 5; i++)

{

cout << "Score " << i + 1 << ": ";

cin >> scores[i];

sum += scores[i];

}

student01.set\_name(name);

student01.set\_last\_name(last\_name);

student01.set\_scores(scores);

double average\_score = sum / 5.0;

student01.set\_average\_score(average\_score);

cout << "Average ball for " << student01.get\_name() << " " << student01.get\_last\_name() << " is " << student01.get\_average\_score() << endl;

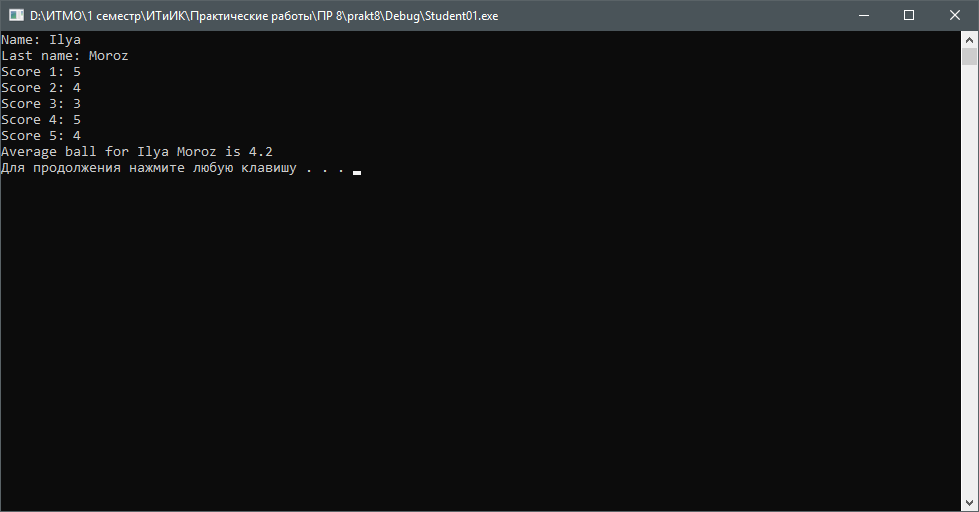


Рисунок 1 – ввод и вывод переменной класса Student

# Упражнение 2.

1. В данном упражнении было произведено разделение реализации и представления. Для этого были созданы файлы student.cpp, student.h, main.cpp
2. Student.cpp – файл, в котором происходит описание методов класса student

#include <string>

#include "student.h"

void Student::set\_name(std::string student\_name)

{

Student::name = student\_name;

}

std::string Student::get\_name()

{

return Student::name;

}

void Student::set\_last\_name(std::string student\_last\_name)

{

Student::last\_name = student\_last\_name;

}

std::string Student::get\_last\_name()

{

return Student::last\_name;

}

void Student::set\_scores(int scores[])

{

for (int i = 0; i < 5; ++i) {

Student::scores[i] = scores[i];

}

}

void Student::set\_average\_score(double ball)

{

Student::average\_score = ball;

}

double Student::get\_average\_score()

{

return Student::average\_score;

}

1. Student.h – файл, в котором находятся прототипы методов класса student

#pragma once /\* Защита от двойного подключения заголовочного файла \*/

#include <string>

using namespace std;

class Student

{

public:

void set\_name(string);

string get\_name();

void set\_last\_name(string);

string get\_last\_name();

void set\_scores(int[]);

void set\_average\_score(double);

double get\_average\_score();

private:

int scores[5];

double average\_score;

string name;

string last\_name;

};

1. В файле main.cpp происходит создание переменной класса Student, далее с ней производят действия, описанные в упражнении 1

#include <iostream>

#include <string>

#include "student.h"

using namespace std;

int main()

{

Student student01;

string name;

string last\_name;

cout << "Name: ";

getline(cin, name);

cout << "Last name: ";

getline(cin, last\_name);

student01.set\_name(name);

student01.set\_last\_name(last\_name);

int scores[5];

int sum = 0;

for (int i = 0; i < 5; ++i) {

cout << "Score " << i + 1 << ": ";

cin >> scores[i];

sum += scores[i];

}

student01.set\_scores(scores);

double average\_score = sum / 5.0;

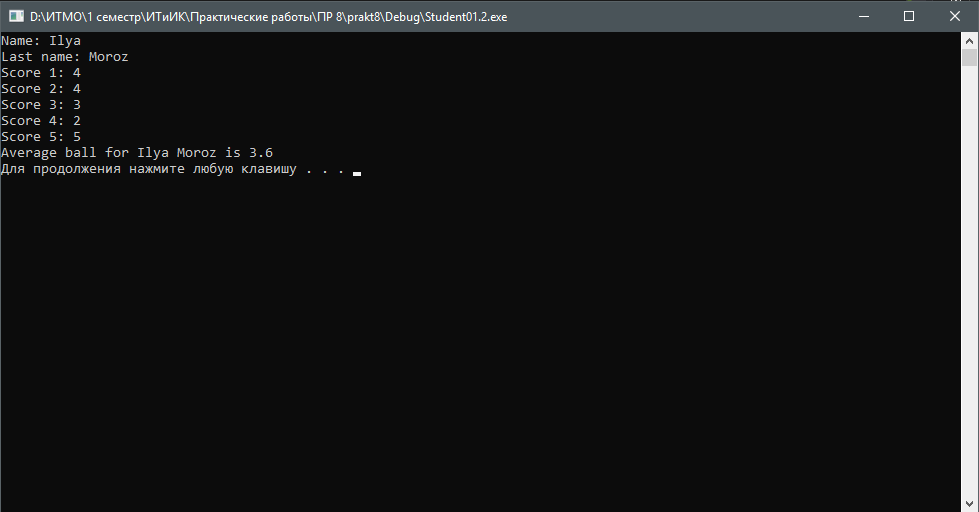
student01.set\_average\_score(average\_score);

cout << "Average ball for " << student01.get\_name() << " " << student01.get\_last\_name() << " is " << student01.get\_average\_score() << endl;

system("pause");

return 0;

}

Рисунок 2 – изменения не повлияли на работу алгоритма

# Упражнение 3.

1. В данном упражнении был создан объект, для которого была выделена память в куче. Для обращения с объектом неоходимо использовать оператор косвенного обращения. После работы с объектом его необходимо удалить из памяти

Student \*student02 = new Student;

student02->set\_name(name);

student02->set\_last\_name(last\_name);

student02->set\_scores(scores);

average\_score = sum / 5.0;

student02->set\_average\_score(average\_score);

cout << "Average ball for " << student02->get\_name() << " " << student02->get\_last\_name() << " is " << student02->get\_average\_score() << endl;

delete student02;

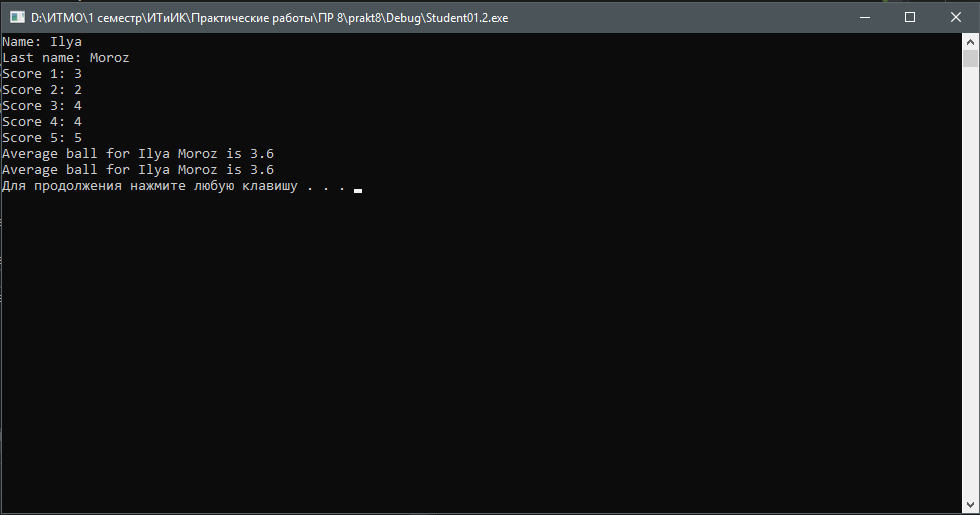


Рисунок 3 – значения объекта и объектной переменной совпадают

# Упражнение 4.

1. Был определён конструктор для класса Student. При создании переменной данного класса необходимо передать значение name и last\_name

Student::Student(string name, string last\_name)

{

Student::set\_name(name);

Student::set\_last\_name(last\_name);

}

//…

string name;

string last\_name;

cout << "Name: ";

getline(cin, name);

cout << "Last name: ";

getline(cin, last\_name);

Student \*student02 = new Student(name, last\_name);

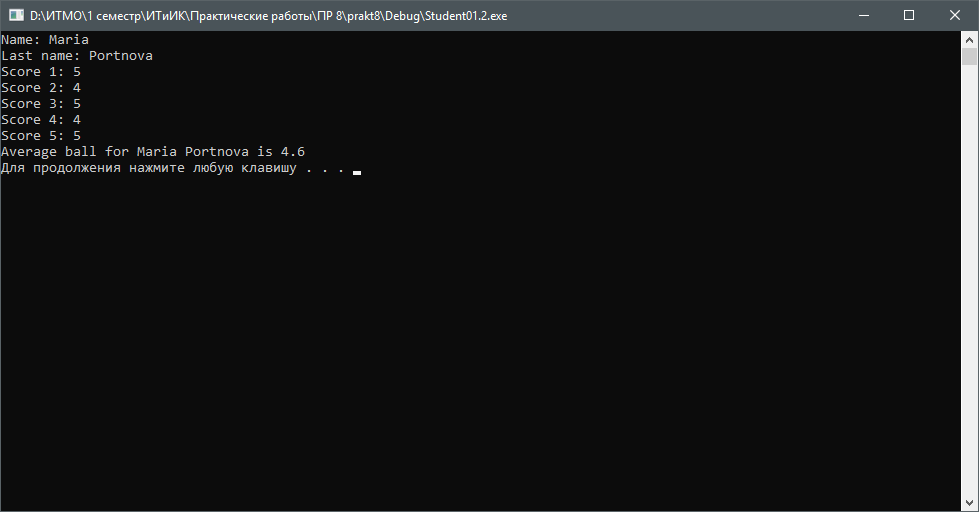


Рисунок 4 – результат работы программы

# Упражнение 5.

1. Был реализован метод сохранения значения переменной класса Student в файл students.txt. Для сохранения значения при удалении объекта был использован деструктор класса. Для работы с файлами использовались методы класса fstream

Student::~Student()

{

Student::save();

}

void Student::save()

{

ofstream fout("students.txt", ios::app);

fout << Student::get\_name() << " " << Student::get\_last\_name() << " ";

for (int i = 0; i < 5; ++i) {

fout << Student::scores[i] << " ";

}

fout << endl;

fout.close();

}

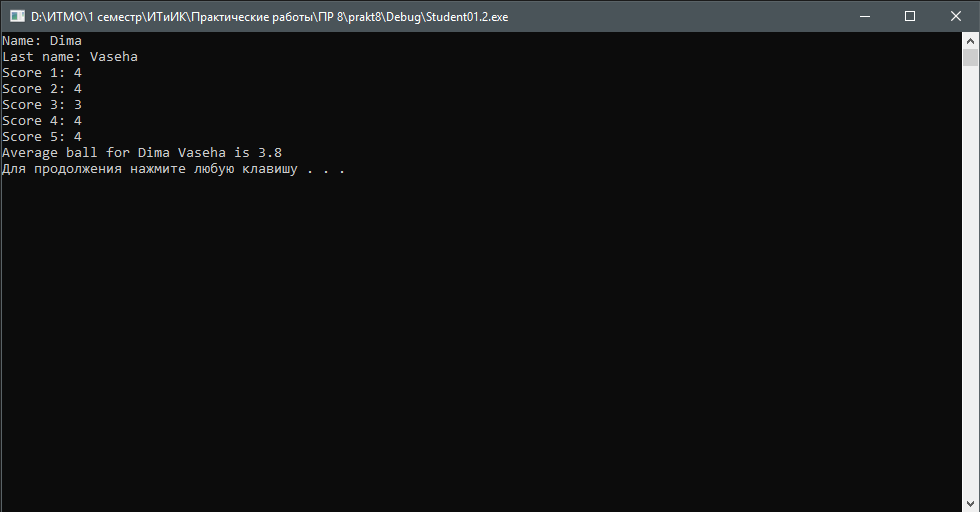


Рисунок 5 – один из нескольких результатов работы программы

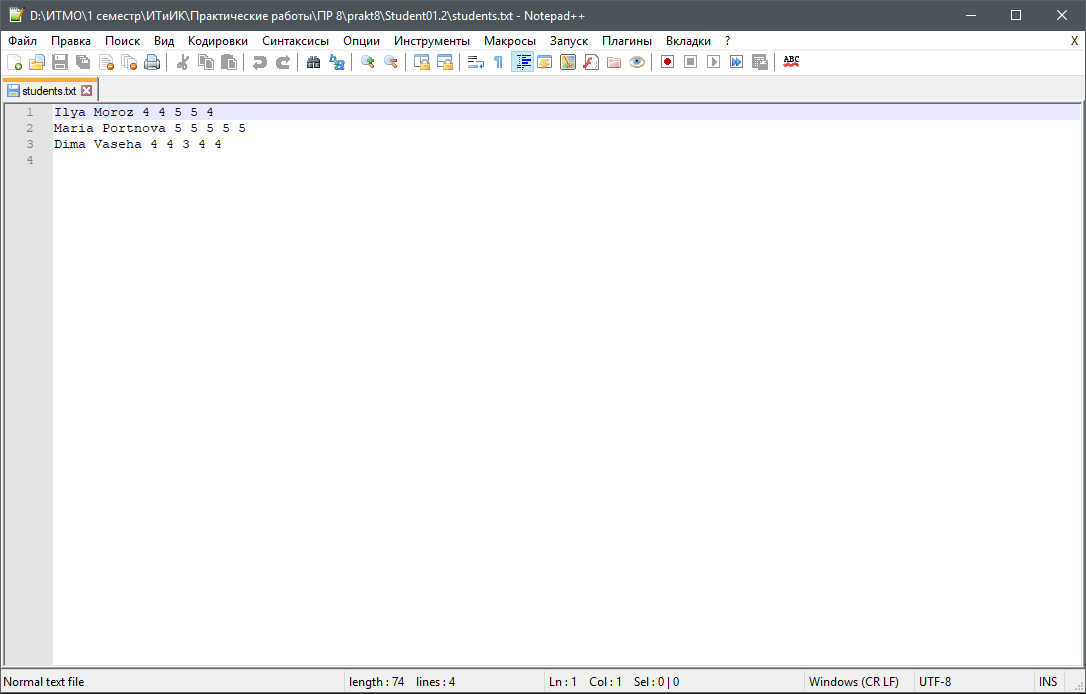


Рисунок 6 – содержимое файла students.txt после работы программы

# Вывод

В данной практической работе были рассмотрены принципы работы с объектами в языке C++. Для этого был создан класс Student, была описана логика взаимодействия с классом. Было произведено разделение реализации и представления. Было рассмотрено создание объекта, конструктора и деструктора, метода для сохранения данных объекта в файл. Полученные навыки были закреплены при выполнении упражнений.

# Контрольные вопросы

1. Какие существуют типы доступа?

Public, private, protected

1. Возможно ли перегружать конструктор и деструктор?

Конструктор – да, деструктор – нет.